Temporal Sequence Distillation: Towards Few-Frame Action Recognition in Videos

Zhaoyang Zhang
Wuhan University
SenseTime Research
zhangzhaoyang@whu.edu.cn

Zhanghui Kuang*
SenseTime Research
kuangzhanghui@sensetime.com

Ping Luo
The Chinese University of Hong Kong
pluo@ie.cuhk.edu.hk

Litong Feng
SenseTime Research
fenglitong@sensetime.com

Wei Zhang
SenseTime Research
wayne.zhang@sensetime.com

ABSTRACT

Video Analytics Software as a Service (VA SaaS) has been rapidly growing in recent years. VA SaaS is typically accessed by users using a lightweight client. Because the transmission bandwidth between the client and cloud is usually limited and expensive, it brings great benefits to design cloud video analysis algorithms with a limited data transmission requirement. Although considerable research has been devoted to video analysis, to our best knowledge, little of them has paid attention to the transmission bandwidth limitation in SaaS. As the first attempt in this direction, this work introduces a problem of few-frame action recognition, which aims at maintaining high recognition accuracy, when accessing only a few frames during both training and test. Unlike previous work that processed dense frames, we present Temporal Sequence Distillation (TSD), which distills a long video sequence into a very short one for transmission. By end-to-end training with 3D CNNs for video action recognition, TSD learns a compact and discriminative temporal and spatial representation of video frames. On Kinetics dataset, TSD+I3D typically requires only 50% of the number of frames compared to I3D [1], a state-of-the-art video action recognition algorithm, to achieve almost the same accuracies. The proposed TSD has three appealing advantages. Firstly, TSD has a lightweight architecture, and can be deployed in the client, *e.g.*, mobile devices, to produce compressed representative frames to save transmission bandwidth. Secondly, TSD significantly reduces the computations to run video action recognition with compressed frames on the cloud, while maintaining high recognition accuracies. Thirdly, TSD can be plugged in as a preprocessing module of any existing 3D CNNs. Extensive experiments show the effectiveness and characteristics of TSD.

*Corresponding author is Zhanghui Kuang.

CCS CONCEPTS

• **Information systems** → *Multimedia information systems*;

KEYWORDS

Video Action Recognition; Temporal Sequence Distillation

1 INTRODUCTION

Action recognition in videos has large progresses due to the developments of deep Convolution Neural Networks (CNNs) [8, 11, 22] and the presences of large-scale video datasets such as Kinetics [1] and UCF101 [24]. Existing work can be generally divided into two categories, 2D CNNs [21, 32] and 3D CNNs [1, 27].

In the first category, 2D CNNs are utilized to extract feature for each frame separately and then pool their predictions across the whole video. For capturing temporal information, 2D CNNs could be adjusted by aggregating features over time [32], or fusing multiple modalities like RGB difference and optical flow [21, 32]. Although 2D CNNs can be trained by sparsely sampling frames to reduce storages (*e.g.*, GPU memory) and computations, 2D CNNs have limitations in at least two aspects. First, 2D CNNs (*e.g.*, TSN [32], which is current state-of-art method) are prone to treat video as an unstructured bag of frames, ignoring the sequential information of video frames. Moreover, for 2D CNNs, the ability to capture long-range dependencies, which is crucial for action recognition performance, is restricted due to the independent frame feature extraction.

In the second category, the above 2D models were extended to 3D models [27]. For instance, I3D [1] introduced the operations of 3D convolutions to capture long-term temporal information, which is achieved by the expansive receptive field from dense stack of temporal convolution. I3D achieved state-of-the-art performances by inflating 2D convolutional filters of InceptionNet [11] into 3D. It was trained using 64-frame video clips and tested using the entire video (*e.g.*, 64 frames on UCF101). Specifically, in training, it randomly samples 64 frames with both spatially and temporal clipping.

Table 1: Comparison of methods in VA SaaS scenarios. #clips and #params indicate the clip number and the parameter number of the model during evaluation. State-of-art methods I3D, TSN, and Res18 ARTNet could only be deployed on cloud, so we report their time of sampling and cropping clips as client runtime. For I3D+TSD, the client runtime consists of both sampling/cropping time and extra computations of TSD. We can see that the I3D+TSD model achieves an accuracy of 72.4, with 238G total FLOPS and only 120 frames required to be transmitted. When efficiency is given precedence, I3D+TSD achieves an accuracy of 70.7, which is comparable to the I3D baseline. But it needs only 60 frames to be transmitted rather than 250 frames.

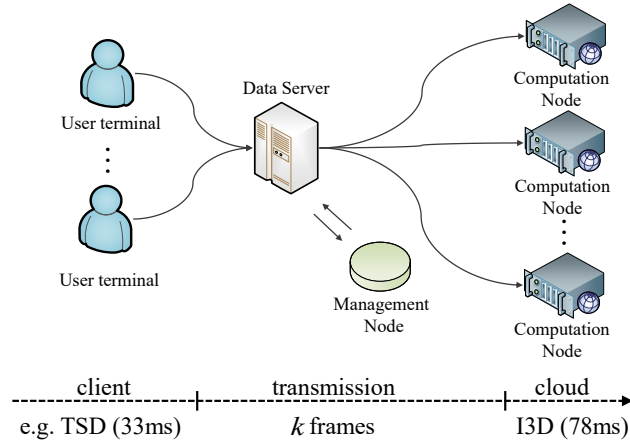| | client | | | | | transmission | server | | | | |
	#clips	#params (M)	FLOPS (G)	runtime (ms)	processed #frames	#frames	backbone	#params (M)	FLOPS (G)	runtime (ms)	accuracy
I3D [1]	1	-	-	1	250 (1×250)	250 (1×250)	Inceptionv1	12	422	327	71.1
TSN [32]	25	-	-	1	250 (25×10)	250 (25×10)	Inceptionv2	10	508	399	69.1
Res18 ARTNet [31]	25	-	-	4	4000 (25×160)	4000 (25×160)	Resnet18	33	4850	3691	69.2
I3D+TSD	3	4	36	33	240 (3×80)	**60** (3×20)	Inceptionv1	12	**101**	**78**	70.7
I3D+TSD	3	4	36	33	240 (3×80)	**120** (3×40)	Inceptionv1	12	**202**	**154**	**72.4**
I3D+TSD	10	4	120	109	800 (10×80)	200 (10×20)	Inceptionv1	12	338	258	72.1
I3D+TSD	10	4	120	109	800 (10×80)	400 (10×40)	Inceptionv1	12	676	511	**73.2**

Figure 1: Architecture of VA SaaS. It consists of the client-part and the cloud-part. Video frames are transmitted between them. The proposed TSD can be deployed on user terminals to learn compact frame representations. It reduces frame transmission and computations on the cloud and maintains high recognition accuracy. For example, TSD+I3D spends 33ms and 78ms on the client and the cloud respectively, while maintaining a high accuracy of 70.7 on Kinetics when only transmits 60 frames between the client and the cloud. It significantly surpasses the conventional I3D, which transmits 250 frames and takes 327ms on the cloud.

In test, the network fetches the center clip of the whole video as inputs, averaging predictions temporally.

Although the above approaches achieved good performances for action recognition, their requirements of dense-frame predictions impede their applications in practice, such as Video Analytics Software as a Service (VA SaaS), which has increasing demands in the recent years. Fig. 1 illustrates the architecture of one typical VA SaaS, which has two parts, including the client-part and the cloud-part. The client-part has multiple terminals such as mobile phones,

which have limited memories and computational resources. The cloud-part contains data server and computation nodes, where the data server distributes data (*e.g.*, video frames) to different computation nodes, while each computation node performs video analysis (*e.g.*, action recognition) and returns results (*e.g.*, predicted action labels) to the data server. The transmission bandwidths between the client and the cloud, storage resources, and computation resources are typically constrained, in the demands of reducing their cost. For example, a recognition system in the cloud is often permitted to access just a small number of frames but is required to produce results as good as possible.

As previous work of action recognition are typically resource consuming, they are not applicable in VA SaaS as summarized in Table 1, which compares different methods in terms of model storages and computations in client and cloud, numbers of frames in transmission and processing, as well as recognition accuracy. The conventional methods [1, 31, 32] employ dense-random sampling to select video frames, forming frame clips. When deploying them in VA SaaS, although each of them has their own merits, we observe that all of the previous methods are time and transmission bandwidth consuming.

To resolve the above issues, this work presents a new problem of **few-frame action recognition**, which aims at maintaining high accuracy, but accessing only a few frames during training and test. To this end, we propose Temporal Sequence Distillation (TSD), which distills a long video sequence into a short one consisting of compact and informative frames, while preserving temporal and spatial discriminativeness as much as possible.

Fig. 2 illustrates the frames of the distilled sequence of TSD compared with uniformly sampled frames from the original video sequences. The output frames of TSD encode actions with "ghosting". In this way, they are compact and informative by reducing the temporal redundancy of videos while distilling action-relevant information.

We design TSD as a lightweight differentiable CNN. It is a general component which can be plugged into any 3D CNNs based action recognition framework. Since I3D [1] achieves very impressive results for action recognition, we choose it as our start point to demonstrate the effectiveness of the proposed TSD.
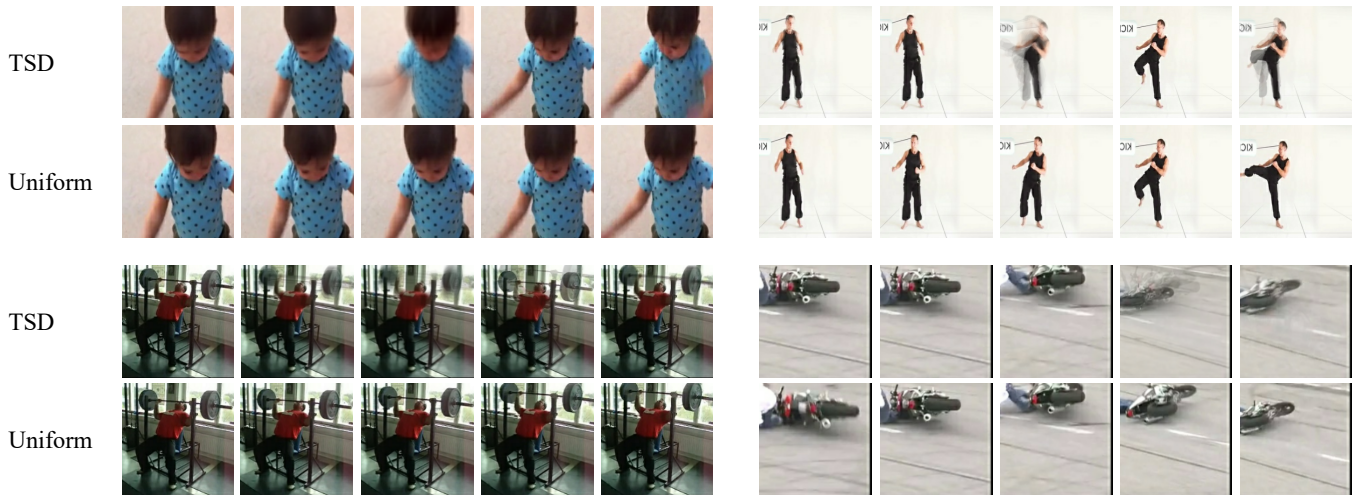
Figure 2: Examples of generated frames of TSD and uniform sampling. The first and the third rows show generated frames of TSD while the second and the fourth show the frames uniformly sampled from original RGB sequences.

We extensively evaluate TSD enhanced I3D (I3D+TSD) with few frames as input and compare with its counterpart I3D. Fig. 3 shows that the performance of I3D degrades greatly when the input frame number decreases from 40, to 20, 10 and 5 on the standard test set of Kinetics and UCF101 split 1. There exists a plenty of redundancy between video frames, and some frames that irrelevant to actions themselves, as shown in Fig. 2. We believe the low performance of I3D on few frame action recognition results from the missing of some key frames that most related to actions. As shown in Fig. 3, the action recognition performance degradation has been alleviated considerably when feeding I3D with distilled frames from TSD.

This work has three main **contributions**.

- We conduct a thorough investigation on the trade-off between the number of frames processed and the accuracy of video action recognition with 3D CNNs, and define the problem of few-frame action recognition.
- We explore various approaches of few-frame action recognition. Naive sampling approach (*i.e.*, random sampling and uniform sampling), and supervised approach (*i.e.*, attention based method, TSD) have been implemented and fairly compared.
- We propose Temporal Sequence Distillation (TSD), a novel technique that can compress long video clips into compact short ones. Together with traditional 3D CNNs, it can be jointly trained end-to-end. TSD+I3D typically requires only 50% of the number of frames compared with its counterpart I3D, to achieve almost the same action recognition accuracies, demonstrating that TSD is very applicable in VA SaaS scenarios.

2 RELATED WORK

In this section, we introduce associated work with our proposed Temporal Sequence Distillation framework. We categorize the related research into four groups: action recognition, temporal-spatial investigation, video summarization and video compression.
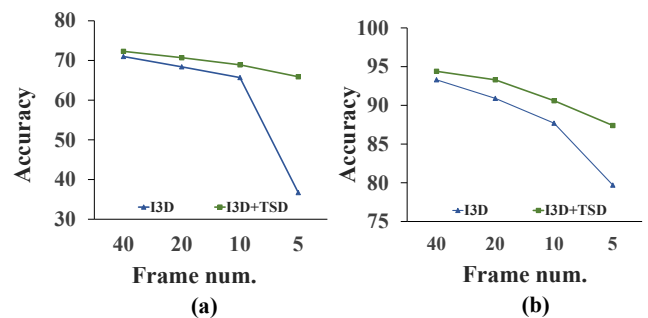

Figure 3: Effect of processed frame numbers of I3D and I3D+TSD during testing. (a) and (b) show performance on Kinetics and UCF101 split 1 respectively. More details can be found in Section 4.

Action recognition. Human action recognition developed rapidly in recent years. Especially boosted with several competitive challenges providing public large-scale video data, deep learning based methods have much overwhelmed previous hand-crafted feature based methods [9, 13, 15, 33]. Early work of action recognition focuses on designing local hand-crafted features generalized for video, such as Space-Time Interest Points (STIP) [16], Histogram of 3D Gradient (HOG3D) [18], Dense Trajectory (DT) [29], improved Dense Trajectory (iDT) [30], Motion Boundary Histograms (MBH) [2] and Optical Flow Guided Feature (OFF) [26]. In order to describe frame-level or clip-level video content with a fixed length feature, feature encoding is utilized to aggregate local features. Typical solutions include histogram, bag of words, fisher vector, and vector of locally aggregated descriptors (VLAD) [12]. These multi-step approaches are highly complicated, and errors occurred in the earlier steps are easily accumulated in the later steps. Therefore, their final performance heavily depends on sophisticated design of each step. Two-stream CNN solution brought a significant breakthrough

in action recognition [21], because deep features pre-trained on ImageNet own better semantic description ability compared with local features. In addition, this deep learning based approach unifies feature extraction, encoding, and classification in a deep neural network trained in an end-to-end fashion. A temporal segmented two-stream network (TSN) can utilize several video segments simultaneously for action classification during training, which achieved the state of the art performance on several databases [32]. Encoding frame-level deep features for a video sequence presentation is useful for the two-stream approach. And skip-links between the RGB and optical flow streams can also increase performance [3, 4, 17].

Temporal-spatial investigation. Though the classic two-stream solution achieved good performance on action recognition, it relies heavily on optical flow calculation, which is a very time-consuming process. There are some fast-implemented temporal-change estimations for video, which are good replacements of optical flow. Zhu *et al.* proposed MotionNet which takes consecutive video frames as input and estimates motion implicitly accompanying the RGB stream [38]. In TSN work, RGB-difference has been investigated to replace optical flow as input for the optical flow stream [32], and RGB-difference can obtain a comparable performance with optical flow. The fast implementation of motion estimation can speed up the two-stream approach, however, two separate neural networks are required by the two streams. Since both appearance and motion information are contained in a video sequence, it is a natural choice to utilize only one 3D CNN (C3D) network to explore spatial-temporal information. The first proposal of C3D for action recognition [27] did not perform better than the two-stream solution. With an inflated initialization of C3D (I3D) from 2D CNN (C2D) model pre-trained using ImageNet data, I3D achieved the state-of-the-art performance on UCF101 and Kinetics [1]. Even using only one RGB stream, I3D can outperform the two-stream approach on kinetics. Due to temporal convolutional operations between frames, I3D is slower than a counterpart C2D with the same depth. Pseudo-3D (P3D) and spatiotemporal-separable 3D (S3D) simulate $3 \times 3 \times 3$ convolutions with $1 \times 3 \times 3$ convolutional filters on spatial domain and $3 \times 1 \times 1$ convolutions to construct temporal connections, thus the computation load of I3D can be reduced by 1.5×. Notice that, typical modalities as optical flow and RGB-difference can model the temporal connections of frames but at the expense of even more redundant computations, which is different from the proposed TSD.

Video summarization. Potapov *et al.* [19] utilized Support Vector Machines (SVMs) to predict one importance score for every video shot. After that, video shots with the highest scores were assembled as a sequence to generate a summary. Gong *et al.* [6] applied sequential DDP to select diverse subsets, and feature embedding was learned with the guide of manual summary labels. In the work above, hand-crafted features were explored for representing videos shots. Recently, many deep neural networks based video summarization approaches [5, 14, 36, 37] were proposed, which achieved the state of the art performance on public databases [7, 23]. The proposed TSD can be considered as one kind of summarization of videos. However, video summarization approaches focus on extracting frames with content highlight and visual aesthetics. Instead, the proposed TSD targets at maximizing action recognition

accuracy with little transmission bandwidth and computation cost by distilling compact frames.

Video compression. High-Efficiency Video Coding (HEVC) is the newest video compression standard [25]. Its efficiency mainly relies on accurate frame prediction, including intra-prediction and inter-prediction, which explore spatial redundancy within a single frame, and temporal redundancy between inter-frame respectively. Videos are encoded with only key frames, motion vectors describing block-wise motion between consecutive frames, and residual errors. Although video compression can reduce transmission bandwidth in VA SaaS applications, different from the proposed TSD, it cannot reduce the computation cost on cloud since the number of frames fed into 3D CNNs (*e.g.*, I3D) keeps unchanged. Moreover, our proposed TSD is orthogonal to video compression approaches, and can be combined with them for further transmission bandwidth reduction.

3 THE PROPOSED APPROACH

In this section, we first define the problem of few-frame action recognition, then introduce the proposed TSD in detail, and finally discuss its relations to other approaches.

3.1 Few-Frame Action Recognition

Few-frame action recognition recognizes action based on only a few frames of videos. It not only reduces the input data size which can save transmission bandwidth in VA SaaS scenarios, but also speeds up the recognition procedure. Traditional action recognition acceleration approaches explore reducing computation complexity in the dimension of spatial size [10, 31, 35], or network architecture [20, 34]. Few-frame action recognition attempts to accelerate inference from another dimension, *i.e.*, the number of frames fed into networks.

3.2 Temporal Sequence Distillation

Towards few-frame action recognition problem, we present *Temporal Sequence Distillation* to transform a long video sequence of length T into a short one of length T_s ($T_s < T$) automatically. In order to achieve compactness and informativeness, each distilled frame should be generated according to the global context of the long video sequence. For simplicity, we assume that each distilled frame is a linear combination of all frames of the long sequence. Formally,

$$Y = XP, \tag{1}$$

where X and Y indicate the frame matrix (with each column as one frame) of the input long sequence and the output short sequence respectively. P is a transformation matrix of shape $T \times T_s$. Its element P_{ij} indicates the importance of the i_{th} frame of the input long video for the j_{th} distilled frame. Ideally, P_{ij} is nonzero if the i_{th} frame of the input video is relevant to the semantic action category encoded by the input video otherwise 0. Herein, we propose a temporal sequence distillation block to predict the transformation matrix P based on input videos.

3.2.1 Temporal Sequence Distillation Block. Fig. 4 shows the architecture of TSD block. It inputs feature maps of all frames of
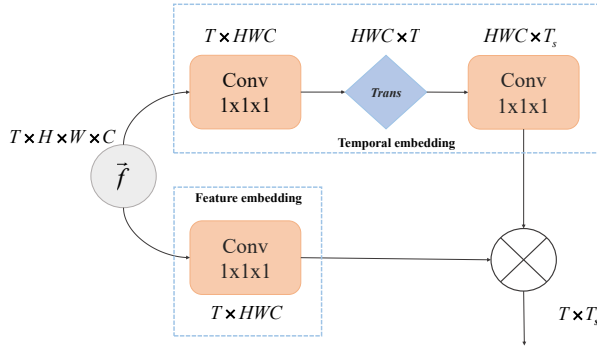
Figure 4: Illustration of TSD block. $\vec{f}$ **indicates the input feature of TSD block.** T, C, H **and** W **indicate the input frame number, the channel number, the feature map height and the feature map width respectively.** T_s **indicates the output frame number of TSD block. The** $Trans$ **unit indicates transpose operation, while** $\otimes$ **represents matrix multiplication.**

the input video sequence, and outputs a transformation matrix. Given the input feature maps $\vec{f}$ with size $T \times H \times W \times C$, it produces the transformation matrix P with one dual-path network inspired by the self-attention [28] on sentence embedding extraction. The first path is designed to extract temporal embedding. It transforms the input feature maps to one feature map O with size $H \times W \times C \times T_s$ via one transpose operation and two convolution operations. The second is designed to extract feature embedding. It transforms the input to one feature map G with size $T \times H \times W \times C$ via one convolution operation. Formally, we have

$$O = \text{Trans}(\vec{f} * w_\alpha) * w_\beta, \tag{2}$$

and

$$G = \vec{f} * w_\gamma, \tag{3}$$

where $*$ indicates convolution operation, and Trans() indicates transpose from a tensor with size $T \times H \times W \times C$ to one with size $H \times W \times C \times T$. w_α, w_β and w_γ are 3D convolution kernels.

The transformation matrix P is given by

$$P = \text{softmax}(O^{'}G^{'}), \tag{4}$$

where $O^{'}$ and $G^{'}$ are two matrices of size $HWC \times T_s$ and $T \times HWC$, which are reshaped from O and G respectively. softmax() column-wisely normalizes an input matrix by softmax.

3.2.2 Temporal Sequence Distillation Network. The proposed Temporal Sequence Distillation network (TSD network) consists of three components. Namely, the coarse feature extractor, the TSD block, and the main network. Fig. 5 visualizes its pipeline. The feature extractor is designed to extract coarse feature of the video snippets X. The successive TSD block takes the coarse feature as input, and predicts the corresponding transformation matrix P. The distilled short sequence Y is calculated as Equation 1. The main network is a CNN with temporal convolution to recognize video action. In this paper, we take I3D (Inception v1 with BN) as the main network. The pipeline is entirely end-to-end, and could be trained and evaluated jointly. During evaluation, the TSD network could

be separated into two components: coarse feature extractor+TSD block, named client component, and main network, named cloud component. The two components are mutually parameter independent, thus they can be deployed onto terminals and cloud separably. In this way, only the distilled frames are transmitted to cloud to carry out the video recognition task. This can accelerate the data transmission and relief the bandwidth pressure.

3.2.3 Coarse Feature Extractor. We design the coarse feature extractor for TSD network with keeping efficiency in mind. We choose the current popular light-weight CNN MobileNet [10] as the feature extractor. For acceleration, we downsample the input of the coarse feature extractor from 224×224 to 112×112.

3.3 Relations to Other Approaches

Sampling a subset of input frames is one natural way to distill long video sequences into short ones. There exist two naive sampling strategies, namely, random sampling and uniform sampling which are widely used in other methods such as TSN [32].

Random Sampling. A subset of input frames are randomly sampled. We denote I3D with the random sampling when training and test as I3D$_R$.

Uniform Sampling. A subset of input frames with equal intervals are uniformly sampled. We denote I3D with uniform sampling when training and test as I3D$_E$.

Temporal Attention. During training, we learn a scalar importance weight for each input frame. All input frames scaled with their corresponding importance weight are fed into the main action recognition network (I3D) for end-to-end training. During testing, we select frames with the top T_s predicted importance, which are fed into the main network for action recognition. We denote I3D with temporal attention as I3D$_A$.

All the above methods follow the same pipeline of frame selection as defined in Equation 1. The differences are how to generate the transformation matrix P. For random and uniform sampling, it randomly or uniformly activates one value at each column. For temporal attention, it learns one $l1$ normalized importance weight for each column.

4 EXPERIMENTS

We conduct comprehensive ablation experiments along with a thorough comparison between the proposed TSD and the state-of-the-art action recognition methods.

4.1 Datasets and Evaluation

We evaluate the proposed TSD on two popular datasets, namely, UCF101 and Kinetics. Both UCF101 and Kinetics contain short trimmed videos, each of which is annotated with one action label. UCF101 consists of 13320 videos from 101 classes and has three splits in the testing set. We report the average performance of 3 splits except as otherwise noted. Kinetics contains 240000 training videos from 400 human action classes with 400 or more samples for each class, and 40000 testing videos with 100 for each class. All the videos are trimmed and last around 10s.

During testing, we randomly sample Q times and report the average of probabilities as final prediction. We fairly compare the proposed TSD with its counterparts with the same sampling times.
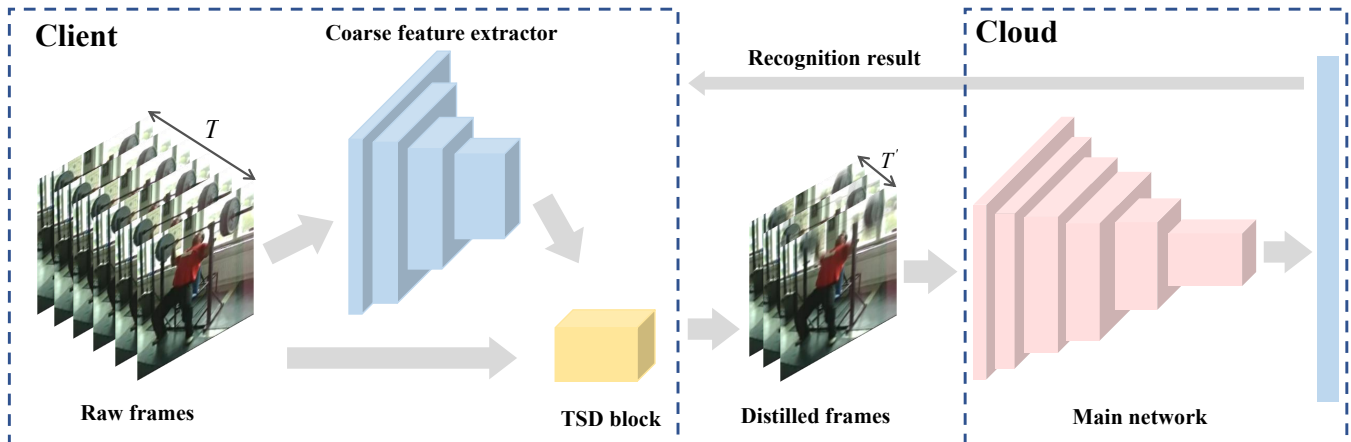
Figure 5: Pipeline of TSD network in VA SaaS scenarios. Only the distilled frames are transmitted.

We also conduct experiments with different Q (*i.e.*, 1, 3, and 10) to study the effect of the sampling times of the proposed TSD.

The evaluation procedures of few frame action recognition are as follows:

- **I3D**. It is trained as done in [1] with randomly sampled consecutive 64-frame clips. During testing, we sample consecutive T_s frames to evaluate and then average predictions temporally as done in [1]. We extensively evaluate its performance on few frame action recognition by setting T_s to 5, 10, 20, and 40.
- **I3D$_R$**. We randomly sample T-frame clips, from each of which T_s ($T \geq T_s$) *ordered* frames are randomly sampled and fed into $I3D$ during both training and testing.
- **I3D$_E$**. We randomly sample T-frame clips, and then sample T_s frames with equal intervals as input from each clip during both training and testing.
- **I3D$_A$**. We randomly sample T-frame clips, which are fed into the I3D network with temporal attention during training. T_s frames with the top T_s importance scores are selected from each clip for action recognition during testing.
- **I3D+TSD**. We randomly sample T-frame clips, which are fed into the TSD network, and then take the outputted distilled clip of length T_s as input to the I3D network.

4.2 Implementation Details

All of our experiments are done using Tensorflow on one cloud with 8 Titan X GPU servers.

Training I3D. We train the I3D model by following the practice in [1].

Training TSD on Kinetics. We initialize the coarse feature extractor with the MobileNet model pretrained on ImageNet, and its sequence distillation module using Gaussian with standard variance 0.01. The main branch is initialized with the pre-trained model on Kinetics. We train TSD in two stages on Kinetics. First, we train the coarse feature extractor and the TSD block while fixing the main branch. We employ multi-step learning rate policy with the base learning rate being 0.1. The learning rate decays by 10 times

every 10000 iterations, and the maximum iteration size is set to 50000. Second, we train the whole TSN including the coarse feature extractor, the TSD block and the main branch end-to-end. We set the learning rate to 0.01 with the decay factor of 0.3 and the decay step of 30000.

Training TSD on UCF101. All the parameters of the TSD network other than the final FC layer are first initialized with the pretrained weights on Kinetics, and then the whole TSD network is trained on UCF101 end-to-end. Note that the learning rate of the final FC layer is set to 10 times of those of other layers.

4.3 Effectiveness of TSD

4.3.1 Comparison of Sampling Methods. We compare the proposed I3D+TSD with its baseline I3D and other sampling approaches in Table 2. It has been shown that all sampling methods outperform the baseline I3D. We believe that this is because of these sampling methods keeping the same frame selection procedure during both training and test. Among all sampling methods, the proposed I3D+TSD achieves the biggest performance improvement. Specifically, I3D+TSD improves the accuracy of I3D on Kinetics by 29.2, 2.9, 2.4, and 1.4 when the sampled frame number T_s is set to 5,10,20, and 40 respectively.

4.3.2 Effect of Input Clip Number (Q). Table 3 investigates the effect of the input clip number. For each clip, we generate a short sequence of frames with TSD which are fed into I3D, achieving category probabilities. The averaged probabilities over all clips are considered as the final prediction. It has been shown that I3D+TSD always outperforms its counterpart I3D regardless of the sampling clip number. Increasing the input clip number can improve action recognition accuracies for both I3D+TSD and I3D.

4.3.3 Experiments on Other Dataset. We further validate the effectiveness of TSD on the classic UCF101 dataset in Table 4. Similarly, I3D+TSD achieves performance improvements as on Kinetics. Specifically, it outperforms I3D by 7.7, 2.9, 1.8, and 0.9 when the sampled frame number T_s is set to 5, 10, 20, and 40 respectively.

Table 2: Comparison with baseline and other frame selection approaches on Kinetics. All are tested on RGB modality with Inception v1 as backbone and with 3-clip evaluation (Q = 3) for fair comparison. Δ indicates the performance improvement compared with the baseline I3D.

#sampled frames (T_s)	Clip length (T)	I3D	I3D$_A$/Δ	I3D$_R$/Δ	I3D$_E$/Δ	I3D+TSD/Δ
5	5	36.7	34.6 /-2.1	-	-	-
	10	-	35.6/-1.1	37.1/+0.4	39.9/+3.2	65.7/+29.0
	20	-	36.1/-0.6	40.9/+4.2	41.7/+5.0	**65.9/+29.2**
10	10	65.7	63.0/-2.7	-	-	-
	20	-	64.1/-1.6	65.8/+0.1	66.1/+0.4	67.9/+2.2
	40	-	65.2/-0.5	65.9/+0.2	66.5/+0.8	**68.6/+2.9**
20	20	68.4	68.6/+0.2	-	-	-
	40	-	68.8/+0.2	69.0/+0.6	69.2/+0.8	69.9/+1.5
	80	-	68.8/+0.2	68.6/+0.2	69.9/+1.5	**70.7/+2.3**
40	40	71.0	71.1/+0.1	-	-	-
	80	-	71.4/+0.4	70.8/-0.2	71.4/+0.3	**72.4/+1.4**

Table 3: Effect of the clip number (Q).

#Clips (Q)	Method	Clip length (T)	#sampled frames (T_s)	Accuracy
1	I3D	5	5	34.7
	I3D+TSD	20	5	58.0
	I3D	10	10	57.9
	I3D+TSD	40	10	62.2
	I3D	20	20	61.9
	I3D+TSD	80	20	64.6
	I3D	40	40	64.7
	I3D+TSD	80	40	**65.9**
3	I3D	5	5	36.7
	I3D+TSD	20	5	65.9
	I3D	10	10	65.7
	I3D+TSD	40	10	68.6
	I3D	20	20	68.4
	I3D+TSD	80	20	70.7
	I3D	40	40	71.0
	I3D+TSD	80	40	**72.4**
10	I3D	5	5	37.4
	I3D+TSD	20	5	67.8
	I3D	10	10	67.9
	I3D+TSD	40	10	70.2
	I3D	20	20	70.6
	I3D+TSD	80	20	72.1
	I3D	40	40	71.8
	I3D+TSD	80	40	**73.2**

Table 4: Comparison between I3D and I3D+TSD on UCF101 split1. All are 1-clip evaluated on RGB modality with Inception v1 with BN as backbone for fair comparison.

#sampled frames (T_s)	Method	Clip length (T)	Accuracy
5	I3D	5	79.7
	I3D+TSD	10	84.9
	I3D+TSD	20	**87.4**
10	I3D	10	87.7
	I3D+TSD	20	89.4
	I3D+TSD	40	**90.6**
20	I3D	20	90.9
	I3D + TSD	20	92.3
	I3D + TSD	20	**92.7**
40	I3D	40	93.3
	I3D+TSD	80	**94.2**

I3D+TSD performs much better than I3D on few frame action recognition. Therefore, I3D+TSD is more applicable than I3D in VA SaaS scenarios.

4.4 Efficiency

TSD can be considered as one way of network acceleration. We compare parameter numbers and FLOPS of the proposed TSD+I3D and its counterpart I3D with comparable accuracy in Table 5. All the speed is measured on servers with 8 Titan X GPUs. Although I3D+TSD has more parameters, however, it has about 32% $\sim$ 33% less FLOPS, and is about 13% $\sim$ 26% faster than I3D. When I3D+TSD

is deployed in VA SaaS scenarios, it has even about 50% $\sim$ 51% less FLOPS, and is about 45% $\sim$ 48% faster than I3D on cloud.

5 CONCLUSIONS

We have proposed a novel Temporal Sequence Distillation technique for few frame action recognition by distilling long video sequences into short compact sequences. It consists of coarse feature extractor, temporal sequence distillation block, and main network (*e.g.*, I3D), all of which can be trained in an end to end way. Because it distills key frames relevant to actions and discards considerable temporal redundancy, it can accelerate action recognition while preserving recognition accuracy. When deploying its coarse feature extractor and TSD block on client terminals, and its main network on cloud in VA SaaS scenarios, it can greatly reduce the data transmission bandwidth between client and cloud.

Table 5: Efficiency comparison. All are tested on Kinetics with RGB modality and 3 clips ($Q = 3$). The I3D$_{(T)}$ represents I3D with length-T clips. The I3D+TSD$_{(T \to T_s)}$ represents the I3D fed with T_s distilled frames from length-T clips in original videos. The I3D+TSD$_{(T \to T_s)}^{cloud}$ indicates I3D+TSD is deployed in VA SaaS scenarios (i.e., only the main network is deployed on cloud while others on client terminals). Δ indicates FLOPS or speed improvement compared with baselines.

Method	Accuracy	#params (M)	FLOPS /Δ (G)	Speed/Δ (ms)
I3D$_{(10)}$	65.7	12.1	51	46.1
I3D+TSD$_{(20 \to 5)}$	65.9	16.4	34/-17	**39.9/-6.2**
I3D+TSD$_{(20 \to 5)}^{cloud}$	65.9	12.1	25/-26	**23.9/-22.2**
I3D$_{(20)}$	68.4	12.1	101	77.9
I3D+TSD$_{(40 \to 10)}$	68.6	16.4	69/-32	**58.2/-19.7**
I3D+TSD$_{(40 \to 10)}^{cloud}$	68.6	12.1	51/-50	**43.1/-34.8**
I3D$_{(40)}$	71.0	12.1	202	150.3
I3D+TSD$_{(80 \to 20)}$	70.7	16.4	137/-65	**111.0/-39.3**
I3D+TSD$_{(80 \to 20)}^{cloud}$	70.7	12.1	101/-101	**77.9/-72.4**

REFERENCES

[1] Joao Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 4724–4733.

[2] Navneet Dalal, Bill Triggs, and Cordelia Schmid. 2006. Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*. Springer, 428–441.

[3] Ali Diba, Vivek Sharma, and Luc Van Gool. 2017. Deep temporal linear encoding networks. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

[4] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. 2016. Spatiotemporal residual networks for video action recognition. In *Advances in Neural Information Processing Systems*. 3468–3476.

[5] Litong Feng, Ziyin Li, Zhanghui Kuang, and Wei Zhang. 2018. Extractive Video Summarizer with Memory Augmented NeuralNetworks. In *ACM Multimedia*.

[6] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. 2014. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*. 2069–2077.

[7] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. 2014. Creating summaries from user videos. In *European Conference on Computer Vision*. Springer, 505–520.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[9] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. 2015. Activitynet: A large-scale video benchmark for human activity understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 961–970.

[10] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[11] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[12] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. 2010. Aggregating local descriptors into a compact image representation. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 3304–3311.

[13] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition*. 1725–1732.

[14] Dotan Kaufman, Gil Levi, Tal Hassner, and Lior Wolf. 2017. Temporal tessellation: A unified approach for video analysis. In *The IEEE International Conference on Computer Vision*, Vol. 8.

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

[16] Ivan Laptev. 2005. On space-time interest points. *International Journal of Computer Vision* 64, 2-3 (2005), 107–123.

[17] Antoine Miech, Ivan Laptev, and Josef Sivic. 2017. Learnable pooling with Context Gating for video classification. *arXiv preprint arXiv:1706.06905* (2017).

[18] Omar Oreifej and Zicheng Liu. 2013. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 716–723.

[19] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. 2014. Category-specific video summarization. In *European Conference on Computer Vision*. Springer, 540–555.

[20] Zhaofan Qiu, Ting Yao, and Tao Mei. 2017. Learning spatio-temporal representation with pseudo-3d residual networks. In *2017 IEEE International Conference on Computer Vision*. IEEE, 5534–5542.

[21] Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*. 568–576.

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

[23] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. 2015. Tvsum: Summarizing web videos using titles. In *The IEEE Conference on Computer Vision and Pattern Recognition*. 5179–5187.

[24] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).

[25] Gary J Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand. 2012. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology* 22, 12 (2012), 1649–1668.

[26] Shuyang Sun, Zhanghui Kuang, Lu Sheng, Wanli Ouyang, and Wei Zhang. 2018. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

[27] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 4489–4497.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 6000–6010.

[29] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. 2013. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision* 103, 1 (2013), 60–79.

[30] Heng Wang and Cordelia Schmid. 2013. Action recognition with improved trajectories. In *2013 IEEE International Conference on Computer Vision*. IEEE, 3551–3558.

[31] Limin Wang, Wei Li, Wen Li, and Luc Van Gool. 2018. Appearance-and-Relation Networks for Video Classification. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

[32] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*. Springer, 20–36.

[33] Zuxuan Wu, Y.-G. Jiang, Larry S Davis, and Shih-Fu Chang. 2017. LSVC2017: Large-Scale Video Classification Challenge. http://bigvid.fudan.edu.cn/LSVC_MM17/. (2017).

[34] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. 2017. Rethinking Spatiotemporal Feature Learning For Video Understanding. *arXiv preprint arXiv:1712.04851* (2017).

[35] Xiaoyu Yue, Zhanghui Kuang, Zhaoyang Zhang, Zhenfang Chen, Pan He, Yu Qiao, and Wei Zhang. 2018. Boosting up Scene Text Detectors with Guided CNN. *arXiv preprint arXiv:1805.04132* (2018).

[36] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Summary transfer: Exemplar-based subset selection for video summarization. In *The IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1059–1067.

[37] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Video summarization with long short-term memory. In *European Conference on Computer Vision*. Springer, 766–782.

[38] Yi Zhu, Zhenzhong Lan, Shawn Newsam, and Alexander G Hauptmann. 2017. Hidden two-stream convolutional networks for action recognition. *arXiv preprint arXiv:1704.00389* (2017).