TEMPORAL AGGREGATION APPROACHES FOR FEW-SHOT HUMAN ACTION RECOGNITION IN VIDEOS

TEMPORAL AGGREGATION APPROACHES FOR FEW-SHOT HUMAN ACTION RECOGNITION IN VIDEOS

BY

YANG BO, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of philosophy (2021)	McMaster University
(Department of Computing and Software)	Hamilton, Ontario, Canada

TITLE:	Temporal Aggregation Approaches for Few-shot Human Action Recognition in Videos
AUTHOR:	Yang Bo M.Sc. (Computer Science), McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Wenbo He
NUMBER OF PAGES:	xx, 136

Lay Abstract

The core challenge of human action recognition in video is generating the descriptor which preserves the spatial (the content in a single frame) and temporal (the correlation between the frames) information of the video. Various actions need a different number of frames to represent. For example, it is hard to distinguish action walking and running given a single frame. Current approaches adopt Deep Neural Networks (DNNs) to study the video descriptor. However, they fail to preserve the temporal information of the entire video and require a large number of training videos, hence cannot be directly used in the few-shot scenario. In this thesis, we propose three video descriptor generation approaches that preserve the temporal information of the entire video and only introduce a few training parameters. We show that our approaches achieve comparable or better performance compared to the state-of-the-art approaches for both regular and few-shot human action recognition tasks.

Abstract

Over the past decade, the research of deep learning has dramatically progressed and achieved overwhelming performance for various tasks. This success is highly dependent on a large number of manually labeled data. However, it is not always possible to collect enough training data, meanwhile manually labelling a large amount of data is labour-intensive. To learn from a limited number of examples with supervised information, a new machine learning paradigm called Few-Shot Learning (FSL) is introduced. For the few-shot human action recognition task, the core challenge is preserving both spatial and temporal information of the video with few labeled videos. Many approaches based on Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) have been proposed to address the human action recognition task. However, these works fail to preserve the temporal information of the entire video and require a large number of training videos, hence directly applying them to solve the few-shot human action recognition would severely overfit. Currently, there are only a few approaches to address the few-shot human action recognition problem. They either focus on learning how to compare the similarity between video descriptors with few training samples or solving the training samples deficiency by data augmentation. In this thesis, we propose three approaches that preserve the temporal information of the entire video given the frame/segment features:

the Discriminative Video Descriptor (DVD), Temporal Attention Vector (TAV) and Contents and Length based Temporal Attention (CLTA). These methods preserve the temporal information of the entire video by convolving frame features with the basis for small dimensional space recursively, aggregating the frame/segment features with temporal weights which are manually defined, aggregating the frame features with temporal weights which is provided by learned Gaussian distribution functions based on both length and content of the video, respectively. We evaluated our approaches on different datasets in various scenarios (regular or few-shot), our approaches achieve comparable or better results compared to the state-of-the-art approaches on different datasets.

Acknowledgements

First and foremost, I would like to express my great appreciation to my supervisor, Dr.Wenbo He, for allowing me to work with her as a Ph.D. student. I was encouraged and motivated by her great enthusiasm, insight and criticism for academic research.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Rong Zheng, Dr. Sanzheng Qiao and Dr. Alan Wassyng, for their insightful comments and encouragement, but also for the hard questions which are incented me to widen my research from various perspectives.

Besides that, my sincere thanks also go to my fellow groupmates Yangdi Lu, Shusheng Li for the stimulating discussions, the sleepless nights we were working together before deadlines, and all the fun we have had in the last four years.

Last but not least, I would like to thank my family for their understanding and support during my life.

Contents

List of Figures

List of Tables

Notation, Definitions, and Abbreviations

Abbreviations

Adascan	Adaptive Scan Pooling
ARC	Attentive Recurrent Comparators
ARN	Action Relation Network
ARTNet	Appearance-and-Relation Network
BoW	Bag of Words
C3D	Convolutional 3D
CLTA	Contents and Length based Temporal Attention
CMN	Compound Memory Network
CNNs	Convolutional Neural Networks
DNNs	Deep Neural Networks

DOVF	Deep Local Video Feature
DVD	Discriminative Video Descriptor
ECO	Efficient Convolutional Network for Online Video
FSL	Few-shot Learning
FstCN	Factorized Spatiotemporal Convolutional Networks
GANs	Generative adversarial networks
GNNs	Graph Neural Networks
GRU-RCN	Gated Recurrent Unit-Recurrent Convolutional Network
IDT	Improved Dense Trajectories
KNN	K-nearest Neighbours
LRCN	Long-term Recurrent Convolutional Network
LSTM	Long Short-term Memory
LTC	Long-term Temporal Convolutions
MAML	Model-Agnostic Meta-Learning
MANN	Memory-Augmented Neural Networks
MIFS	Multi-skIp Feature Stacking
MLP	Multilayer Perceptron
ProtoNet	Prototypical Networks

R3D	3D Residual Network
RNNs	Recurrent Neural Networks
ResNet	Residual Network
SIFT	Scale-invariant Feature Transform
SMN	Spatiotemporal Multiplier Network
3D SoSN	3D Second-order Similarity Network
ST-ResNet	Spatiotemporal Residual Network
STIP	Space Time Interest Point
STPN	Spatiotemporal Pyramid Network
SURF	Speeded Up Robust Feature
SVM	Support Vector Machine
SVR	Support Vector Regression
TAM	Temporal Alignment Module
TARN	Temporal Attentive Relation Network
TAVs	Temporal Attention Vectors
TDD	Trajectory-pooled Deep-convolutional Descriptors
TLEN	Temporal Linear Encoding Network
TSF	Temporal Structure Filter

TSN	Temporal Segment Network

Publications and the Statement of Co-Authorship

Conference Papers:

[1] Bo, Yang, Chen, Yixin, He, Wenbo and Xiang, Jie. "DVD: Constructing a Discriminative Video Descriptor by Convolving Frame Features," 2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM), Xi'an, 2018, pp. 1-5, doi: 10.1109/BigMM.2018.8499251.

Yang Bo researched the near duplicate video detection and action recognition scenario, proposed the algorithm, wrote the code, and composed the manuscript; Yixin Chen provide the results of VGG based results; Wenbo He and Jie Xiang provide the supervision of this work. The main contributor to this paper is Yang Bo (contributes more than 80%).

[2] Bo, Yang, Lu, Yangdi and He, Wenbo. "Few-Shot Learning of Video Action Recognition Only Based on Video Contents," 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, CO, USA, 2020, pp. 584-593, doi: 10.1109/WACV45572.2020.9093481.

Yang Bo researched the near duplicate video detection and action recognition scenario, proposed the algorithm, wrote the code, and composed the manuscript; Yangdi Lu reviewed the code and the paper; Wenbo He provides supervision of this work. The main contributor to this paper is Yang Bo (contributes more than 80%).

[3] Bo, Yang, Yangdi Lu, Wenbo He, Bohua Qiu and Muheng Wei. "CLTA: Contents and Length based Temporal Attention for Few-shot Video Classification." Submitted to CVPR2021.

Yang Bo researched the near duplicate video detection and action recognition scenario, proposed the algorithm, wrote the code, and composed the manuscript; Yangdi Lu reviewed the code and the paper; Wenbo He, Bohua Qiu and Muheng Wei provide supervision of this work. The main contributor to this paper is Yang Bo (contributes more than 80%).

Declaration of Academic Achievement

DVD: Constructing a Discriminative Video Descriptor by Convolving Frame Features

1. Proposed a Discriminative Video Descriptor (DVD), which is a general way to build the video descriptor on top of various frame features.
2. Demonstrated that DVD can preserve the temporal information of the entire video with various lengths. Provided the time complexity of the DVD generation and shown that DVD boosts the action recognition accuracy even with a simple SVM on multiple datasets.

Few-Shot Learning of Video Action Recognition Only Based on Video Contents

1. Proposed Temporal Attention Vectors (TAVs) which adapt to various lengths of videos to preserve the correlation among frame features.
2. Demonstrated that TAVs only introduce 2.1M parameters but outperforms the state-of-the-art video action recognition benchmarks with very few labelled training

videos.

CLTA: Contents and Length based Temporal Attention for Few-shot Video Classification

1. Proposed a Contents and Length based Temporal Attention (CLTA) for the few-shot action recognition task.
2. Shown that CLTA can generate customized temporal attention for individual video based on their contents and length. Demonstrated that CLTA achieves comparable or better performance compared to the state-of-the-art few-shot methods by even using a linear classifier and a not fine-tuned backbone.

Chapter 1

Introduction

Action recognition in a video clip is an easy task for human vision systems, but automating this procedure is challenging and important in a wide range of applications, including healthcare (Avci *et al.*, 2010; Mulroy *et al.*, 2003), human–computer interaction (Rautaray and Agrawal, 2015; Mitra and Acharya, 2007), surveillance (Vishwakarma and Agrawal, 2013; Leo *et al.*, 2004) or sociology (Coppola *et al.*, 2019, 2016). For example, if the suspicious actions can be detected automatically by a surveillance system, it can launch a warning in advance and take measures against any danger. Another instance is the use of action recognition for rehabilitation, recognizing the action that the patients are performing makes the system having the ability to evaluate the recovery status of the patients.

In computer vision, the automated action recognition can be described as a subtask of video classification. In video classification, the goal is to classify a video into one or more categories (labels), whereas in human action recognition the video must contain at least one person, and the person performs at least one action (e.g. shooting the basketball), several examples are shown in Figure 1.1. Unlike the images,

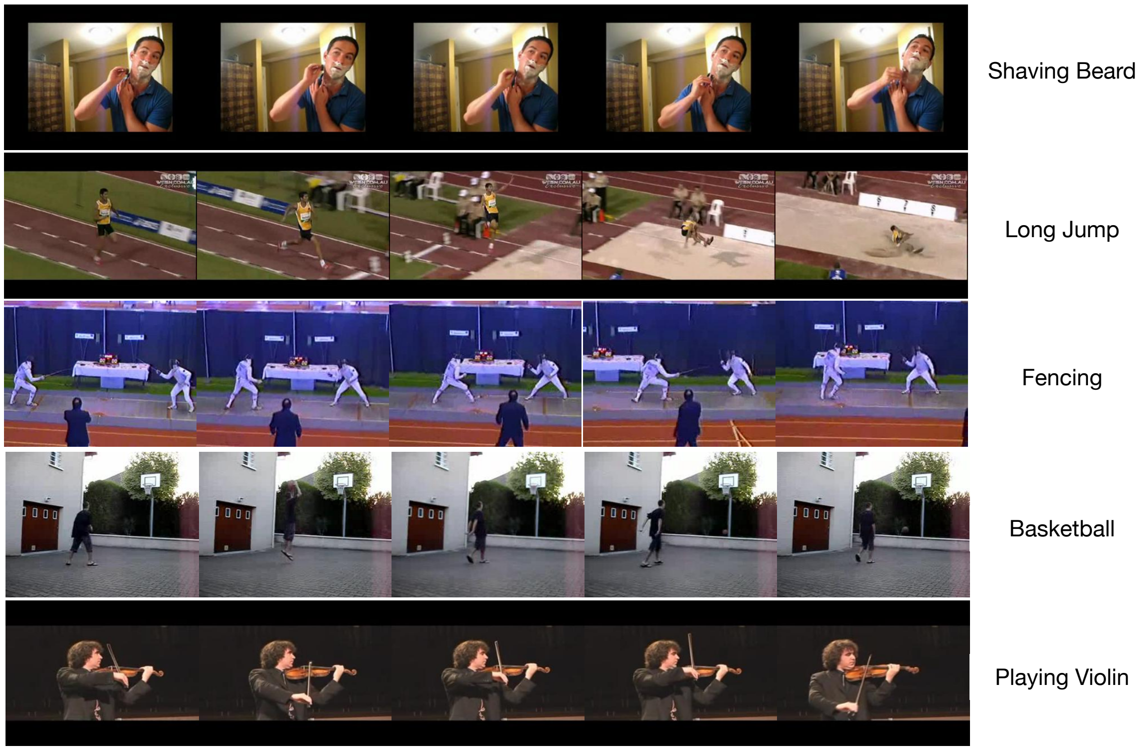


Figure 1.1: Examples of videos contained human actions and their labels.

the video has spatial aspects (the contents in individual frames) and temporal aspects (the ordering of the frames). Usually, actions can be described as the combination of multiple simpler movements of a specific body part (rising the right leg then move the right leg forward). A few actions (e.g. standing) can be probably recognized by using a single or only a few frames but more complex actions (e.g. walking vs running) might require the combination of multiple frames for better accuracy. Local temporal information plays an important role in differentiating between such actions. Moreover, long duration temporal information might be needed. Therefore, finding a good action representation, which contains both appearance and temporal information, is the foundation of human action recognition in videos.

In recent years, due to the development of powerful computing devices (e.g., GPU and distributed platforms), advanced algorithms (Convolutional Neural Networks (CNNs) (Krizhevsky *et al.*, 2012) and Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997)) and larger datasets (e.g., ImageNet data with 1000 classes (Deng *et al.*, 2009)), deep learning-based approaches defeat humans in many fields. For example, AlphaGo (Silver *et al.*, 2016) defeats human champions in the ancient game of Go and residual network (ResNet) (He *et al.*, 2016) obtains better classification performance than humans on ImageNet. In the human action recognition field, deep learning-based rather than handcraft -based approaches is becoming a trend. Lots of powerful models are proposed (Simonyan and Zisserman, 2014; Karpathy *et al.*, 2014; Tran *et al.*, 2015; Carreira and Zisserman, 2017; Feichtenhofer *et al.*, 2017; Tran *et al.*, 2018; Varol *et al.*, 2018) and achieve outstanding performance based on the abundant labeled training videos. However, acquiring labeled training samples is error-prone and labour-intensive. It is even harder to obtain correct labeled videos than images, especially for the tasks which need precise boundaries (e.g. multi-activities detection) because it is hard to say the action starts and ends at a specific frame.

Current deep learning-based approaches cannot be rapidly generalize from a few examples. In contrast, humans learn new tasks rapidly by utilizing what they learned in the past. For example, a child who learned how to add can rapidly transfer his knowledge to learn multiplication given a few examples (e.g., $3 \times 2 = 2 + 2 + 2$ and $3 \times 1 = 1 + 1 + 1$). Another example is that given a few photos of a stranger, a human can easily identify that person from a large number of photos. In order to learn from a limited number of examples with supervised information, a new machine learning paradigm called Few-Shot Learning (FSL) (Fink, 2005; Fei-Fei *et al.*, 2006)

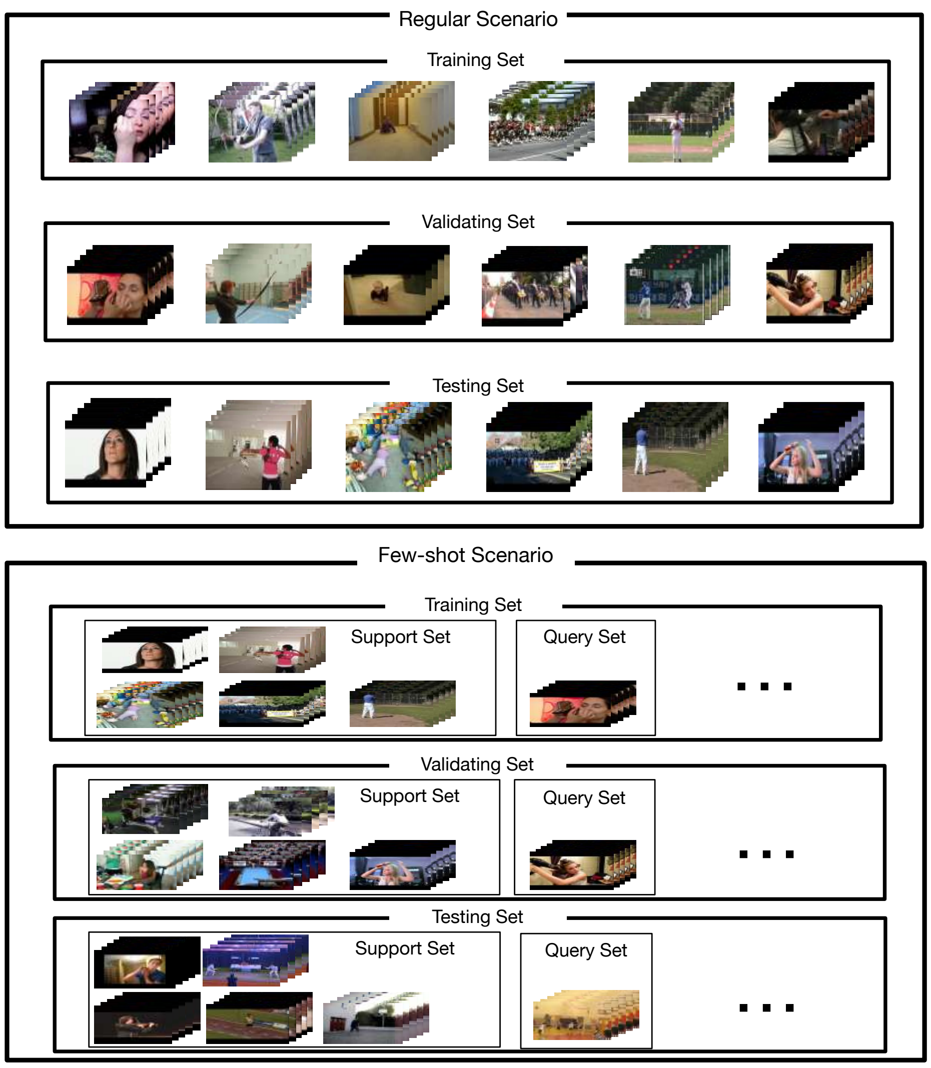


Figure 1.2: Comparison of dataset splits between human action recognition in regular and few-shot scenarios. For the regular human action recognition task, videos in training, validating and testing sets have the same labels. For the few-shot human action recognition task, videos in these three sets have no overlapped labels. However, the query set contains one or more videos that have the same labels as the videos in the support set.

is proposed. The typical scenario of few-shot human action recognition is classifying the actions given a dataset that has several different actions (classes), but each of the actions only has few (usually 1 to 10) labeled videos. In the few-shot scenario, the dataset is split into the training set, validating set, and testing set without overlapped classes (in the regular scenario , training, validating and testing sets contain the same classes but different samples from these classes. See the top figure in Figure 1.2). The training, validating and testing sets are further split into multiple episodes (similar to the concept batch in the regular scenario). Each episode contains a support set and a query set. The support and query sets are constructed as follows: 1. Randomly select N classes from the training, validating or testing set. 2. For each of the N classes, randomly select k samples to form the support set and several (usually different from k) samples to form the query set. Samples in the query set have no overlap with samples in the support set (see the bottom figure in Figure 1.2). The model is first trained on the training set and the episode which achieves the highest accuracy on the validating set is chosen. Similar to the regular scenario that the network updates its parameters every batch, in the few-shot scenario, the network updates its parameters every episode. The objective is to train the network that minimizes the N-way prediction loss of the samples in the query set. During the testing phase, the model is further fine-tuned on the testing support set then evaluated with the samples in the query set.

In the following of this chapter, we first review current deep learning-based approaches for regular and few-shot action recognition tasks as well as the widely used datasets. Then we elaborate on the objective, scope and contributions of this thesis.

1.1 Background

1.1.1 Deep Neural Network-based Action Recognition Models

Recently, Deep Neural Networks (DNNs) have been widely applied to many areas and have shown outstanding performance, such as image classification and captioning, speech recognition, natural language processing, etc. The input data for these tasks are either 1D (e.g. sentences) or 2D (e.g. images). However, human action recognition usually uses videos as input data, which contain three dimensions, width, height and time. Therefore, such high complex data make video harder to be represented by a high-level descriptor compared to the sentences or still images. The challenge of action recognition is extracting one or more descriptors that could preserve both spatial and temporal information. Over recent years, many approaches were proposed, and can be roughly separated into two categories based on how the descriptor is generated: 1. One-stage approaches. These approaches usually use one or more deep networks that take the video and/or optical flow clips as input to generate the descriptor directly. The descriptor contains both spatial and temporal information and can be directly used for classification. 2. Two-stage approaches. This type of methods first generates the frame or clip features, each of them contains partial information of the video, then aggregates these features to form the video descriptor before classification.

One-stage Approaches

As classifying human action in the video requires both appearance and motion information, early proposed deep learning-based methods (Baccouche *et al.*, 2011; Ji

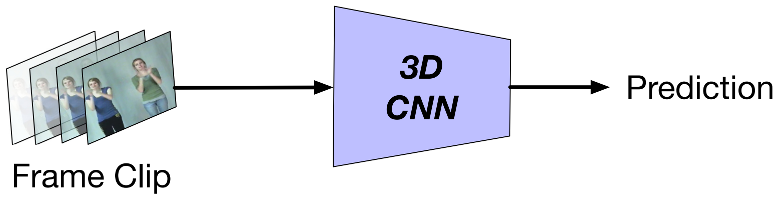


Figure 1.3: Solving the action recognition task with 3D CNN. Unlike the 2D CNNs which are used to solve image-related problems, the filters in 3D CNN have 4 dimensions, height, width, time and channel. This figure only shows a single 3D CNN, but in practice, more 3D CNNs may be adopted.

et al., 2013; Karpathy *et al.*, 2014; Tran *et al.*, 2015) apply 3D CNN to generate the video descriptor. As the name indicates, the 3D CNNs are constructed by extending an additional time dimension to the 2D CNNs. Therefore, the 3D CNNs can take short video clips as input (usually 16 frames) and capture both spatial and temporal information, see Figure 1.3. In 2011, Baccouche et al. (Baccouche *et al.*, 2011) proposed the first 3D CNN which only has 3 layers and takes the grayscaled frames as input to classify the human actions. In order to let the network can capture more information of the videos, Ji et al. (Ji *et al.*, 2012) adopted the grayscale, gradient-x, gradient-y, optical flow-x and optical flow-y of each frame as the input for their 3D network. In 2014, Karpathy et al. (Karpathy *et al.*, 2014) evaluated the 3D CNN on a large video action recognition dataset (Sports-1M). They tried to find the best temporal connectivity pattern for action recognition by evaluating multiple CNN architectures that each take a different approach to combine information across the time domain. To explore the performance of the 3D CNNs with various sized kernels on different video understanding tasks, Tran et al. (Tran *et al.*, 2015) proposed a network called C3D which contains eight 3D convolutional layers, five 3D max-pooling layers and two fully connected layers. After the experiments, they find that using the kernel with size $3 \times 3 \times 3$ (width, height, time) and pooling with stride

$2 \times 2 \times 2$ (except the first pooling layer) gives the best results. However, compared to the handcrafted approaches (e.g. improved dense trajectories (IDT) (Wang and Schmid, 2013) or Multi-skIp Feature Stacking (MIFS) (Lan *et al.*, 2015)), these 3D CNN-based methods have not demonstrated an overwhelming performance. Varol et al. (Varol *et al.*, 2018) claimed that the long-range temporal information is significant for the action recognition task. They extended the 3D CNN to accept longer video clips (maximum 100 frames) and have shown better results compared to the handcrafted methods (Wang and Schmid, 2013; Lan *et al.*, 2015). Wang et al. (Wang *et al.*, 2018a) proposed a non-local block that aims to capture the long-range relationship between frames. The block using dot products to measure the similarity of every frame feature map pairs. Different from the model proposed by Varol et al. (Varol *et al.*, 2018), non-local block not only captures the long-range temporal correlation between video frames but also the spatial information between different frame pairs. A deeper 3D CNN, called I3D (Carreira and Zisserman, 2017), was proposed recently. The authors provided an approach that could let 3D network pre-training on image datasets by repeating a single image multiple times to form a boring video. They showed that the action recognition accuracy is boosted even pre-train the 3D CNN with image data. Since the 3D CNNs bring huge computational overhead, Sun et al. (Sun *et al.*, 2015) separated the 3D convolution into the Kronecker product of 2D and 1D convolutions to reduce the number of training parameters of the network. Thus, the kernel complexity is reduced by an order of magnitude from $x \times y \times t$ to $x + y + t$ where x, y and t are the width, height and depth of the kernel, respectively. To further investigate the differences (except time complexity) between the 3D and (2+1)D convolutions, Tran et al. (Tran *et al.*, 2018) replaced the residual blocks of the resNet (He *et al.*, 2016) with the 3D residual blocks and (2+1)D residual blocks. Note that the number of parameters in the (2+1)D blocks approximately is equal to that implementing full 3D convolution. They find that the (2+1)D residual network performs better than the 3D residual network for the action recognition task. Li et al. (Li *et al.*, 2019) captured the features of video along height-weight, height-time and weight-time dimensions by applying three 2D convolution operations. The features are then fused with various weights according to the video sample. By replace 3D convolution with multiple 2D convolution operations, the network is able to choose the most important view of features during classification. Unfortunately, these deep 3D CNNs have not shown better performance compared to handcrafted methods if they are only trained on medium-sized datasets (e.g. UCF101 (Soomro *et al.*, 2012)). On the other hand, by pre-training them on large image classification datasets (e.g. ImageNet) or large video action recognition datasets (e.g. Kinetics (Kay *et al.*, 2017)), all of them dramatically outperformed the handcrafted approaches.

By analyzing the above models, we can make the following conclusions: 1. The long-range temporal information is important for human action recognition in videos. By extending the 3D network to accept longer video clips as input, the network learns the long-range temporal correlation between frames which benefits the performance. 2. The deeper or wider architectures may not necessarily increase the performance for the action recognition task. Although the deeper model has more power to express more complex functions, the outstanding performance heavily depends on abundant labeled training samples.

Another branch of works apply multiple standard CNNs to solve the human action recognition task, like Figure 1.4. Simonyan and Zisserman (Simonyan and Zisserman,

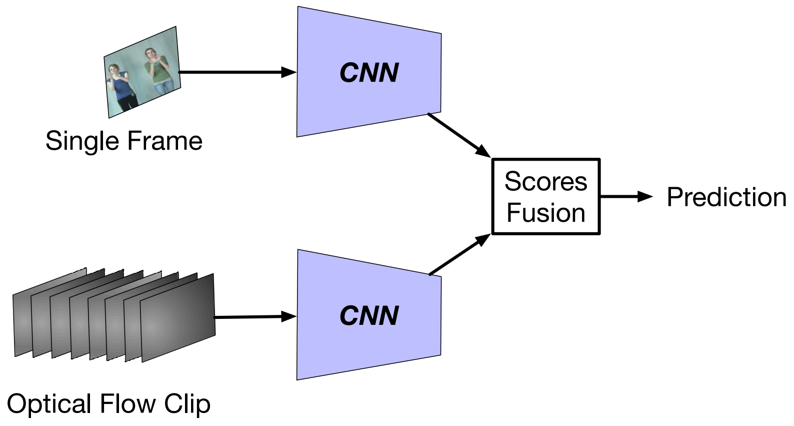


Figure 1.4: Solving the action recognition task with two-stream networks. The spatial stream takes a single frame as input meanwhile the temporal stream takes the stack of optical flow around the selected frame as input. The sub-networks of the spatial and temporal streams may have some connection between certain layers. Usually, the class scores of two streams are averaged as final scores. However, some approaches fuse (i.e. concatenate) the output before the class scores as the video descriptor then predict the result based on that.

2014) proposed a two-stream network that adopts one standard CNN (spatial stream) to capture the appearance information and try out using optical flow data in an additional standard CNN stream (temporal stream) to capture the motion information. The input for the spatial stream is a randomly selected RGB frame from the video. The input for the temporal stream is an optical flow clip of 10 consecutive RGB frames, where the frames are selected around the input of the spatial stream (the optical flow clip only has 1 color channel, thus it can be processed by a 2D CNN). The two-stream network has achieved similar performance compared to handcrafted approaches (e.g. IDT). The spatial and temporal streams make the prediction separately then the class scores are averaged as the final class score. Instead of independent processing the spatial and temporal stream, researchers further investigate the impact of fusing the feature map of the temporal stream to spatial stream earlier (Feichtenhofer *et al.*, 2016a), fusing the temporal stream to spatial stream at each block of resNet (Feichtenhofer *et al.*, 2016b) and fusing two streams at each residual block (Feichtenhofer *et al.*, 2017). By adding the interaction between two streams, the performance of action recognition has been further increased. Some researchers proposed to use both 2D and 3D CNNs to capture spatial and temporal information of video separately. ARTNets (Wang *et al.*, 2017a) are constructed by stacking multiple generic building blocks, called SMART block, whose goal is to simultaneously model appearance and motion from RGB input. Specifically, SMART blocks decouple the spatiotemporal learning module into an appearance branch for spatial modeling and a relation branch for temporal modeling. Some researchers combine the handcrafted features with the learned features. Wang et al. (Wang *et al.*, 2015) proposed the trajectory-pooled deep-convolutional descriptor (TDD) which uses the IDT algorithm (Wang and Schmid, 2013) to find the trajectories based on the original RGB frames then sum the value in learned feature maps according to the coordinates of the trajectories. To capture the long-range relationship between video frames, researchers adopted sparse sampled (Wang *et al.*, 2016) or key volume mined (Zhu *et al.*, 2016) frame stacks over the entire video as the input to the two-stream network. The evaluation of these approaches has shown the long-range temporal information boosts the performance for the action recognition task. Instead of making the prediction based on the video descriptor, Girdhar et al. (Girdhar *et al.*, 2017) proposed ActionVLAD which uses the residual vector between the frame features and the anchor vectors to make the prediction. The model subtracts the frame features with pre-selected anchors (represents a standard sub-action) to calculate the residual vector and make the prediction based on that residual vector. Also, some researchers consider adding more streams during the video classification. Wang et al. (Wang *et al.*, 2017b) proposed a spatiotemporal pyramid network that learns video features at multiple abstraction levels. More specifically, the video descriptor is generated by three parts, the spatial feature, the temporal feature and the spatial-temporal feature.

By taking a closer look at these two-stream-based approaches, we observe that the input of the motion stream is usually a short optical flow clip. It may cause the wrong prediction if two actions resemble in a short snippet, though distinguishable in the long-range (e.g. run vs long jump). Although using the sparse selected frames over the entire videos as input improves the performance of two-stream-based approaches, comparing the performance between (Wang *et al.*, 2016; Zhu *et al.*, 2016) which apply a sparse selection of frames and (Feichtenhofer *et al.*, 2016a,b, 2017) which connect the appearance and motion streams, the latter approaches achieve higher recognition accuracy. Therefore, we may conclude that sparse selection cannot guarantee that important frames are selected, these approaches may still fail to predict some composite actions.

Two-stage Approaches

Different from the above one-stage approaches (e.g. using frames and/or optical flow clips as the input of deep network to generate video descriptors directly. The network studies both spatial and temporal information of videos simultaneously), other approaches (Yue-Hei Ng *et al.*, 2015; Lan *et al.*, 2017; Fernando *et al.*, 2015; Fernando and Gould, 2016; Zolfaghari *et al.*, 2018; Kar *et al.*, 2017) apply a two-stage strategy to solve the human action recognition problem. This type of approaches

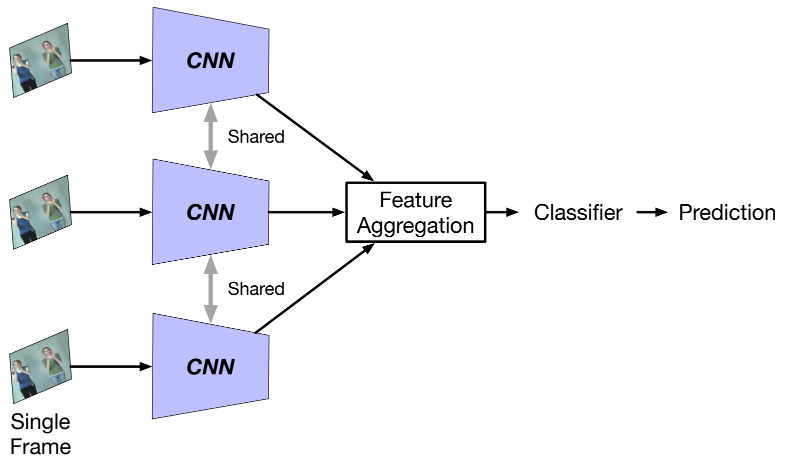


Figure 1.5: Solving the action recognition task with the feature aggregation approach. The network (2D or 3D) extract the feature for each frame (or video clip) then applies a certain algorithm to aggregate (e.g. max pool, weighted sum, etc.) these features as video descriptor. The network shares parameters when extracting features and makes the prediction based on the video descriptor.

extract the frame features first by deep networks then aggregates these features as the descriptor of the video, see Figure 1.5. (Yue-Hei Ng *et al.*, 2015; Lan *et al.*, 2017) attempted to preserve the spatial features by applying a 2D CNN over each frame and generate the video descriptor by using static pooling methods (e.g. max-pooling) in the time-domain over the frame features. However, the static-pooling operation is insensitive to the order of the video frames, hence a few actions cannot be distinguished. For example, opening the door versus closing the door are not distinguishable using max-pooling operation. To deal with this problem, Fernando and Gould (Fernando and Gould, 2016) applied a dynamic pooling approach, which is called rank-pooling, to encode the frame features. The rank-pooling operation uses a regularized support vector regression (SVR) to find a vector u such that $u{\cdot}v_a < u{\cdot}v_b$ for all v that $a < b$, where v_a and v_b are the features of frame a and b, respectively. Kar et al. (Kar *et al.*, 2017) claimed each frame should have different importance for a certain action. Therefore, they proposed an Adaptive Scan Pooling (AdaScan) approach which encodes the frame features with the weighted mean operation. The weights are calculated based on current frame features and all previous features by using a three layers MLP. Zolfaghari et al. (Zolfaghari *et al.*, 2018) proposed the Efficient Convolutional Network for Online Video (ECO) to catch the long-range temporal relations between video frames. The videos are split into the same length subsections and one frame is selected from each subsection. The ECO uses a 2D CNN to extract the frame feature maps for each of the selected frames, then a 3D CNN is applied to study the spatiotemporal information between the frame feature maps. However, these methods cannot adapt to videos with various lengths. The Recurrent Neural Networks (RNNs) have also been used with CNNs for the video action recognition task. (Donahue *et al.*, 2015; Ballas *et al.*, 2015) made an effort to use CNN + RNN to study the spatiotemporal information of the video. However, these frameworks introduce more computation overhead and do not show overwhelming performances compared to other only CNN-based approaches.

1.1.2 Few-shot Learning for Image and Video Classification

Although the DNNs have shown outstanding performance on visual recognition tasks, the performance heavily relies on the abundant labeled training instances. In a specific circumstance, examples with supervised information are hard or impossible to acquire due to privacy, safety or ethical issues. In order to learn from a limited number of examples with supervised information, a new machine learning paradigm called Few-Shot Learning (FSL) is proposed. Current FSL approaches use prior knowledge to help model recognition. Based on how prior knowledge is used, FSL approaches can be categorized into three perspectives. 1. Use the prior knowledge to augment training data. Since the number of training data is increased, a standard supervised approach can be applied and more accurate results can be obtained. 2. Use prior knowledge to reduce the size of the hypothesis space. Since the hypothesis space becomes smaller, a small number of training data is sufficient to obtain accurate results. 3. Use prior knowledge to alter the search for the best hypothesis in the given hypothesis space. Prior knowledge alters the search strategy by providing a good initialization or guiding the search steps.

Since current few-shot approaches mainly focus on image classification and few-shot image classification has strong connections with few-shot action recognition, we first introduce the recent progress of few-shot image classification in this section. After that, we will introduce current progress for few-shot action recognition.

Data Augmentation Approaches

Applying data augmentation to enrich training samples is widely used as pre-processing in the few-shot image and video classification tasks. For example, on images, one can use the translation, flipping, shearing, scaling, reflection, cropping and rotation. By introducing the augmentations, networks are able to capture different kinds of invariance of training samples. However, designing these handcraft-based rules needs the knowledge of the dataset domain and consumes expensive labor costs. Furthermore, the augmentation rules may be specific to a certain dataset which makes it hard to apply to other datasets. In order to solve this problem, some advanced data augmentation methods were proposed to solve the FSL problem. An early FSL paper (Miller *et al.*, 2000) learns a set of geometric transformations from a similar class by iteratively aligning each sample with the other samples. Some researchers applied a single transformation function between sample pairs by assuming all categories share some transformable variability (Hariharan and Girshick, 2017). Another strategy augments the dataset by selecting samples with the target label from a large data set which is weakly labeled or unlabeled. For example, apply an SVM classifier on the current dataset, then use it to predict the samples in a weakly labeled dataset and add these samples with target labels to the current dataset (Pfister *et al.*, 2014). Instead of training a classifier, label propagation is directly used to label the unlabeled data (Douze *et al.*, 2018) or select the informative unlabeled data by progressive strategy (Wu *et al.*, 2018).

Reduce the Hypothesis Space

In the presence of multiple related tasks, multitask learning-based approaches learn multiple related tasks simultaneously by exploiting both task-generic and task-specific information. For a certain FSL task, only task-specific information (reduced hypothesis) needs to be learned since the task-generic can be learned from other related tasks which have sufficient data. Hence, they can be naturally used for FSL. In (Zhang *et al.*, 2018), the authors applied two networks, each of them for different tasks, share the first few layers for the generic information, and learn different final layers to deal with different outputs. In (Motiian *et al.*, 2017), a variational auto-encoder is first pre-trained from the source tasks and then cloned to the target task. Some layers in the two variational auto-encoders are shared in order to capture the generic information while allowing both tasks to have some task-specific layers. Instead of share the network parameters, some works penalize or regularize the difference between the parameters of networks (Yan *et al.*, 2015; Luo *et al.*, 2017). In (Salakhutdinov *et al.*, 2012), a set of data sets are grouped into a hierarchy via unsupervised learning. Data sets in each group together learn the class prior probabilities. For a new few-shot class, one first finds the group this new class belongs to and then models it by the class prior drawn from the group-wise shared prior probability. In (Salakhutdinov *et al.*, 2011), the feature learning step in (Salakhutdinov *et al.*, 2012) is further improved with the use of deep Boltzmann machines (Salakhutdinov and Hinton, 2009).

Another branch of study applies embedding learning to reduce the dimension of features. By reducing the feature dimension, one can then construct a smaller hypothesis space which subsequently requires fewer training samples. Matching Network (Vinyals *et al.*, 2016) studies different embedding functions for training and testing samples. An active learning variant of Matching Network (Bachman *et al.*, 2017) labels the most important unlabeled samples and added them to the training set. The Matching Network was also extended to set-to-set matching (Choi *et al.*, 2018), which is useful in labeling multiple parts of a sample. Instead of comparing the embedding of the testing sample with the embeddings of every training sample, Prototypical Networks (ProtoNet) (Snell *et al.*, 2017) only compares it to the class prototypes. Empirically, this leads to more stable results and reduces the computation cost. This idea of using prototypes is also introduced to the Matching Network (Wang *et al.*, 2018b). In (Ren *et al.*, 2018), the authors applied the data augmentation via a semi-supervised way during training the protoNet. The Attentive Recurrent Comparators (ARC) (Shyam *et al.*, 2017) applies an LSTM with attention (Bahdanau *et al.*, 2014) to compare different regions of the testing sample with the class prototype, and then uses a bidirectional LSTM to embed all comparisons as the embedding. The Relation

Network (Sung *et al.*, 2018) uses a CNN to embed both testing and training samples, then concatenates them as the embedding, which is fed to an MLP to output the similarity score. The Graph Neural Networks (GNNs) are also used in (Liu *et al.*, 2018; Garcia and Bruna, 2017) to leverage information from local neighbourhoods.

There is another way to deal with the FSL problem which is introducing external memory. The features of training samples are first extracted and stored in the memory, then each new sample in the testing set is represented by a weighted average of contents extracted from the memory. This limits testing samples to be represented by contents in the memory, and thus essentially reduces the size of hypothesis space. Memory-Augmented Neural Networks (MANN) (Santoro *et al.*, 2016) learns the embedding function which maps samples of the same class to the same value. Samples of the same class refine their class representations in the memory together. This class representation can be viewed as a refined class prototype in ProtoNet. The abstract memory (Xu *et al.*, 2017) applied two memories. One extracts relevant key-value pairs from a fixed memory containing a large-scale machine annotated dataset, another refines the extracted values and abstracts out the most useful information for the few-shot image classification.

Improve the Searching Algorithm

According to the supervised learning theory, when the supervised information is rich, the network has enough information to update the training parameters and find an appropriate step size to update these parameters. However, in FSL related tasks, the provided dataset is not large enough, and the obtained network is unreliable. Although the network could be pre-trained on a large dataset with a similar task, simply fine-tune the training parameters may still lead to over-fitting given a few-shot dataset. To address this problem, some works apply regularization to prevent over-fitting. In (Yoo *et al.*, 2017), the filters of a pre-trained CNN are clustered together according to some auxiliary information, and then fine-tuned by group-wised back-propagation using a few-shot training dataset. A model regression network (Wang and Hebert, 2016b) captures the task-agnostic transformation which maps the parameter values obtained by training on a few examples to the parameter values that will be obtained by training on a lot of samples. Similarly, in (Kozerawski and Turk, 2018), the transformation function that maps the embedding to a classification decision boundary is learned. Some other works utilize the network which is well trained for other tasks. For example, (Wang and Hebert, 2016a) pre-trains the networks from the unlabeled data to cluster the data samples, then adapts them to the new task with the few-shot dataset. Although no supervised information is fed to the network, similar samples still can be grouped together. (Bart and Ullman, 2005) leverages the samples and classifiers from other similar classes. It first replaces the sample features from similar classes with the features of samples from the new class, then reuses the learned classifier and only adjusts the classification threshold. The improvement (Gidaris and Komodakis, 2018) was proposed to learn how to combine the existing parameters which are learned from a similar dataset. The pre-trained network may not be enough to encode the samples from a few-shot dataset, some works introduce additional parameters to tackle this problem (Hoffman *et al.*, 2013). Model-Agnostic Meta-Learning (MAML) (Finn *et al.*, 2017) utilizes a set of mata-learners, each of them is for a specific task. These learners update the network with the sum of losses over the training samples in a few-shot dataset.

Few-shot Human Action Recognition in Video

Since labeling the videos are labor-intensive and error-prone, a large number of correctly labeled videos are usually not available. Since the training samples are few, current few-shot classification approaches use prior knowledge from similar or dissimilar tasks to help the prediction. Compared to the few-shot image classification (Chen *et al.*, 2019; Finn *et al.*, 2017; Snell *et al.*, 2017; Sung *et al.*, 2018; Vinyals *et al.*, 2016), fewer efforts have been made to address the few-shot human action recognition task. The few-shot video classification needs to capture both spatial information and the temporal correlation among frames to achieve satisfactory classification accuracy. One type of approach trains a deep network on multiple datasets for different but related tasks, the network studied both task-generic and task-specific prior knowledge from different datasets. When applying the network for the few-shot classification task, it will generate more accurate features even with only few training examples since the task-generic information is already learned from other tasks and only need training a classifier. Srivastava et al. (Srivastava *et al.*, 2015) trained the two layers LSTM to predict the next 13 frames given the first 16 frames of video. The LSTM is pre-trained on 300 hours YouTube data then combined with a linear classifier to address the few-shot human action recognition task. Another branch of work address the problem by training the network to compare video samples. The intuition is that if a model can determine the similarity of two videos, it can easily classify the unseen videos with the labeled videos. Zhu et al. proposed a Compound Memory Network (CMN) (Zhu and Yang, 2018) which embeds the video as multiple descriptors. The descriptors of a video are generated by calculating the weighted sum of residual between the frame features and trainable hidden variables (salience variables). The weights are the dot product between the frame features and the salience variables. CMN introduces additional abstract memory to store the video descriptors which need to be frequently updated. The final prediction is done by comparing the Euclidean distance between the testing sample descriptors and stored descriptors. To preserve the temporal information of the video, another possible way is to align the segment (Bishay *et al.*, 2019) or frame (Cao *et al.*, 2019) features between videos, then compare the sum of similarity scores between aligned segments/frames to make the final prediction. The similarity scores can be calculated by a trainable relation module or cosine similarity. Although the temporal information is captured, these methods bring huge additional computational overhead for alignment operation. Other approaches directly increase the number of training videos by Generative Adversarial Networks (GANs) or data augmentation. Since the training samples are sufficient, the few-shot action recognition problem can be solved by standard machine learning methods. Dwivedi et al. (Kumar Dwivedi *et al.*, 2019) proposed a Prototype GAN to generate additional examples for novel actions. The generated examples are semantically similar to real examples (closed to real examples in a certain space) belonging to that class and are used to train a more generalized classifier. However, they use C3D (Karpathy *et al.*, 2014) to extract video features hence the long-range temporal information is discarded (because C3D only accepts 16 frames video clip as input). Zhang et al. (Zhang *et al.*, 2020) applied spatial/temporal jigsaw and rotation to increase the number of training samples and train a shallow 3D CNN by a self-learning manner, a relation module was used to calculate the similarity between video samples. However, the shallow 3D CNN may fail to extract distinctive video descriptors even with the number of training samples is increased.

Dataset	# Video	# Class	Background	Camara Motion	Year	Resource
KTH	2391	6	Static	Slight	2004	Actor Staged
HOLLYWOOD2	2517	12	Dynamic	Yes	2009	Movies
UCF Sports	182	10	Dynamic	Yes	2009	TV, Movies
HMDB51	6681	51	Dynamic	Yes	2010	Youtube
UCF101	13320	101	Dynamic	Yes	2012	Youtube
Sports-1M	1133158	487	Dynamic	Yes	2014	Youtube
Charades	9848	157	Dynamic	Yes	2016	Actor Staged
Kinetics	500000	600	Dynamic	Yes	2017	Youtube
Something Something	220847	174	Almost static	Slight	2017	Actor Staged

Table 1.1: Summary of Major Action Recognition Datasets.

1.1.3 Dataset

The number of action recognition datasets has grown steadily in recent years. These datasets can vary widely in a number of characteristics. Not only because they differ in the number of action categories and videos, but also in the average length of video clips and resolution, whether videos are trimmed (i.e. they are short clips containing only the action of interest) or untrimmed (i.e. actions can happen anywhere in a long video not necessarily focused in the action), whether they are multi-class (i.e. a video can belong only to a single class) or multi-label (i.e. a clip can belong to multiple classes at the same time), and in the different data modalities and ground-truth annotations they provide. Each dataset is normally also associated with a particular evaluation protocol that needs to be followed when reporting results in the literature to ensure performance numbers are always comparable across different works. We summarize the widely used datasets in Table 1.1, and introduce the details of each dataset.

KTH

The KTH dataset (Schuldt *et al.*, 2004) contains 6 classes of human actions performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. The dataset contains 2,391 videos. All videos are taken over homogeneous backgrounds with a static camera with 25fps frame rate. The videos are down sampled to the spatial resolution of 160×120 pixels and have a length of four seconds on average.

HOLLYWOOD2

HOLLYWOOD2 dataset (Marszalek *et al.*, 2009) provides a dataset with 12 classes of human actions distributed over 3,669 video clips and approximately 20.1 hours of video in total. The dataset intends to provide a comprehensive benchmark for human action recognition in realistic and challenging settings. The dataset is composed of video clips from 69 movies.

UCF Sports

UCF Sports dataset (Rodriguez *et al.*, 2008) consists of various sports actions collected from broadcast television channels including ESPN and BBC. The dataset represents a natural pool of actions featured in a wide range of scenes and viewpoints. The dataset includes 10 actions and total of 150 videos. All videos have the resolution of 720×480.

HMDB51

HMDB51 dataset (Kuehne *et al.*, 2011) collects videos from various sources, mostly from movies, and a small proportion from public databases such as the Prelinger archive, YouTube and Google videos. The dataset contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips. The actions categories can be grouped into five types: 1. General facial actions. 2. Facial actions with object manipulation. 3. General body movements. 4. Body movements with object interaction. 5. Body movements for human interaction.

UCF101

The UCF101 dataset (Soomro *et al.*, 2012) has 13,320 videos from 101 action categories, and gives the largest diversity in terms of actions and with the presence of large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc. The videos in 101 action categories are grouped into 25 groups, where each group can consist of 4-7 videos of an action. The videos from the same group may share some common features, such as similar background, similar viewpoint, etc.

Sports-1M

The Sports-1M dataset (Karpathy *et al.*, 2014) consists of 1,133,158 YouTube videos annotated with 487 classes. The classes are arranged in a manually-curated taxonomy that contains internal nodes such as Aquatic Sports, Team Sports, Winter Sports, Ball Sports, Combat Sports, Sports with Animals, and generally becomes fine-grained by the leaf level. For example, the dataset contains 6 different types of bowling, 7 different types of American football and 23 types of billiards. There are 1000 to 3000 videos per class and approximately 5% of the videos are annotated with more than one class. The annotations are produced automatically by analyzing the text metadata surrounding the videos.

Charades Dataset

Charades dataset (Sigurdsson *et al.*, 2016) collects hundreds of people recording videos in their own homes, acting out casual everyday activities. The dataset is composed of 9,848 annotated videos with an average length of 30 seconds, showing activities of 267 people from three continents, and over 15% of the videos have more than one person. Each video is annotated by multiple free-text descriptions, action labels, action intervals and classes of interacted objects. In total, Charades provides 27,847 video descriptions, 66,500 temporally localized intervals for 157 action classes and 41,104 labels for 46 object classes.

Kinetics Dataset

Kay et al. (Kay *et al.*, 2017) proposed a deepMind Kinetics human action video dataset. The dataset consists of approximately 500,000 video clips, and covers 600 human action classes with at least 600 video clips for each action class. Each clip lasts around 10 seconds and is labeled with a single class. The actions cover a broad range of classes including human-object interactions such as playing instruments, as well as human-human interactions such as shaking hands and hugging.

Something Something V2 Dataset

The Something-something dataset (Goyal *et al.*, 2017) collects a large number of densely-labeled video clips that show humans performing pre-defined basic actions with everyday objects. It contains 220,847 videos and for each video in the training and validation sets the object annotations and video labels are provided. For example, for a label like ”Putting [something] onto [something]” there is also an annotated version like ”Putting a cup onto a table”. In total, there are 318,572 annotations involving 30,408 unique objects.

1.1.4 The Datasets for Few-shot Human Action Recognition

Since addressing the few-shot human action recognition does not require large-scale datasets, researchers modify the existing action recognition datasets to make them more reliable under the few-shot setting. The dataset is split into the training set, validating set, and testing set without overlapped classes. Usually, the model is trained on the training set and the validating set is used to decide the best number of training episode of the model. The support set is constructed by uniformly selecting n classes from the training/validating/testing set, each of them contains k randomly selected samples. The query set contains several samples from one of the n classes. Samples in the query set have no overlap with samples in the support set.

UCF101 for few-shot learning

Zhang et al. (Zhang *et al.*, 2020) uniformly selected 70 classes as the training set, 10 classes as the validating set and 21 classes as the testing set. The detailed classes for each set are shown in the appendix of this chapter.

HMDB51 for Few-shot Learning

Zhang et al. (Zhang *et al.*, 2020) split the HMDB51 dataset into the training (31 classes), validating (10 classes) and testing sets (10 classes). The details of splits are shown in the appendix of this chapter.

Kinetics for Few-shot Learning

Zhu et al. (Zhu and Yang, 2018) provided the modified kinetics for the few-shot video classification which randomly chooses 64 classes for training, 12 classes for validation and 24 classes for testing, each of the classes contains 100 uniformly chosen videos. The details of splits are shown in the appendix of this chapter.

Something Something V2 for Few-shot Learning

Cai et al. (Cao *et al.*, 2019) reported that they randomly selected 100 classes from the whole dataset. The 100 classes are then split into 64, 12 and 24 classes as the training, validation and testing set, respectively. However, the details of classes are not provided in (Cao *et al.*, 2019).

1.1.5 Performance of Recent Approaches on Different Datasets

In this section, we report the action recognition accuracy of current deep networks-based approaches on different datasets, see Table 1.2. The first and second part of Table 1.2 contains the performances of the one-stage approaches and the two-stage approaches, respectively. The UCF101 and HMDB51 are the most widely used datasets for the human action recognition task. The Kinetics dataset is became widely used by the most recent approaches.

Approaches	KTH	HOLLOYWOOD2	UCF-Sports	HMDB51	UCF101	Sports-1M	Charades	Something Something	Kinetics
3D CNNs (Ji *et al.*, 2013)	91.7								
3D CNNs (Baccouche *et al.*, 2011)	94.4								
None-local (Wang *et al.*, 2018a)							39.5		77.7
Fusion CNNs (Karpathy *et al.*, 2014)					65.4	60.9			
C3D (Tran *et al.*, 2015)					85.2	61.1			
F$_{ST}$CN (Sun *et al.*, 2015)				59.1	88.1				
Two-stream (Simonyan and Zisserman, 2014)				59.4	88				
Key Volume Mining (Zhu *et al.*, 2016)				63.3	93.1				
LTC (Varol *et al.*, 2018)				64.8	91.7				
TDD (Wang *et al.*, 2015)				65.9	91.5				
ActionVLAD (Girdhar *et al.*, 2017)				66.9	92.7				
ST Pyramid Network (Wang *et al.*, 2017b)				68.9	94.6				
Two-stream Fusion (Feichtenhofer *et al.*, 2016a)				69.2	93.5				
TSN (Wang *et al.*, 2016)				69.4	94.2				
ST-ResNet (Feichtenhofer *et al.*, 2016b)				70.3	94.6				
ARTNet (Wang *et al.*, 2017a)				70.9	94.3				78.7
SMN (Feichtenhofer *et al.*, 2017)				72.2	94.9				
R3D (Tran *et al.*, 2018)				78.7	97.3	73.3			75.4
I3D (Carreira and Zisserman, 2017)				80.9	97.8				74.2
Rank Pooling (Fernando and Gould, 2016)		40.6	87						
AdaScan (Kar *et al.*, 2017)				54.9	89.4				
TLE (Diba *et al.*, 2017)				71.1	95.6				
DOVF (Lan *et al.*, 2017)				71.7	94.9				
ECO (Zolfaghari *et al.*, 2018)				72.4	94.8			43.9	70
LRCN (Donahue *et al.*, 2015)					82.6				
GRU-RCN (Ballas *et al.*, 2015)					85.7				
Conv Pooling (Yue-Hei Ng *et al.*, 2015)					88.6	71.8			

Table 1.2: The recognition accuracy of current deep learning-based approaches on different action recognition datasets.

	UCF101		HMDB51		Kinetics		SomethingV2	
Methods	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Prototype GAN (Kumar Dwivedi *et al.*, 2019)	57.8	80.2	34.7	54.0				
3D Prototypical Net (Zhang *et al.*, 2020)	57.1	78.3	38.1	53.2				
3D RelationNet (Zhang *et al.*, 2020)	58.2	78.4	38.2	53.2				
3D SoSN (Zhang *et al.*, 2020)	62.6	81.5	40.8	55.2				
ARN (Zhang *et al.*, 2020)	66.3	83.1	45.5	60.6				
Matching Net (Zhu and Yang, 2018)					53.3	74.6		
MAML (Zhu and Yang, 2018)					54.2	75.3		
CMN (Zhu and Yang, 2018)					60.5	78.9		
TARN (Bishay *et al.*, 2019)					66.6	80.7		
TSN++ (Cao *et al.*, 2019)					64.5	77.9	33.6	43.0
CMN++ (Cao *et al.*, 2019)					65.4	78.8	34.4	43.8
TRN++ (Cao *et al.*, 2019)					68.4	82.0	38.6	48.9
TAM (Cao *et al.*, 2019)					73.0	85.8	42.8	52.3

Table 1.3: Recent approaches reported mean accuracy of 5-way video classification on UCF101, HMDB51, Kinetics and SomethingV2.

We also report the recent few-shot human action recognition approaches in Table 1.3. These approaches use the datasets for FSL as mentioned in Section 1.1.4. Most approaches use the few-shot Kinetics as a standard evaluation dataset.

1.2 Objective and Scope

Upon the study of current approaches for human action recognition in videos, we made several observations that heavily affect the recognition accuracy. First of all, although the deep models (e.g. I3D (Carreira and Zisserman, 2017), ST-Multiplier (Feichtenhofer *et al.*, 2017)) show better performance compared to the shallow models (C3D (Tran *et al.*, 2015)), the performance heavily relies on abundant labeled data. Without enough training data, deep models do not necessarily outperform other approaches. Second, by utilizing additional input resources (e.g. optical flow), the deep learning-based models can capture more distinctive features of the video which benefit the action recognition task. Third, the long-range temporal information is important for identifying human actions in videos. Extending the temporal dimension of 3D CNNs to accommodate longer video clips (Varol *et al.*, 2018) or using sparse selection approaches (Wang *et al.*, 2016; Lan *et al.*, 2017; Zolfaghari *et al.*, 2018) are both effective ways to enable networks to study the long-range temporal information. However, both ways fail to preserve the temporal information of the entire video since the input size (length of video clips) is limited. Fourth, current few-shot human action recognition approaches (Zhu and Yang, 2018; Bishay *et al.*, 2019; Cao *et al.*, 2019) mainly focus on using the unsupervised classifiers to identify the class of the action in videos (e.g. compare the similarity/distance between the descriptor of labeled and unlabeled videos). A major reason is that the unsupervised classifiers do not need to be trained with labeled data, thus the number of labeled data does not affect their performance.

Given the above observations and issues, the objective of this research is to explore and develop new approaches that learn the video descriptor from a limited number of training samples for human action recognition in videos. The aim is to reduce the dependency on large manually annotated collections of videos, helping the research in action recognition become more feasible for those with limited computing resources, or for those with limited personnel to collect and annotate data for new action recognition tasks. To develop new approaches, the following questions should be answered.

1. Since long-range temporal information is important for action recognition, would it be possible to preserve the temporal information of the entire video? Simple 1D/3D convolution fails to fully preserve the temporal information since they only accept the fixed-size input. CNN+RNN model could accept the inputs with various sizes, but it would severely overfit in few-shot scenarios. Would it be possible to use the weighted sum to aggregate the frame features and fully preserve the temporal information? How to generate the weights for each video frame?

2. A major problem for few-shot learning is how to prevent over-fitting. One of the possible solutions is reducing the training parameters of the network. Would it be possible to use two-stage approaches to solve the few-shot human action recognition in videos? Given the fact that it is hard to train a deep network given only few training videos, is it possible to use an image pretained backbone to extract frame features without fine-tuning then adopting certain methods to preserve temporal information?

3. Current approaches for few-shot human action recognition in videos mostly measure the distance between the descriptor of videos to identify the class of actions. However, the power of the linear classifier may be underestimated. Would it possible to use a linear classifier for few-shot action recognition in videos? How many labeled videos are sufficient to train a linear classifier?

Leave the above questions aside, at the same time we will be limiting the scope of this thesis to keep the research feasible. In particular, we only consider recognizing the actions performed by the real person, which means the action recognition of the skeleton is not addressed. Also, in this work, we focus on visual feature generation, and audio features will not be addressed in the main course of the dissertation.

1.3 Contributions

The main contribution of this thesis shall be the investigation of the different ways to generate effective video descriptors for the few-shot human action recognition in videos. We focus on temporal information preservation (frame/segment correlation) in this thesis. The frame/segment features are handcrafted (e.g. SIFT or SURF) or extracted by image pre-trained 2D CNNs. More specifically, in this work we made the following contributions:

1. We built a general video descriptor on top of various frame/segment features, called Discriminative Video Descriptor (DVD). DVD preserves the temporal information of the entire video by recursively convolving the frame/segment feature sequence with the basis for a lower-dimensional space over the temporal dimension (similar with the convolutional operation on temporal dimension in 3D CNN, except the filter values are pre-defined and fixed). We demonstrated that DVD can preserve the temporal information of entire videos with various lengths. We provided the time complexity of the DVD generation and demonstrate DVD boosts the action recognition accuracy even with a simple SVM on multiple datasets.

2. We proposed a frame/segment features aggregation approach which adapts various lengths of videos, called Temporal Attention Vectors (TAVs). TAVs preserve the

temporal information of video by aggregating the temporal weighted frame/segment features where the temporal weights are manually defined (not trainable). We showed that TAVs extremely reduce the number of training parameters compared to the two-stream and CNN+LSTM-based approaches and can be fast calculated. We introduced various ways to generate the value of TAVs and report their performance. We experimentally demonstrated that the optimal design of TAVs outperforms the previous state-of-the-art human action recognition benchmarks in few-shot scenarios with not fine-tuned backbones (ImageNet pre-trained resNet152 and Kinetics pre-trained I3D). We also demonstrated that TAVs achieve the same level of performance compared to the state-of-the-art approaches in regular action recognition scenario.

3. We proposed a temporal attention generation approach which adapts video with various lengths to address the few-shot human action recognition task, called Contents and Length -based Temporal Attention (CLTA). CLTA uses Gaussian distribution function as the template to generate temporal attention weights for each frame and the mean and standard deviation are studied from frame contents and video length. We provided visual analysis between CLTA and other temporal attention approaches, which demonstrates that CLTA generates customized temporal attention for different videos. Compared to other temporal attention approaches that share the same temporal weights over different videos, CLTA achieves better few-shot action recognition accuracy. Combining CLTA with a linear classifier (e.g. soft-max classifier), we showed that CLTA achieves comparable (1-shot scenario) or better (5-shot scenario) few-shot action recognition accuracy compared to the state-of-the-art method on various datasets.

1.4 Outline

This thesis is organized as follows. We introduce the generation details of DVD approaches, as well as the experiment results in chapter 2. In chapter 3, we introduce the TAVs and show the experiments with various fusion methods. We also show the comparison with the state-of-the-art in the regular and few-shot scenario. In chapter 4, we introduce the CLTA approaches and explain the difference from other temporal attention approaches. We evaluate CLTA on three datasets in the few-shot scenario and compare it to other temporal attention and state-of-the-art approaches. We conclude the thesis in chapter 5.

1.5 Appendix

1.5.1 The splits details for few-shot UCF101 dataset

Actions of Train Split

ApplyEyeMakeUp, Archery, BabyCrawling, BalanceBeam, BandMarching, BaseballPitch, Basketball, BasketballDunk, BenchPress, Biking, Billiards, BlowDryHair, BodyWeightSquats, Bowling, BoxingPunchingBag, BoxingSpeedBag, BreastStroke, BrushingTeeth, CricketBowling, Drumming, Fencing, FieldHockeyPenalty, FrisbeeCatch, FrontCrawl, Haircut, Hammering, HeadMassage, HulaHoop, JavelinThrow, JugglingBalls, JumpingJack, Kayaking, Knitting, LongJump, Lunges, MilitaryParade, Mixing, MoppingFloow, Nunchucks, ParallelBars, PizzaTossing, PlayingCello, PlayingDhol, PlayingFlute, PlayingPiano, PlayingSitar, PlayingTabla, PlayingViolin, PoleVault, Pullups, PushUps, Rafting, RopeClimbing, Rowing, ShavingBeard, Skijet, SoccerJuggling, SoccerPenalty, SumoWrestling, Swing, TableTennisShot, Taichi, ThrowDiscus, TrampolineJumpling, Typing, UnevenBars, WalkingWithDog, WallPushups, WritingOnBoard, YoYo.

Actions of Validation Split

ApplyLipstick, CricketShot, HammerThrow, HandstandPushups, HighJump, HorseRiding, PlayingDaf, PlayingGuitar, Shotput, SkateBoarding.

Actions of Test Split

BlowingCandles, CleanAndJerk, CliffDiving, CuttingInKitchen, Diving, FloorGymnastics, GolfSwing, HandstandWalking, HorseRace, IceDancing, JumpRope, PommelHorse, Punch, RockClimbingIndoor, SalsaSpin, Skiing, SkyDiving, StillRings, Surfing, TennisSwing, VolleyballSpiking.

1.5.2 The splits details for few-shot HMDB51 dataset

Actions of Train Split

brush hairs, catch, chew, clap, climb, climb stairs,dive, draw sword, dribble, drink, fall floor, flic flac, handstand, hug, jump, kiss, pullup, punch, push, ride bike, ride horse, shake hands, shoot bow, situp, stand, sword, sword exercies, throw, turn, walk, wave.

Actions of Validation Split

cartwheel, eat, golf, hit, laugh, shoot ball, shoot gun, smile, somersault, swing baseketball.

Actions of Test Split

fencing, kick, kick ball, pick, pour, pushup, run, sit, smoke, talk.

1.5.3 The splits details for few-shot Kinetics dataset

Actions of Train Split

air drumming, arm wrestling, beatboxing, biking through snow, blowing glass, blowing out candles, bowling, breakdancing, bungee jumping, catching or throwing baseball, cheerleading, cleaning floor, contact juggling, cooking chicken, country line dancing, curling hair, deadlifting, doing nails, dribbling basketball, driving tractor, drop kicking, dying hair, eating burger, feeding birds, giving or receiving award, hopscotch, jetskiing, jumping into pool, laughing, making snowman, massaging back, mowing lawn, opening bottle, playing accordion, playing badminton, playing basketball, playing didgeridoo, playing ice hockey, playing keyboard, playing ukulele, playing xylophone, presenting weather forecast, punching bag, pushing cart, reading book, riding unicycle, shaking head, sharpening pencil, shaving head, shot put, shuffling cards, slacklining, sled dog racing, snowboarding, somersaulting, squat, surfing crowd, trapezing, using computer, washing dishes, washing hands, water skiing, waxing legs, weaving basket.

Actions of Validation Split

baking cookies, crossing river, dunking basketball, feeding fish, flying kite, high kick, javelin throw, playing trombone, scuba diving, skateboarding, ski jumping, trimming or shaving beard.

Actions of Test Split

blasting sand, busking, cutting watermelon, dancing ballet, dancing charleston, dancing macarena, diving cliff, filling eyebrows, folding paper, hula hooping, hurling (sport), ice skating, paragliding, playing drums, playing monopoly, playing trumpet, pushing car, riding elephant, shearing sheep, side kick, stretching arm, tap dancing, throwing axe, unboxing.

Chapter 2

DVD: Constructing a Discriminative Video Descriptor by Convolving Frame Features

2.1 Citation and Main Contributor

Bo, Yang, Chen, Yixin, He, Wenbo and Xiang, Jie. ”DVD: Constructing a Discriminative Video Descriptor by Convolving Frame Features,” 2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM), Xi’an, 2018, pp. 1-5, doi: 10.1109/BigMM.2018.8499251.

The main contributor to this paper is the first author - Yang Bo (contributes more than 80%).

2.2 Copyright

2.3 Abstract

The core to organize, classify, search, compare and retrieve videos is comparing the video descriptors. In this paper, we propose a Discriminative Video Descriptor (DVD) which is a general way to build the video descriptors on top of various frame features. We built the DVD on top of the HSV-color distribution and evaluated its performance for the Near-Duplicate Video Detection task by using the CC WEB VIDEOS dataset. The average detection accuracy achieved 94.4%. We also evaluated the DVD for Human Action Recognition task by building the DVD on top of the 3D-SIFT with Weizmann human action dataset. The average recognition accuracy achieved 97.84%. In practice, the DVD only introduces slightly computational overhead. The average time to build the DVD on top of the HSVcolor distribution and 3D-SIFT for a single video was 0.128s (average 11 frames) and 0.04s (200 interest points), respectively.

2.4 Introduction

In spite of recent breakthroughs in image big data analysis (e.g. ImageNet competition), it is still an enormous headache when we confront the welter of web videos. Video data, considered as the biggest big data, belongs to the unstructured data which refers to the information that either does not have a pre-defined data model or is not organized in a predefined manner. Hence, it is challenging to organize, classify, search, compare and retrieve video contents, especially when video data are uploaded at a tremendous speed and dominant in the volume of Internet traffic. In practice, we seek to design an effective descriptor to present the video contents and efficiently extract the video descriptors. Hence, the content-based comparison, classification, and information retrieval will be accomplished by comparing video descriptors (or features).

A video is composed of a frame sequence, hence a video descriptor is usually generated based on individual video frame. There are three categories of the existing approaches to generate the video descriptor. (1) Aggregate the frame features to form a video descriptor as histogram (Chen *et al.*, 2016; Wang *et al.*, 2011) makes the extraction and comparison of video descriptors efficient, but the temporal information of video frames cannot be preserved. As an example, the same set of frames but with different orders will yield the same video descriptor. (2) Find certain number of interest points over the video and extract the features of them from 3D space, then cluster with the bag of words (Scovanner *et al.*, 2007; Laptev, 2005) or Fisher Vector representations (Perronnin *et al.*, 2010). However, this approach only preserve the spatiotemporal information locally (the neighbours of the interest points). (3) Use the deep networks to extract the video descriptor. This can be done by using the 3D Convolutional Neural Networks (CNNs) (Ji *et al.*, 2012; Tran *et al.*, 2015; Varol *et al.*, 2018) or applying two 2D CNNs to independently learn the spatial and temporal information (Feichtenhofer *et al.*, 2017, 2016a; Simonyan and Zisserman, 2014). However, due to the fixed input size limitation of the CNNs, it is impossible to learn the temporal correlation between all frames for various length videos.

In this paper, we propose a Discriminative Video Descriptor (DVD) which can preserve the temporal information among all video frames by representing the high dimensional feature sequences with multiple lower dimensional mappings. We show that the DVD can be built on the wide range of frame features (e.g. global or local features) and improves the video classification accuracy. In addition, only slight computational overhead is introduced to generate the DVD. To verify the effectiveness, robustness and efficiency of the DVD, we evaluate the DVD for both Near Duplicate Video Detection and Human Action Recognition tasks. We observe the significant performance improvement in both scenarios. Our contributions are as follows.

1. We designed a discriminative video descriptor, which is a general way to build the video descriptor on top of various frame features. By convolving the frame feature sequences with the basis for a lower dimensional space, we obtain multiple lower dimensional feature sequence mappings. If we use the concatenation of the mappings as the video descriptor, we will get the same descriptor of two videos if two sequences of frame features are identical.

2. To compare the DVD with other video descriptors, we implement two Near Duplicate Video Detection systems which are adopted hand-craft features (SIFT (Lowe, 2004) or SURF (Bay *et al.*, 2006)) and Convolutional Neural Networks (CNNs) based features (generated by ImageNet dataset (Deng *et al.*, 2009) pre-trained 16-layer and 19-layer VGGNets (Simonyan and Zisserman, 2014)). Both of the systems average the frame features as the video descriptor and use K-nearest neighbours (KNN) approach and softmax score to make the final prediction, respectively. For the Human Action Recognition, the baseline uses Dollar's (Dollár *et al.*, 2005) approach to select the Space Time Interest Points(STIPs), then uses the bag of words (BoW) representation of the 3D-SIFT features (Scovanner *et al.*, 2007) as the video descriptor. A linear support vector machine (SVM) is used to make the final prediction.

3. We evaluate the performance of the DVD in different experiment scenarios and over different frame features. The results show that adopting the DVD lead to a 94.4% average detection accuracy which gives more than 14% improvement compared to the baselines and the Video Cuboid Based Detection System (Zhou and Chen, 2010) for the Near Duplicate Video Detection. By building the DVD on top of the 3D-SIFT, 97.84% average recognition accuracy is achieved for the Human Action Recognition task. Comparing to using the 3D-SIFT in the same scenario, the recognition accuracy improvements are 5.37%.

In the rest of the paper, we summarize the related work in Section 2.5 (Section 2). In Section 2.6 (Section 3), we propose the DVD generation approach and show the time complexity. In Section 2.8 (Section 4), we describe the evaluation scenarios and experiment results. Finally, we conclude the paper in Section 2.9 (Section 5).

2.5 Related Work

Constructing a content-based video descriptor is the key component in video classification tasks, such as action recognition (Laptev, 2005), near-duplicate video detection (Chen *et al.*, 2016), event and action detection (Over *et al.*, 2013). The classical image interest points and feature extraction approaches are extended to 3D (e.g. 3D-STIPs (Laptev, 2005), 3D-SIFT (Scovanner *et al.*, 2007), 3D-HoG (Klaser *et al.*, 2008)). In 2010, Xiangmin Zhou and Lei Chen (Zhou and Chen, 2010) proposed a video cuboid signature to monitoring near duplicates video streams. Qianru Sun and Hong Liu proposed the Normalized Google-Like Distance Correlogram (NGLDC) approach (Sun and Liu, 2012) which adopts the Normalized Google Distance (NGD) to measure the co-occurrence distance of pairwise visual words. Heng Wang and Cordelia Schmid (Wang and Schmid, 2013) proposed an Improved Dense Trajectories (IDT). In 2016, Yixin Chen, Wenbo He, Yu Hua and Wen Wang (Chen *et al.*, 2016) sought using multiple image feature aggregation to reduce the computational overhead of the video descriptor generation in a Near Duplicate Video Detection system. Recently, the convolutional neural networks (CNNs) are applied to the problem of human pose estimation in videos (Jain *et al.*, 2014) and video feature learning in an unsupervised setting (Le *et al.*, 2011). The CNNs are extended into 3 dimensions to study the video descriptors (Ji *et al.*, 2012; Karpathy *et al.*, 2014; Tran *et al.*, 2015). Gul Varol, Ivan Laptev and Cordelia Schmid (Varol *et al.*, 2018) improved the 3D-CNNs by extending the temporal size of video clips. In (Simonyan and Zisserman, 2014), the two stream approach is introduced which uses pretrained one 2D CNNs to learn spatial information from static RGB frames and uses another to learn temporal information from flow inputs. Based on that, Feichtenhofer et al. apply multiplicative gate to combine the two streams in (Feichtenhofer *et al.*, 2016a).

2.6 Discriminative Video Descriptor

Since videos are made up of static frames, which play at roughly 30 frames per second sequentially giving us an illusion of motion, extracting features from a frame independently of the past or future frames in the video is less stable than extracting features that contain temporal correlation of frames in the whole video. In this section, we propose a Discriminative Video Descriptor (DVD) which can be built on various frame features while containing the temporal correlation of the frame sequence. Meanwhile,

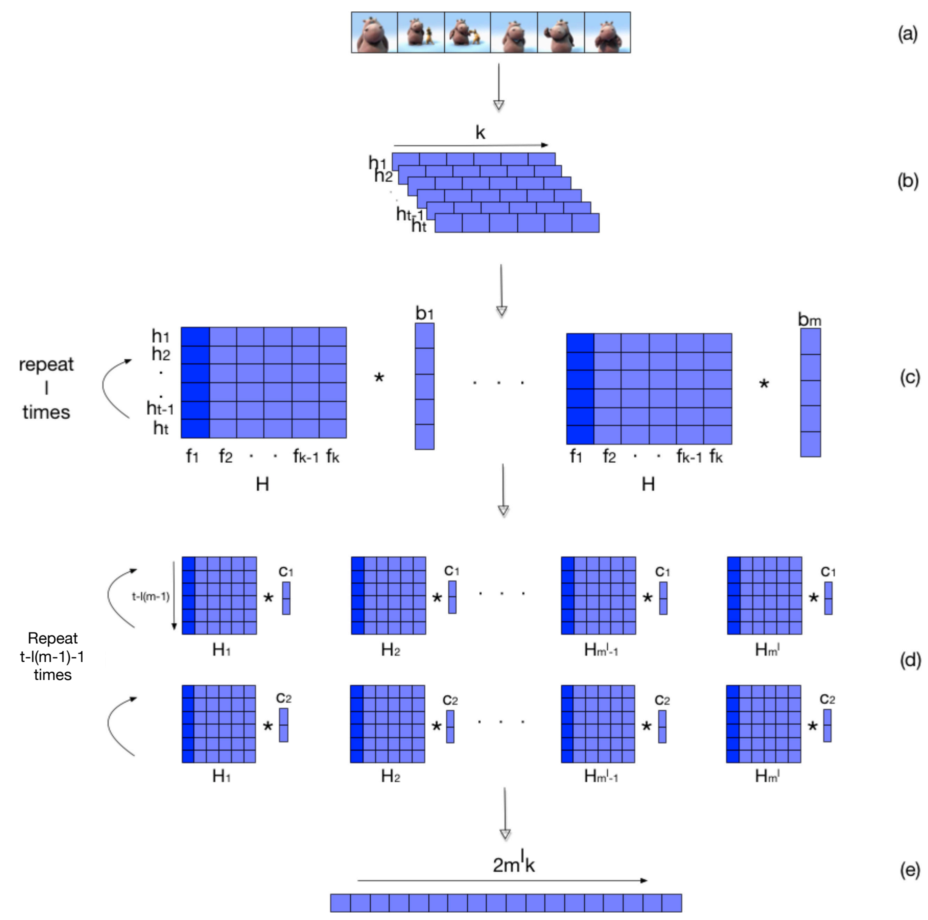


Figure 2.6: Overview of the DVD generation approach: (a) Extract the frame of a video; (b) Extract the frame features; (c) Arrange the frame features over time to form the feature matrix and randomly choose a non-diagonal basis B for $\mathbb{R}^m$. Convolve each basis vector with every column of the feature matrix l times without 0 padding; (d) Randomly choose a non-diagonal basis C for $\mathbb{R}^2$, repeat same operations as the step (c) until all feature matrices contain only one row; (e) Concatenate all feature matrices to form the video descriptor.

only slightly time overhead is needed to compute the DVD. To design a stable and efficient video descriptor, we adopt the convolution which represents the overlap of one function as it is shifted over another. It is like "projecting" one function or sequence onto another. Hence, this allows us to transform the sequence of frame features in a "projected representation". If we find the linearly independent vectors for such representation space, we will be able to use the projected representation to preserve the temporal information of the feature sequence.

2.6.1 DVD Generation

The core of the DVD is letting the frame features to preserve the spatial information of video frames independently then representing each high dimensional feature sequence with multiple lower dimensional feature vectors by convolving the sequence with lower dimensional basis vectors to preserve the temporal information of the video. The DVD generation approach is shown in Figure 2.6. Assume a video v contains t frames as shown in Figure 2.6 (a). First, we extract the feature for frame i and represent them as the fixed length k histograms h_i, as shown in Figure 2.6 (b). Each bin in the histograms can be seen as a 1×1 feature map for the static frames. Therefore, one frame is described by k features. Next, we arrange the feature histograms in time order to describe the video v by an $t \times k$-dimensional feature matrix H. Each row in H is a frame feature histogram (row 1 is the feature histogram for frame 1 denotes as h_1 and so on). Each column in H is a feature sequence which describes the change of certain feature over time (column 1 is the feature sequence for the video called f_1 and so on). Thus, we can preserve the temporal correlation between frames by preserving the temporal correlation of the feature sequences. We also choose a non-diagonal basis $B = (b_1, b_2, \ldots, b_m)$ from $\mathbb{R}^m$. The value of elements in B are chosen from random normal distribution. We then convolve each basis vector with every feature sequence in H without 0 padding l times as shown in Fig 2.6 (c). We call the resulted feature matrices $H_1, H_2, \ldots, H_{m^l}$. All features matrices will have same dimensionality which is $t - l(m-1) \times k$ after l times convolution operations with B. This step preserves the temporal correlation between frame features and reduces the dimensionality of the feature sequences but increases the number of the feature matrices (one $t \times k$ feature matrix becomes $m^l t - l(m-1) \times k$ feature matrices). To further reduce the dimensionality of the feature sequences, we use same approach to choose a non-diagonal basis C for $\mathbb{R}^2$ then repeatedly convolve c_1, c_2 with each column of the feature matrices until the dimensionality of all feature sequence is reduced to one as shown in Figure 2.6 (d). By choosing a non-diagonal basis, more temporal correlation could be preserved than using a diagonal basis. For example, given two vectors $p = (1, 2, 3, 4, 5)$ and $q = (1, 2, 4, 4, 5)$, let B and C both be the identity basis for $\mathbb{R}^2$. Follow the DVD generation approach, the result are both $(1, 4, 2, 5)$. However, if we let $B = C((1, 0)^T, (1, 2)^T)$, the results are $(1, 81, 5, 297)$ and $(1, 93, 5, 321)$. By convolving with the basis vectors for R^2, the dimensionality of feature sequences can be guaranteed to reduced to one since each convolution operation will reduce the dimensionality by one. This step letting the DVD can be applied for various length videos. Finally, we concatenate these $1 \times k$-dimensional feature matrices to form the video descriptor, as shown in Figure 2.6 (e). The kt dimensional feature sequence is now represented by $2m^l k$ one dimensional features.

Method	**Average Accuracy**
SIFT-BOW	79.27
SURF-BOW	78.66
VGG-16	76.93
VGG-19	78.8
VC	80
HSV-DVD (Linear SVM)	84.17
HSV-DVD (RBF SVM)	91.77
HSV-DVD (KNN)	94.4

Table 2.4: Near duplicate video detection results on cc web videos dataset. we compare HSV-DVD with baselines and the video cuboid near duplicate video detection system.

2.7 The DVD Generation Approach is Efficient

Assume given a video v which contains t frames, the extracted frame feature histograms have the same length k. Given a randomly chosen basis $B = (b1, b2, ..., bm)$ for $\mathbb{R}^m$. The time complexity of the DVD generation step (c) is $O(\frac{mk(t-m+1)(1-m^l)}{1-m})$. In step (d) we choose a basis from $\mathbb{R}^2$, the time complexity is $O(m^l k(t-lm+l)^2)$. Therefore, the time complexity of generating DVD on top of frame features is $O(\frac{mk(t+m)(1-m^l)}{1-m} + m^l k(t-lm+l)^2)$.

2.8 Experiment Results

In this section, we evaluate the DVD under the scenarios of Near Duplicate Video Detection and Human Action Recognition. All of our experiments are implemented on a machine with Intel Core i7 quad-core processor with 16 GB memory.

2.8.1 Near-duplicate Video Detection

1) Dataset: We evaluate the DVD on CC WEB VIDEO dataset (Wu *et al.*, 2007) which is widely used for the Near Duplicate Video Detection task. The dataset is comprised of 24 independent groups and totally contains 12790 videos. In each group, one video is designated as the seed and others are compared with it and labeled accordingly. There are 7 labels for each group: Exactly Duplicate, Similar, Different Version, Major Change, Long Version, Dissimilar, and Do not Exist. In the experiments, we are not considering the Do not Exist label since it can be simply classified by checking the existence of the video.

2) Experiments details: We implemented a parallel Near Duplicate Video Detection system which adopts the architecture in (Chen *et al.*, 2016). We extract the HSV-color distribution as the frame features and represent them as 256-dimensional histograms. We consider each axis of the HSV-color independently. The non-diagonal basis $B = (b1, b2)$ is chosen for $\mathbb{R}^2$ and set $l = 1$ since the experiment gives the highest detection rate. Therefore, the DVD of each video are three 512-dimensional vectors and each of them preserves the video temporal information of one axis of the HSV-color distribution. We call the DVD that is built on top of the HSV-color distribution the HSV-DVD. We use three learners, one for each video descriptor, to classify the video. For each learner, linear SVM, SVM (RBF kernel) and KNN classifiers are used to classify the videos with 10 fold cross-validation. We take the predicted results from each learner with maximum vote strategy to generate the final predictions.

3) Baselines: We implement two Near Duplicate Video Detection systems. The first one uses BoW approach to aggregate the SIFT (Lowe, 2004) or SURF (Bay *et al.*, 2006) features of frames. Another one fine-tunes the ImageNet (Bay *et al.*,

2006) pre-trained 16-layer and 19-layer VGGNets (Simonyan and Zisserman, 2014) for the Near Duplicate Video Detection task. We average the frame features as the video descriptor then make the final prediction.

4) Results: In the experiments, we compare the average detection accuracy of HSV-DVD to the baselines and the Near Duplicate Video Detection system proposed in (Zhou and Chen, 2010). The average detection accuracy for the entire dataset is calculated by averaging the sum of the detection accuracies for each group. The results are shown in Table 2.4. The HSV-DVD with KNN achieves 94.4% average detection accuracy which is 15.13% and 15.74% better than SIFT-BoW and SURFBoW, respectively. Compare to the CNN based features, HSV-DVD with KNN classifier outperforms the VGGNet-16 and VGGNet-19 by 17.74% and 15.6%, respectively even the feature representations generated by these networks are more effective in vision tasks than the simpler descriptions such as the color histogram. When compared with the Video Cuboid Based Detection System (VC), a streaming Near Duplicate Video Detection system based on the video cuboid feature, the HSV-DVD with KNN classifier performances 14.4% better. Note that the VC uses Locality Sensitive Hashing (LSH) to retrieve the results thus some accuracies may be sacrificed for the time efficiency. However, the HSV-DVD is generated by convolving columns of feature matrices with the basis vector from $\mathbb{R}^2$ (only two 2-dimensional vectors). In addition, even with the simplest linear classifier, the HSV-DVD still outperforms the VC by 4.17%.

Method	Average Accuracy
3D-SIFT	82.6
STIPs+3D-SIFT	92.47
NGLDC	100
STIPs+3DSIFT-DVD	97.84

Table 2.5: Action recognition results on Weizmann dataset. we compare 3DSIFT-DVD with baselines and NGLDC.

2.8.2 Human Action Recognition

1) Dataset: We use the Weizmann Human Action Recognition dataset (Blank *et al.*, 2005) to evaluate the performance of the DVD. The Weizmann actions dataset consists of ten different types of actions classes. Each action class is performed at least once by 9 subjects resulting in 93 video sequences in total. The background is homogeneous and static. We didn't choose larger datasets (e.g. UCF101 (Soomro *et al.*, 2012), HMDB51 (Kuehne *et al.*, 2011)) since our machine does not have enough computational power to generate enough interest point feature for large amounts of videos, which will be discussed in the next subsection.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0	.0
jack	.0	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0
jump	.0	.0	**.67**	.0	.11	.11	.11	.0	.0	.0
pjump	.0	.0	.0	**1.0**	.0	.0	.0	.0	.0	.0
run	.0	.0	.1	.0	**.8**	.0	.1	.0	.0	.0
side	.0	.0	.0	.0	.0	**1.0**	.0	.0	.0	.0
skip	.0	.0	.2	.0	.3	.0	**.5**	.0	.0	.0
walk	.0	.0	.0	.0	.11	.0	.0	**.89**	.0	.0
wave1	.0	.0	.0	.0	.0	.0	.0	.0	**.78**	.22
wave2	.0	.0	.0	.0	.0	.0	.0	.0	.22	**.78**

Table 2.6: The confusion matrix of the 3D-SIFT approach.

2) Experiments details: Given a video v, we first detect the STIPs which are

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0	.0
jack	.0	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0
jump	.0	.0	**1.0**	.0	.0	.0	.0	.0	.0	.0
pjump	.0	.0	.11	**.89**	.0	.0	.0	.0	.0	.0
run	.0	.0	.0	.0	**1.0**	.0	.0	.0	.0	.0
side	.0	.0	.0	.0	.0	**1.0**	.0	.0	.0	.0
skip	.0	.0	.0	.0	.1	.0	**.9**	.0	.0	.0
walk	.0	.0	.1	.0	.0	.0	.0	**.9**	.0	.0
wave1	.0	.0	.0	.0	.0	.0	.0	.0	**.89**	.11
wave2	.0	.0	.0	.0	.0	.0	.0	.0	.33	**.67**

Table 2.7: The confusion matrix of the STIPs + 3D-SIFT approach.

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0	.0
jack	.0	**1.0**	.0	.0	.0	.0	.0	.0	.0	.0
jump	.0	.0	**1.0**	.0	.0	.0	.0	.0	.0	.0
pjump	.0	.0	.0	**1.0**	.0	.0	.0	.0	.0	.0
run	.0	.0	.0	.0	**.8**	.0	.2	.0	.0	.0
side	.0	.0	.0	.0	.0	**1.0**	.0	.0	.0	.0
skip	.0	.0	.0	.0	.0	.0	**1.0**	.0	.0	.0
walk	.0	.0	.0	.0	.0	.0	.0	**1.0**	.0	.0
wave1	.0	.0	.0	.0	.0	.0	.0	.0	**1.0**	.0
wave2	.0	.0	.0	.0	.0	.0	.0	.0	.0	**1.0**

Table 2.8: The confusion matrix of the STIPs + 3DSIFT-DVD approach.

obtained from Dollar's detector (Dollár *et al.*, 2005). We extracted enough STIPs by executing the detector multiple times with different blur parameters and different Gabor filter lengths. Then, we calculate the 3D-SIFT descriptors (Scovanner *et al.*, 2007) for all detected interest points. The dimensionality for each 3DSIFT descriptor is 640. We sort the interest point feature histograms by the time (the spatial information between the interest point feature histograms are ignored). Similar to the experiments for Near Duplicate Video Detection, we choose B for $\mathbb{R}^2$

and set $l = 1$. We omit the step (d) since the video descriptor for the Human Action Recognition should concentrate on the action pattern preserving, which means the temporal information between the adjacent local regions (or the interest points) should be preserved. Assume c interest points are obtained for a video, we will get two $(c-1) \times 640$-dimensional video descriptors. We call these descriptors 3DSIFT-DVD. We then apply the BoW approach to aggregate the video descriptors, construct two co-occurrence matrices for the vocabularies and populate them using frequency histograms with threshold = 0.8 (Scovanner *et al.*, 2007). Finally, we concatenate the feature grouping histograms together as the input for the classification. The dimensionality is $x^2 - x$. A linear SVM is trained for the recognition and use leave-one-out cross-validation.

3) Baselines: We implement a action recognition system based on the STIPs and 3D-SIFT. The STIPs are shared with the baseline and the DVD.

4) Results: The average recognition accuracy are shown in Table 2.5. The results are reported over 10 runs. The first line of Table 2.5 is the average recognition accuracy by using original 3D-SIFT method (Scovanner *et al.*, 2007). The recognition rate is 82.6%. The confusion matrix Table 2.6 shows some 3D-SIFT features are ambiguous to each other (jump is ambiguous with run, side and skip, wave by one hand is ambiguous with wave by two hands). When STIPs are used instead of randomly chosen interest points, the recognition rate achieves 92.47% (the second line of Table 2.5). However, some ambiguities still exist as shown in Table 2.7. By adopting the 3DSIFT-DVD on the same STIPs which are used by the 3D-SIFT, the recognition rate raises to 97.84% (last line of Table 2.5). All actions are perfectly recognized expect the run action (Table 2.8). Reference (Sun and Liu, 2012) reported

	HSV-DVD	3DSIFT-DVD
Frame Feature Generation	0.897 s	566.4 s
Convolution	0.128 s	0.04 s
Total Time Cost	1.025 s	566.44 s
Portion	12.48%	0.007%

Table 2.9: The average generation time of the DVD for one video.

the perfectly recognition rate by adopting the NGLDC (the third line of Table 2.5). However, to calculate NGLDC, the numbers of each visual word in every frame are needed. This increases the computation time.

2.8.3 Running Time Analysis

To evaluate the efficiency of the DVD. We randomly select 10% videos (average 11 frames for a video) in the CC WEB VIDEOS to evaluate the HSV-DVD generation time and use whole Weizmann Human Action Recognition dataset (200 interest points for each video) to evaluate the 3DSIFT-DVD generation time. As shown in Table 2.9, we only need a small period of time to generate DVD on top of the frame features (or interest point descriptors). For the HSV-DVD, the average generation time for one video is 1.025 seconds, it takes 0.897 seconds to extract the HSV-color distribution and only 0.128 seconds to generate the HSV-DVD for one video. The convolution part occupies 12.48% of the total HSV-DVD generation time. For the 3DSIFT-DVD, it takes 566.4 seconds to extract the 3D-SIFT features for 200 points (use the code provided by the authors) and only 0.04 seconds are needed to generate 3DSIFT-DVD. The convolving part only occupies 0.007% of the total time.

2.9 Conclusion

In this paper, we have designed a video descriptor called DVD which is generally built on top of different frame features. We have theoretically and experimentally demonstrated DVD generation is efficient. We have shown that the high detection (recognition) rate is gained by applying DVD with even a linear classifier for the Near Duplicate Video Detection and Human Action Recognition tasks.

2.10 Appendix

Algorithm 1: MatrixConvolve

```
Inputs: Frame feature matrix $H$, vector $b$;
**for** *column vector* $f$ *in* $H$ **do**
    $f \leftarrow f * b_j$;
**end**
$H \leftarrow (f_1, f_2, \dots, f_k)$;
**return** $H$;
```

Algorithm 2: Discriminative Video Descriptor

```
Inputs: Frame feature matrix $H \in \mathbb{R}^{t \times k}$, basis $B \in \mathbb{R}^m, C \in \mathbb{R}^2$;
Add $H$ in set $S$;
$i \leftarrow 1$;
while $i \leq l$ do
    for $j$ form 1 to $m$ do
        for every $H = (f_1, f_2, \ldots, f_k)$ in $S$ do
            $H_{tmp} \leftarrow MatrixConvolve(H, b_j)$;
            $S$ append $H_{tmp}$;
        end
    end
    for $q$ from 1 to $m^i$ do
        remove $S[q]$ from $S$
    end
    $i \leftarrow i + 1$;
end
$i \leftarrow 1$;
while $i \leq t - l(m-1) - 1$ do
    for $j$ form 1 to 2 do
        for every $H = (f_1, f_2, \ldots, f_k)$ in $S$ do
            $H \leftarrow MatrixConvolve(H, c_j)$;
        end
    end
    $i \leftarrow i + 1$;
end
return concat(S)
```

Bibliography

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417.

Blank, M., Gorelick, L., Shechtman, E., Irani, M., and Basri, R. (2005). Actions as space-time shapes. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1395–1402. IEEE.

Chen, Y., He, W., Hua, Y., and Wang, W. (2016). Compoundeyes: Near-duplicate detection in large scale online video systems in the cloud. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE.

Dollár, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE.

Jain, A., Tompson, J., LeCun, Y., and Bregler, C. (2014). Modeep: A deep learning framework using motion features for human pose estimation. In *Asian Conference on Computer Vision*, pages 302–315. Springer.

Klaser, A., Marszałek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association.

Laptev, I. (2005). On space-time interest points. *International journal of computer vision*, **64**(2-3), 107–123.

Le, Q. V., Zou, W. Y., Yeung, S. Y., and Ng, A. Y. (2011). Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3361–3368. IEEE.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, **60**(2), 91–110.

Over, P., Awad, G., Michel, M., Fiscus, J., Sanders, G., Shaw, B., Kraaij, W., Smeaton, A. F., and Quéot, G. (2013). Trecvid 2012-an overview of the goals, tasks, data, evaluation mechanisms and metrics.

Perronnin, F., Sánchez, J., and Mensink, T. (2010). Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer.

Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, Q. and Liu, H. (2012). Action disambiguation analysis using normalized google-like distance correlogram. In *Asian Conference on Computer Vision*, pages 425–437. Springer.

Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. In *CVPR 2011*, pages 3169–3176. IEEE.

Wu, X., Hauptmann, A. G., and Ngo, C.-W. (2007). Practical elimination of near-duplicates from web video search. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 218–227. ACM.

Zhou, X. and Chen, L. (2010). Monitoring near duplicates over video streams. In *Proceedings of the 18th ACM international conference on multimedia*, pages 521–530. ACM.

Chapter 3

Few-Shot Learning of Video Action Recognition Only Based on Video Contents

3.1 Citation and Main Contributor

Bo, Yang, Lu, Yangdi and He, Wenbo. "Few-Shot Learning of Video Action Recognition Only Based on Video Contents," 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, CO, USA, 2020, pp. 584-593, doi: 10.1109/WACV45572.2020.9093481.
The main contributor to this paper is the first author - Yang Bo (contributes more than 80%).

3.2 Copyright

3.3 Abstract

The success of video action recognition based on Deep Neural Networks (DNNs) is highly dependent on a large number of manually labeled videos. In this paper, we introduce a supervised learning approach to recognize video actions with very few training videos. Specifically, we propose Temporal Attention Vectors (TAVs) which adapt various length videos to preserve the temporal information of the entire video. We evaluate the TAVs on UCF101 and HMDB51. Without training any deep 3D or 2D frame feature extractors on video datasets (only pre-trained on ImageNet), the TAVs only introduce 2.1M parameters but outperforms the state-of-the-art video action recognition benchmarks with very few labeled training videos (*e.g.* 92% on UCF101 and 59% on HMDB51, with 10 and 8 training videos per class, respectively). Furthermore, our approach can still achieve competitive results on full datasets (97.1% on UCF101 and 77% on HMDB51).

3.4 Introduction

The use of Deep Neural Networks (DNNs) in the field of computer vision has expanded significantly in recent years. For video action recognition, several frameworks

(Feichtenhofer *et al.*, 2016b, 2017; Tran *et al.*, 2018) have shown outstanding performance. The success of these approaches is largely sustained by the manual annotation of the large-scale datasets. However, it is still challenging to recognize human actions with very few manually labeled training videos. There are several attempts to deal with this problem. Srivastava *et al.* (Srivastava *et al.*, 2015) trained a Long Short-term Memory (LSTM) with the fixed number of unlabeled video frames to predict future frames of that video then fine-tune it for the supervised video action recognition. However, it still needs a large number of videos to train which brings huge computational overhead. Zhu *et al.* (Zhu *et al.*, 2018), Mettes *et al.* (Mettes and Snoek, 2017) and Jain *et al.* (Jain *et al.*, 2015) attempted to recognize actions without any observed data or with only few labeled data (Zero/Few-Shot Learning). These approaches classify the actions by measuring the similarity (*e.g.* Euclidean distance) between the visual representations and the semantic representations in the embedding space. However, these approaches require additional linguistic contexts or visual representations of objects and the accuracy is heavily dependent on the precise representation of the additional information.

In this paper, we aim to adopt supervised learning to recognize human actions with very few manually annotated training videos. Recent works such as (Fernando *et al.*, 2015; Varol *et al.*, 2018; Wang *et al.*, 2016) pointed out that long term dynamics and temporal patterns are very important cues for the recognition of actions. The key challenge is to generate a video descriptor that precisely captures the important video-wide temporal correlation among frame features with a small number of training parameters. Some current Convolutional Neural Networks (CNNs) based researches are either using sub-sampling (Wang *et al.*, 2016) or longer video clips (Varol *et al.*,

2018) to capture the video-wide temporal dynamics. However, these works only preserve partial temporal information of the video. Some other approaches adapt ranking functions (Fernando *et al.*, 2015) or Recurrent Neural Networks (RNNs) (Ballas *et al.*, 2015; Donahue *et al.*, 2015) to preserve the temporal information. However, they introduce too many training parameters that cannot be trained well in the circumstance of only having very few training videos. Another work (Girdhar *et al.*, 2017) generates the video descriptor by clustering the pixels in all frame features maps both spatially and temporally with k-means clustering algorithm. However, this approach cannot preserve the temporal information of videos.

In this paper, we propose Temporal Attention Vectors (TAVs) which adapt to various lengths of videos to encode the correlation among frame features. The frame features are either manually defined or generated by an ImageNet pre-trained CNN. The initial value of TAVs are manually defined and each TAV highlights a certain period of the video by giving them higher temporal weights than others. After that, the TAVs are aggregated by their importance scores which are learned by a shallow CNN. There are three keys advantages to use TAVs. First, the TAVs capture the video-wide dynamics of the video. Unlike the current end-to-end trainable frameworks, the TAVs encode the temporal correlation between all frames, which is important especially with only very few training videos. Second, the TAVs can model complex temporal patterns. The static temporal weights cannot correctly express the temporal patterns of actions. For example, actions "run" and "long jump" have different patterns, we only need to focus on "run" action for the prior one but need focus on both "run" and "jump" for the latter one. By studying the importance scores, the TAVs can simulate different temporal patterns. Third, the TAVs only introduce few training parameters. With insufficient training samples, a deep neural network (many training parameters) is hard to learn the true temporal patterns of actions. Only few training parameters (importance scores) makes TAVs could capture more accurate temporal pattern than the deep network with very few training videos.

3.5 Related Works

Capturing the spatiotemporal information of videos for action recognition has been a well-studied research domain. Historically, researchers have mostly focused on the handcrafted spatiotemporal features of Space-Time Interest Points (STIP) (Laptev, 2005). Most successful examples are 3D Histogram of Gradient (HOG3D) (Klaser *et al.*, 2008), Histogram of Optical Flow (HOF) (Laptev *et al.*, 2008), and Motion Boundary Histogram (MBH) (Dalal *et al.*, 2006). Also, the trajectory-based approaches (Jiang *et al.*, 2012; Matikainen *et al.*, 2009; Sun *et al.*, 2009; Wang *et al.*, 2011; Wang and Schmid, 2013) have shown a significant improvement in action recognition.

More recently, the CNN-based end-to-end fashions have been widely applied to the video action recognition area and shown the outstanding performance. These approaches can be roughly separated into two categories. The first category, which extends the CNNs to a third, temporal dimension by replacing the 2D filters with 3D ones (Carreira and Zisserman, 2017; Ji *et al.*, 2013; Tran *et al.*, 2015, 2018; Varol *et al.*, 2018) to capture the spatiotemporal information from fixed length video clips. The second category initially processes color and optical flow information in parallel for subsequent late fusion of their separate classification scores (Simonyan and Zisserman, 2014). Several improvements were proposed based on this work. For example, Wang *et al.* (Wang *et al.*, 2015) extracted deep features and conducted trajectory constrained pooling to aggregate convolutional features as video representations. Feichtenhofer *et al.* (Feichtenhofer *et al.*, 2017) tried different two-stream fusion approaches to fuse the two streams. Carreira and Zisserman (Carreira and Zisserman, 2017) recently introduced a model (I3D) that combines two-stream processing with 3D convolutions.

An alternative solution models the temporal structure of video by various pooling approaches (Girdhar and Ramanan, 2017; Kar *et al.*, 2017; Yue-Hei Ng *et al.*, 2015), rank functions (Fernando *et al.*, 2015), k-means clustering (Girdhar *et al.*, 2017) and different distributions (Piergiovanni *et al.*, 2017; Piergiovanni and Ryoo, 2018a,b). Recurrent Neural Networks have also been used to encode temporal information for learning video representations (Donahue *et al.*, 2015; Srivastava *et al.*, 2015; Sun *et al.*, 2015a,b; Wu *et al.*, 2015). Donahue *et al.* (Donahue *et al.*, 2015) used the LSTM together with CNN to either output an action label or a video description. Srivastava *et al.* (Srivastava *et al.*, 2015) proposed to learn video descriptions with the encoder-decoder LSTM in an unsupervised manner.

Our work is similar to the temporal structure filters in (Piergiovanni and Ryoo, 2018a). They train Cauchy distributions to detect multi-actions in videos. The centers and width of Cauchy distributions are initialized by some trainable and uniformly selected "seed" between -1 to 1 then scaled according to the length of videos and exponential function, respectively. On the contrary, our approach initializes both center and width according to the length of video and studies the importance scores instead of learning the distribution parameters themselves. We compare temporal structure filters with TAVs in Section 3.8.

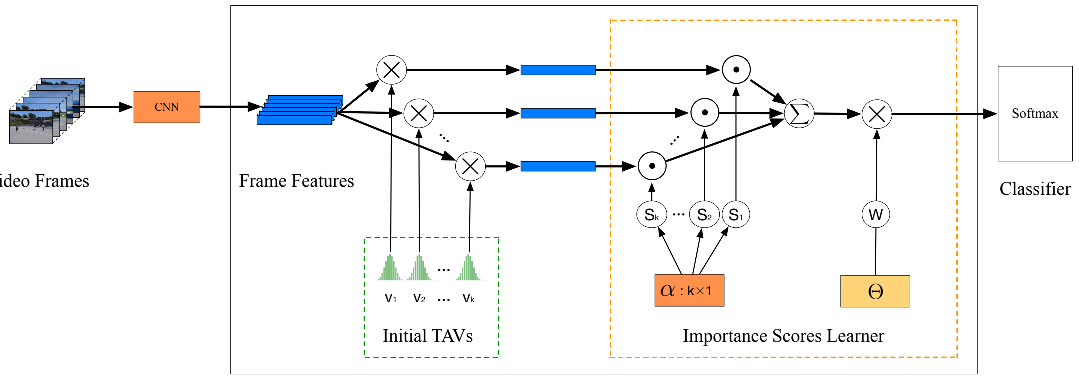


Figure 3.2: The working procedure of TAVs. The CNN which is used as the frame feature extractor pre-trained on ImageNet and the weights are frozen during the task, *e.g.* not trained on the video dataset. "$\otimes$" denotes matrix multiplication, "$\odot$" denotes the element-wise multiplication and "Σ" denotes the addition. v_k denotes the initialized TAVs, s_k denotes the importance score for the kth TAV and W denotes the linear embedding weights matrix. α denotes $k \times 1$ convolution and Θ denotes the linear embedding operation. Here, we show the base importance learner, the inflate-shrink learner can be done by adding one convolution layer with 32 1×1 filters. We use a fully connected layer with softmax function as the classifier.

3.6 Temporal Attention Vectors

To recognize actions with very few training samples, the video representation should preserve video-wide temporal information meanwhile the generation procedure of the representation should involve only a small number of training parameters.

We now describe the TAVs which are shown in Figure 3.2. We denote a video as $X = (x_1, x_2, \ldots, x_T)$, where $x_t \in \mathbb{R}^{W \times H \times C}$ and $t = [1 : T]$. We use W and H to represent the width and height of the frame respectively. Each frame either represents an RGB image ($C = 3$) or a horizontal / vertical optical flow image ($C = 1$). The length T usually varies for different videos. We use $\phi(x_t) \in \mathbb{R}^D$ to represent the feature of frame x_t, where ϕ represents the operations of a CNN (the frame feature extractor). The video representation is generated with two steps: i). The frame features $(\phi(x_1), \phi(x_2), \ldots, \phi(x_T))$ are aggregated with the elements of K TAVs $\{v_1, v_2, \ldots v_K\}$, where $v_k \in \mathbb{R}^T$ and initial values are manually defined. ii). All aggregated features are added with the importance scores s for each TAV then linear embedded with weight W to form the video representation. s and W are learned by a convolution layer and a fully connected layer, respectively. More formally, we rewrite the above operations as follows:

$$\mathcal{D} = W \otimes \sum_{k=1}^{K} \sum_{t=1}^{T} s_k \phi(x_t) v_{t,k}, \tag{3.6.1}$$

where $v_{t,k}$ is the tth element in the kth TAV v_k, s_k is the importance score for the kth TAV and $\otimes$ denotes the matrix multiplication.

The TAVs are able to study the temporal pattern with very few training videos because of the follows. i). It encodes video-wide temporal information by using multiple TAVs which adapt to the various lengths of frame sequences and does not introduce any training parameters. ii). It is able to select the important temporal information by studying the importance scores. In the following of this section, we provide the details of how to initialize the TAVs and the architecture of the shallow CNN that is used to study the importance scores of the TAVs.

3.6.1 Initialization of the TAVs

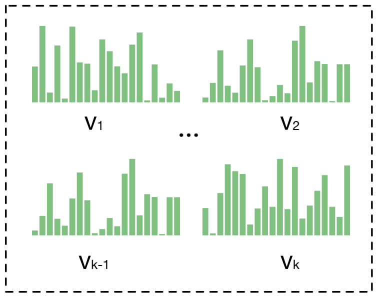
The static pooling approaches (*e.g.* average pooling) fail to preserve the sequential information of the frame features. We propose to encode the video-wide temporal information by calculating the weighted sum of all frame features. The weights are the elements of the TAVs which adapt to various length of videos. We introduce several initialization approaches of the TAVs below and leave the experiments in appendix.

Random. One simplest way to initialize the TAVs is uniformly choosing them from the interval $(0, 1)$ (Figure 3.3a). More formally, let L denotes a number larger or equal to the frames number of the longest video in a dataset, we can choose the temporal attention weights as follows:

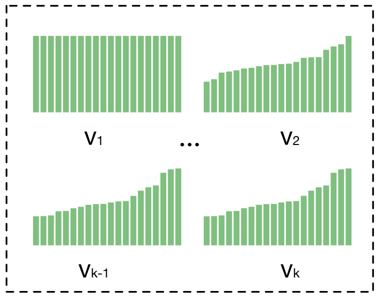
$$\begin{aligned}
&a_k = [a_{1,k}, a_{2,k}, \dots, a_{L,k}],\ k = [1 : K],\\
&a_{l,k} \sim \mathcal{U}(0, 1),\ l = [1 : L],\\
&v_k = [v_{1,k}, v_{2,k}, \dots, v_{T,k}],\\
&v_{t,k} = \frac{e^{a_{t,k}}}{\sum_{t=1}^{T} e^{a_{t,k}}}.
\end{aligned} \tag{3.6.2}$$

We normalize the weights in v_k such that let they sum up to 1 to eliminate the impact brought by the various length of videos. For example, as the video length increases, the weights for each frame feature become smaller. Therefore, the same action represented with the various number of frames will not be misclassified.

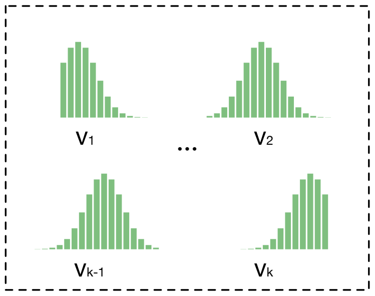
Single Switching Distinguishable (SSD). An alternative way is using the combination of a constant sequence and the strictly monotone sequences as the initial values of the TAVs, which are shown in Figure 3.3b, to distinguish a single pair of frames switching (*e.g.* $X_1 = (x_1, x_2, x_3)$ and $X_2 = (x_1, x_3, x_2)$. The elements in the constant sequence are all c and the elements in strictly monotone sequences are randomly chosen from $\mathcal{U}(0, 1)$ then sort with ascending or descending orders. Same as the random initialization, we normalize the weights in v_k. The approach can be



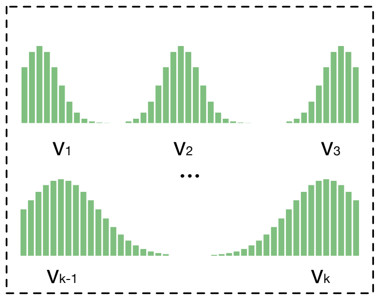
(a) Random



(b) SSD



(c) DG



(d) SLDG

Figure 3.3: The illustration of TAVs which are initialized by different approaches.

formally represented as follows:

$$\begin{aligned}
&v_1 = [c, c, \dots, c],\\
&a_k = [a_{1,k}, a_{2,k}, \dots, a_{L,k}],\ k = [2:K],\\
&a_{l,k} \sim \mathcal{U}(0,1),\ l = [1:L],\\
&sort(a_k),\\
&v_k = [v_{1,k}, v_{2,k}, \dots, v_{T,k}],\\
&v_{t,k} = \frac{e^{a_{t,k}}}{\sum_{t=1}^{T} e^{a_{t,k}}}.
\end{aligned} \tag{3.6.3}$$

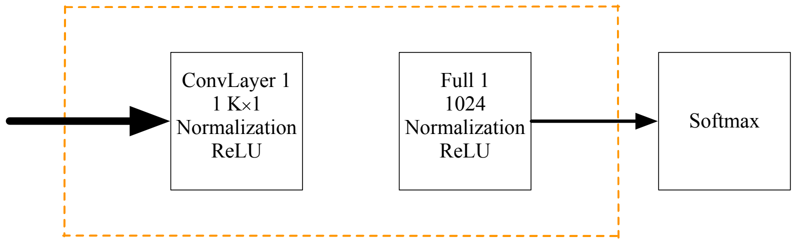
By using these TAVs, it is obvious that the switching order of single pair of frames could be detected by the TAVs based on a reasonable assumption that the frame features of two different frames are different.

Dynamic Gaussian (DG). This TAVs initialization approach is based on the following observations: i). The "key" frames (the most important frames for classification task) that are used to represent an action are consecutive. ii). These key frames form only a clip rather than the entire video. iii). The number of key frames that are needed to represent different actions varies. Based on the above observations, we propose a way to initialize the TAVs from the Probability Density Function (PDF) of Gaussian Distributions. The results are shown in Figure 3.3c. We evenly divide a video into several clips and use the TAVs highlight each of them. The TAVs give the higher temporal weights for the frame features in the clip and lower weights for the frame features out of that clip. The mean and standard deviation are dynamically selected based on the length of the video clips. Specifically, we evenly partition the frame number sequence $(1, 2, \dots, T)$ of the video X into K chunks $\{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_K\}$. For each chunk, we choose the element in the middle position as the mean μ_k and use the length of the chunk $len(\mathcal{C}_k)$ divide by a factor θ as the standard deviation α_k. More formally, these operations can be formulated as:

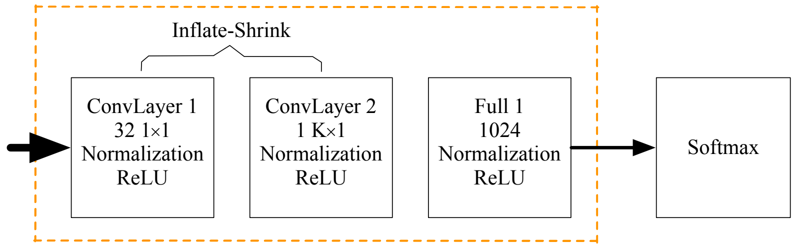
$$\begin{aligned} v_k &= [v_{1,k}, v_{2,k}, \dots, v_{T,k}],\ k = [1:K], \\ v_{t,k} &= PDF(t|\mu_k, \alpha_k^2),\ t = [1:T], \\ \mu_k &= \mathcal{C}_k(\frac{len(\mathcal{C}_k)}{2}),\ \alpha_k = \frac{len(\mathcal{C}_k)}{\theta}. \end{aligned} \tag{3.6.4}$$

The DG initialized TAVs highlight different clips cross the entire video, but each of them encodes the video-wide temporal information.

Short-Long Dynamic Gaussian (SLDG). This initialization approach is inspired by the DG approach. The original DG can only highlight certain range of temporal information of a video. However, we are able to highlight shorter or longer



(a) The base learner.



(b) The inflate-shrink learner.

Figure 3.4: The architecture of the importance score learner.

ranges of temporal information by decreasing or increasing the standard deviation value in the DG generation approach, respectively (Fig 3.3d). More formally, we define a set of chunk numbers $m = \{m_1, m_2, \ldots, m_N\}$. Then we repeat the DG vector generation approach N times and use m_n as the chunk number in the nth iteration. Thus, the total number of temporal vectors is $m_1 + m_2, + \ldots m_N$ and the value of standard deviation is dynamically defined by m_n.

3.6.2 Importance Score Learner

The importance score learner is used to generate video descriptor which contains important temporal information based on the aggregated frame features. We propose two types of learner. i). Base learner: The importance scores of TAVs are directly learned by a single convolution layer (Figure 3.4a). ii). Inflate-shrink learner: The dimensionality of aggregated frame features is first expanded using 1×1 convolution then compressed into a single one as shown in Figure 3.4b. The base learner has a single convolutional layer with one $K \times 1$ filter and the inflate-shrink learner has 2 convolutional layers with 32 and 1 filter response maps with 1×1 and $K \times 1$ filters for the first and second convolutional layers, respectively. All convolutional layers are followed by a batch normalization layer and a rectified linear unit (ReLU). Filter stride for all dimensions is 1 for convolution operations. Then we apply 1 fully connected layers of sizes 1024 for both learners. We use ReLU after the fully connected layers.

3.7 Implementation Details

The architecture of entire framework is shown in Figure 3.2. We adopt all convolution blocks and the global pooling layer of the ImageNet (Krizhevsky *et al.*, 2012) pre-trained ResNet-152 (He *et al.*, 2016) and I3D (Carreira and Zisserman, 2017) as the backbone networks (frame feature extractors). The weights for the extractors are shared and frozen for all experiments. The final video descriptor is predicted using a fully connected layer with softmax function.

3.7.1 Input Configuration

We use the same input configurations for both training and testing in our experiments. The frame extractor generates the features for both RGB frames and the optical flow images. The optical flow images (Zach *et al.*, 2007) are pre-computed and stored as JPEG images (with displacement vectors > 20 pixels clipped). We follow the data argumentation from (Wang *et al.*, 2016). We randomly crop from the four corners and the center of input images and sample the width and height of each crop randomly as $W, H \in \{256, 224, 192, 168\}$, followed by re-sizing to 224×224. The argumentation is applied for both original and horizontal flipped images.

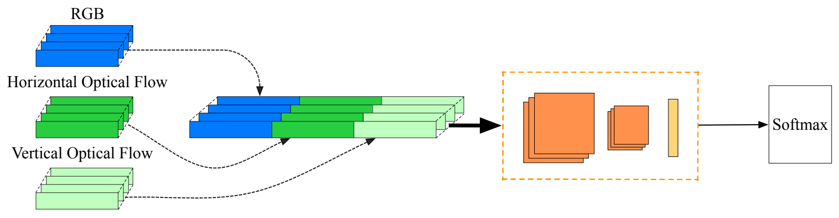
3.7.2 Two Streams Fusion

Since our framework adopts the two-stream architecture which takes RGB and optical flow fields as inputs. We consider three ways to fuse the spatial and temporal streams. The detailed comparison is shown in appendix.

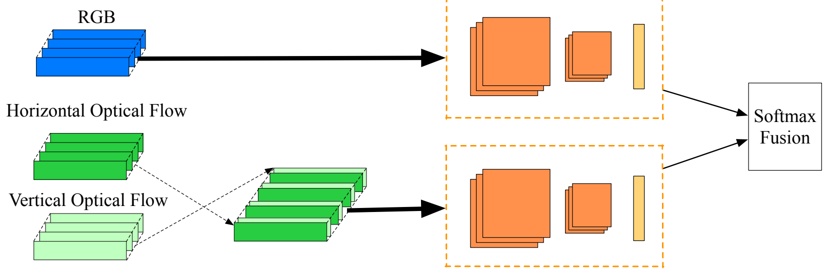
Early concatenation. As shown in Figure 3.5a, the importance scores of both spatial and temporal TAVs are studied by a single learner. In this case, the input of the learner $\mathcal{F} \in \mathbb{R}^{K \times 3D}$.

Late fusion. In contrast with the early concatenation approach, we study the importance scores of spatial and temporal TAVs separately by two learners. The input for the temporal learner is $F \in \mathbb{R}^{K \times D \times 2}$. The class scores of the spatial and temporal streams are combined by late fusion as the final class scores as illustrated in Figure 3.5b.

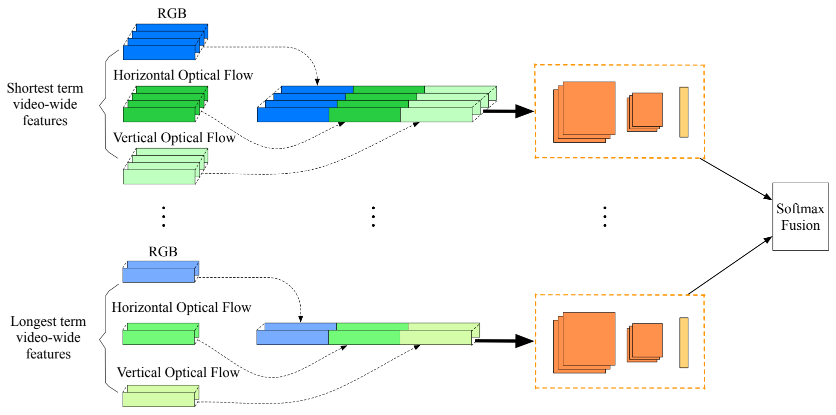
Short-Long term fusion. This fusion approach is designed for the SLDG initialized TAVs. As shown in Figure 3.5c, this approach learn the importance score of TAVs for different temporal periods separately by adopting multiple learners. The spatial and temporal TAVs share same importance score (same as the early concatenation). The class scores of the different terms are then late fused as the final class score.



(a) Early concatenation.



(b) Late Fusion.



(c) Short-Long term Fusion.

Figure 3.5: Three spatial and temporal fusion approaches.

3.7.3 Training and Testing

The training procedures depend on different fusion approaches. For the early concatenation and short-long term fusion, we use the batch size of 64 which are randomly selected (uniform across all video samples). The initial learning rate is set to 10^{-5} and reduce by a factor of 10 after the validation error saturates. The training stopped when the learning rate reaches 10^{-8}. For the late fusion, we first separately train both streams then train them together as in (Simonyan and Zisserman, 2014). The learning rate starts at 10^{-4} and is reduced by a factor of 10 two times after the validation error saturates. When train two streams together, we set the learning rate as 10^{-5} and reduce by a factor of 10 after the validation error increases. We stop the training when learning rate reaches 10^{-8}. The batch size is 128 for single stream training and 64 for two streams together. The kernel and bias weights are all initialized by Xavier initialization (Glorot and Bengio, 2010). The learner weights are learned using the Adam algorithm (Kingma and Ba, 2014). During the testing, the class scores for the whole video are obtained by averaging the scores across the inputs.

3.8 Experiments and Results

In this section, we first introduce the evaluation datasets then we provide detailed analysis of the effectiveness of different TAVs initialization, fusion approaches and the inflate-shrink structure. Finally, we compare the performance of our method to the state-of-the-art on the datasets with change in the size of training videos and the full datasets.

3.8.1 Datasets

We evaluate our approach on two popular action recognition datasets. First, UCF101 (Soomro *et al.*, 2012), which consists of 13320 action videos in 101 categories. The second dataset is HMDB51 (Kuehne *et al.*, 2011), which contains 6766 videos that have been annotated for 51 actions. For both datasets, we use two different sets of training/testing splits: i). the official splits, ii). the smaller splits. The official splits is provided by the datasets. For the smaller splits, we uniformly choose various numbers of videos from the official training splits for each action and the testing splits are the same as the official one.

There are two main reasons that we choose UCF101 and HMDB51: **a.** our objective is action recognition under very few training data scenario. If there are enough data (e.g. Kinetics (Carreira and Zisserman, 2017)), the 3D-CNNs (e.g. I3D, I2+1D networks) are the better choices, **b.** we only find (Srivastava *et al.*, 2015) have done the similar task. For a fair comparison, we choose the same datasets as them.

3.8.2 Evaluation of the Effectiveness of TAVs and Importance Score Learners

In this section, we investigate the effectiveness of TAVs and importance score learners (Section 3.6), the average accuracies are reported in Table 3.7. We choose the DG initialized TAVs ($K = 4$) to encode the frame features and evaluate the performances. The value of θ is set to 2 since we find it results the best performance. In the following experiments, we use $\theta = 2$ as default for DG vectors. We follow the training and testing procedures as described in Section 3.7. In Table 3.7, we can see applying DG initialized TAVs gives 3.7% and 3% accuracy improvements for ResNet-152 and I3D

Aggregator \ Backbone	ResNet-152 (He *et al.*, 2016)	I3D (Carreira and Zisserman, 2017)
Average pooling	64.4	86.1
DG	68.1	89.1
DG + Base learner	70	89.5
DG + Inflate-Shrink learner	70.1	90.6

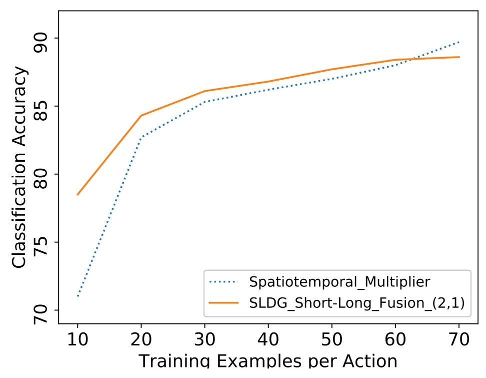
Table 3.7: Average accuracy (%) on the UCF101 split 1 (RGB) with different importance score learners and backbones. The experiments are repeated 10 times and each time 10 training videos are uniformly chosen for each class.

backbones compared to the average pooling, respectively. The accuracy is further increased by applying the score learner. For example, the performance improves from 68.1% to 70% when use the base learner for DG+ResNet152, and further increases 0.1% when apply the inflate-shrink learner. When use I3D as the backbone, the inflate-shrink learner yields 1.5% improvements compared to only apply the pure DG initialized TAVs. In the following experiments, we use the inflate-shrink learner as the default choice.

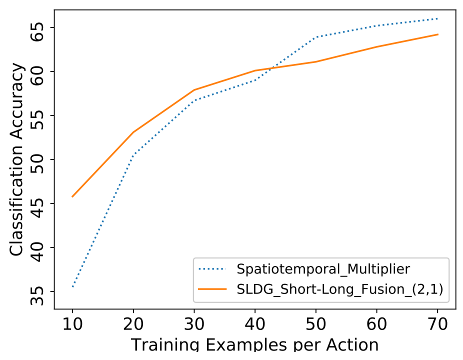
Methods	# Parameter
St Multiplier (two-stream) (Feichtenhofer *et al.*, 2017)	85.8M
LSTM (Srivastava *et al.*, 2015)	83.8M
TAVs (ours)	2.1M

Table 3.8: Parameter number of Spatiotemporal Multiplier Networks, unsupervised LSTM and TAVs

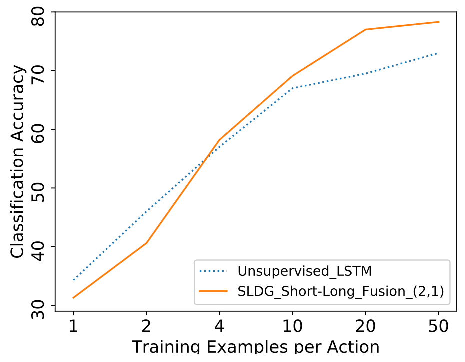
We also evaluate the effectiveness of hyper-parameters (TAVs number, initialization approaches, fusion approaches etc.) with few training parameters. Please see appendix for detail.



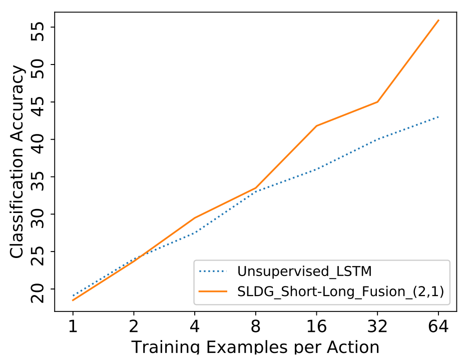
(a) UCF101



(b) HMDB51



(c) UCF101-RGB



(d) HMDB51-RGB

Figure 3.6: Comparisons with the Spatiotemporal Multiplier Network ((a) and (b)) and fine-tuned unsupervised LSTM ((c) and (d)) for action recognition with change in the size of the labeled training set on UCF101 and HMDB51 split 1. The training videos are uniformly chosen for each action class. For the comparison with St Multiplier Network, the experiments are repeated 5 times and the average accuracy (%) are reported for both TAVs and St Multiplier Network. For comparison with the fine-tuned unsupervised LSTM, only RGB frames are used and the average accuracy of 10 times experiments are reported.

3.8.3 Comparison with state-of-the-art with Very Few Training Videos

In this section, we compare our model to other frameworks on UCF101 and HMDB51 with change in the size of the labeled training set. We use the same training sets for

# Training Videos	I3D (Carreira and Zisserman, 2017)		I3D+TSF (Piergiovanni and Ryoo, 2018a)		I3D+SLDG(2,1)	
	Round Acc	Average Acc	Round Acc	Average Acc	Round Acc	Average Acc
UCF101						
1	20.3 / 24.4 / 25.6	23.4	45.8 / 52.6 / 46.7	48.4	65.8 / 68.2 / 71	68.3
2	39.4 / 39.8 / 39.6	39.6	61.7 / 63.2 / 61.5	62.1	82.4 / 83.4 / 83.3	83
4	63.3 / 63.1 / 62.9	63.1	74.1 / 69.2 / 65.2	70	88.4 / 88.4 / 88.1	88.3
10	74.4 / 74.9 / 74.4	74.6	84.7 / 85.8 / 83	84.5	91.8 / 92.5 / 91.7	92
20	83.3 / 83 / 83.4	83.2	88.6 / 85.5 / 86.4	86.3	93.3 / 93.2 / 93.4	93.3
50	90.5 / 90.2 / 90.4	90.4	90.7 / 90.4 / 87.8	89.6	94.1 / 93.8 / 94	94
HMDB51						
1	13.1 / 13.8 / 14.2	13.7	28.3 / 25.4 / 27.1	27	34.7 / 35.8 / 37.4	36
2	18.6 / 18.2 / 18.3	18.4	36.5 / 34.1 / 34.8	35.1	44.4 / 42.5 / 43.3	43.4
4	30 / 31.2 / 29.7	30.3	46.1 / 44.5 / 42.6	44.4	53.3 / 56.3 / 49.8	53.1
8	42.9 / 44 / 43.3	43.4	49.5 / 52 / 49.5	50.3	56.5 / 61.1 / 59.6	59
16	53.2 / 54.1 / 53.9	53.7	52.9 / 50.2 / 52.9	52	62.1 / 63.9 / 63.5	63.2
32	61.1 / 61.8 / 61	61.3	56 / 55.2 / 56.2	55.8	65.2 / 66.3 / 65	65.5
64	68.6 / 68.8 / 68.2	68.5	56.9 / 58.2 / 59	58	67.4 / 67.6 / 67.1	67.4

Table 3.9: Comparisons with I3D (left column) and Temporal Structure Filter (TSF) (middle column) for action recognition with different number of training samples on UCF101 and HMDB51 split 1 (RGB). The experiments are repeated 3 times and each time the training videos are uniformly chosen for each class. We report both round accuracies (left part of each column) and average accuracy (right part of each column).

all models. During the comparison of the TAVs initialization approaches, we find the SLDG initialization with $K = 3$ (# of TAVs) and $m = (2, 1)$ (# of chunks for each iteration) gives best results. We believe this is because the SLDG catch both short and long temporal information of video. In all following experiments, we use SLDG with this setting as default choice.

The first model is the Spatiotemporal Multiplier network (Feichtenhofer *et al.*, 2017). The comparisons are shown in Figure 3.6a and Figure 3.6b. Our model outperforms the Spatiotemporal Multiplier network when only giving few labeled training videos. For example, with only 10 labeled training video per class, our model achieves 78.5% on UCF101 and 45.8% on HMDB51 which are 7.5% and 10.3% higher than the accuracy of Spatiotemporal Multiplier network, respectively. Overall, our model outperforms the Spatiotemporal Multiplier network when the number of training videos per class is less than 60 and 40 on UCF101 and HMDB51, respectively.

We also try to use ResNet-152 for both spatial and temporal stream but the accuracy is much lower than original model with very few training samples. We believe the reason is the 152 layers ResNet contains much more parameters than 50 layers one which is not suitable for the few shot task. The second framework is the unsupervised LSTM (Srivastava *et al.*, 2015) which is pre-trained on a 300 hours YouTube data then transformed to the supervised learning. For fair comparisons, our framework also only uses the RGB frames as inputs. The results are shown in Figure 3.6c and Figure 3.6d. We notice that our model slightly underperforms the LSTM when the size of the training set is extremely small (1 or 2 videos per class). We believe that is due to the LSTM is pre-trained on a large number of videos and this is confirmed when we using Kinetics pre-trained I3D as frame feature extractor. As the size of the labeled dataset grows, the gap becomes smaller. When the number of training video per class is 4, our model has accuracy 58.2% on UCF101 and 29.5% on HMDB51 which are higher than the unsupervised LSTM on both datasets. We calculate the training parameters that are introduced by the importance score learner, since the only training procedure is applied on the learner. As shown in Table 3.8, the TAVs only introduce 2.1M parameters, which is 97.5% and 97.4% smaller than the Spatiotemporal Multiplier Network and unsupervised LSTM, respectively.

Instead of using 2D network as the backbone, we also evaluate the TAVs with 3D frame feature extractor. We use I3D (Carreira and Zisserman, 2017) as the backbone and compare the performance of TAVs with base I3D and Temporal Structure Filter (TSF) (Piergiovanni and Ryoo, 2018a) with few training videos. The results are shown in Table 3.9. We use the authors provided codes for base I3D and implement the TSF according to the paper. For the experiments, all layers before the 3D average pooling layer are frozen. We repeat the experiments 3 times and report both round accuracies and average accuracy. In Table 3.9, we can see that as the number of training videos increases, the accuracies for all three frameworks also increase. The TSF outperforms the base I3D when the number of training videos < 50 and < 8 on UCF101 and HMDB51, respectively. Our approach outperforms the base I3D and TSF with all small training sets except when the number of training videos $= 64$ on HMDB51. We try to fully fine-tune the I3D on both datasets. However, the results are much lower than only train the fully connected layer with few training videos. Furthermore, we also implement the TSF according to the code provided by the authors (initialize the width of Cauchy distribution according to the length of videos which is different to the paper) and evaluate on HMDB51 with 8 training videos per class. It achieves 53.8 over three rounds bu still 5.2% less than ours. We also compare our approach with I3D on Diving48 dataset (Li *et al.*, 2018), see details in the appendix.

3.9 Comparison with the state-of-the-art on Full Datasets

Finally, we compare our model to the state-of-the-art action recognition results on full UCF101 and HMDB51 datasets. The performance is summarized in Table 3.10. The table is divided into three sets. The first set compares models that use only RGB data. The second set compares models that use optical flow features only. Models in the third set use both.

On RGB data, our ResNet-152 based model performs 10.2% and 18.4% better than

Methods	Backbone	Pretrain	Fully Fine-tuning	UCF101	HMDB51
Two-stream (spatial) (Simonyan and Zisserman, 2014)	Two-stream	ImageNet	Yes	73	40.5
Two-stream (spatial) (Feichtenhofer *et al.*, 2017)	ResNet-152	ImageNet	Yes	83.4	46.7
Unsupervised LSTM (spatial) (Srivastava *et al.*, 2015)	LSTM	300 hrs vids of Sports-1M	No	75.8	44.0
I3D (Carreira and Zisserman, 2017)	I3D	ImageNet+Kinetics	Yes	95.6	74.8
TSF (Piergiovanni and Ryoo, 2018a)	I3D	ImageNet+Kinetics	No	91.1	60
Ours	ResNet-152	ImageNet	No	83.2	58.9
Ours	I3D	ImageNet+Kinetics	No	95	69.8
Two-stream (temporal) (Simonyan and Zisserman, 2014)	Two-stream	-	Yes	83	54.6
Two-stream (temporal) (Feichtenhofer *et al.*, 2017)	ResNet-152	-	Yes	87.2	60
Unsupervised LSTM (temporal) (Srivastava *et al.*, 2015)	LSTM	300 hrs vids of Sports-1M	No	77.7	-
I3D (Carreira and Zisserman, 2017)	I3D	ImageNet+Kinetics	Yes	96.7	77.1
TSF (Piergiovanni and Ryoo, 2018a)	I3D	ImageNet+Kinetics	No	92.3	62.8
Ours	ResNet-152	ImageNet	No	78.5	51.3
Ours	I3D	ImageNet+Kinetics	No	96	73.4
Two-stream (Simonyan and Zisserman, 2014)	Two-stream	ImageNet (spatial)	Yes	88	59.4
Two-stream (Feichtenhofer *et al.*, 2017)	ResNet-152	ImageNet (spatial)	Yes	91.8	63.8
St Multiplier (Feichtenhofer *et al.*, 2017)	ResNet-50,152	ImageNet	Yes	94.2	68.9
Unsupervised LSTM (Srivastava *et al.*, 2015)	LSTM	ImageNet	No	84.3	-
ActionVLAD (Girdhar *et al.*, 2017)	VGG16	ImageNet	Yes	92.7	66.9
I3D (Carreira and Zisserman, 2017)	I3D	ImageNet+Kinetics	Yes	98	80.7
TSF (Piergiovanni and Ryoo, 2018a)	I3D	ImageNet+Kinetics	No	93.3	63.8
Ours	ResNet-152	ImageNet	No	89.8	64.2
Ours	I3D	ImageNet+Kinetics	No	97.1	77

Table 3.10: Comparison with state-of-the-art action recognition models on full UCF101 and HMDB51. The fully fine-tuning: Yes indicates the backbone is end-to-end fine-tuned on UCF101 and HMDB51, No indicates the backbone is frozen during the experiments.

the original two-stream model on UCF101 and HMDB51, respectively. The ResNet-152 based two-stream framework performs slightly better than our model on UCF101, but ours do 12.2% better than theirs on HMDB51 even our frame feature extractor is not trained on the UCF101 and HMDB51 datasets. When switch the backbone to I3D, our model achieve 95% and 69.8% on UCF101 and HMDB51 without training the frame feature extractor, which are 4.9% and 9.8% higher and only 0.6% and 5% lower than TSF and base I3D, respectively.

When use ResNet-152 as the frame feature extractor, the performance of our model on optical flow data is just passable. We believe this is due to the feature extractor is only trained on ImageNet. As the distribution of optical flow is different from the RGB images, the extracted features can not correctly represent the optical flow images. This is proved when we use I3D as backbone (pre-training with optical flow features of Kinetics). Without training the I3D on UCF101 and HMDB51, our model achieve 96% on UCF101 and 73.4% on HMDB51.

When we combine predictions from the RGB and flow models, we obtain 89.8% and 64.2% on UCF101 and HMDB51 with ResNet-152, respectively. Our results are 2% lower and 0.4% better than the ResNet-152 based two-stream network on UCF101 and HMDB51, respectively. The performance of our model is 4.5% and 4.7% lower compared to the St Multiplier network. However, it is interesting to note that our model outperforms TSF on HMDB51 even we use ResNet-152 instead of I3D as the backbone. When use I3D as frame feature extractor, the performance of our model is really closed to the base I3D which are only 0.9 and 2.7 lower than base I3D on UCF101 and HMDB51, respectively.

Our approach also outperform Temporal Segment Networks (TSN) (Wang *et al.*, 2016) and partially fine-tuned I3D with only use RGB images on Diving48 dataset, please see details in the appendix.

3.10 Conclusion

In this paper, we propose TAVs to recognize human actions with very few labeled training videos. We analyze the performances of different initialized TAVs with optimized parameters with very few training videos. The best performance is achieved by using the SLDG TAVs (details are in appendix). We show that the framework which adopts TAVs can learn discriminative video representation with very few labeled training samples. The performance is boost when apply stronger backbones (*e.g.* I3D). We believe our approach can be applied to other models to capture the temporal information of video in other tasks.

3.11 Appendix

3.11.1 Introduction

We provide the hyper-parameters evaluation of Temporal Attention Vectors (TAVs) on UCF101. We also evaluate the TAVs on the Diving48 dataset with two backbones (frame/clip features extractor), ImageNet pre-trained resNet-152 (He *et al.*, 2016) and Kinetics 400 pre-trained I3D (Carreira and Zisserman, 2017) networks. Both backbones are not trained with Diving48 during the entire evaluation.

3.11.2 Diving48 Dataset

The newly released Diving48 dataset (Li *et al.*, 2018) which contains $15,943$ training and 2096 testing videos of professional divers performing 48 types of dives. We choose this dataset because unlike datasets such as Kinetics or UCF101, Diving48 is designed to minimize the bias towards particular scenes or objects.

3.11.3 Implementation details

During the following evaluations, we use the Short-long dynamic Gaussian (SLDG) to initialize the TAVs and use the Short-long term Fusion to fuse the TAVs for different temporal terms. The training and testing procedures are the same as we explained Sec. 4.3 in the paper. All evaluations of hyper-parameters are performed with both RGB and optical flow as input and the evaluation on Diving48 are performed with only RGB frames as input.

resNet-152 backbone. The input configuration is the same as we explained in Sec. 4.1.

I3D backbone. The videos are resized preserving aspect ratio so that the smallest dimension is 256 pixels, with bilinear interpolation. We use $10\times$ data argumentation with randomly select an 224×224 image crop. Pixel values are then rescaled between -1 and 1. We use video clip with 8 continuous frames as input. If the video length is not divisible by 8, we just simple discard the rest frames. We modify the kernel size of 3D average pooling layer to 1×7 and use the output as clip features (1024D).

Base I3D. The input configuration is the same as the I3D backbone except we use 64 frames video clip as the input. We follow the frame selection approach in (Wang *et al.*, 2016). First split the whole video into 64 chunks. If the video which has insufficient frames we just simply repeat the video multiple times. Then from each chunk, we uniformly choose a frame to construct the video clip. We only retrain the last layer on Diving48 dataset.

Importance Score Learner We use the inflate-shrink during the evaluations. The first convolution layer has 16 1 filters and others remain the same as indicate in the paper.

3.11.4 Analysis of the SLDG Vector

We introduce the SLDG initialized TAVs in Section 3.6 which is generated by repeating the DG initialization N times and each time the number of chunks is set to m_n. As the number of iteration increases, we highlight more information on different temporal range. In this section, we focus on finding a good combination between the iteration number N and the set of chunk numbers m.

The results are shown in Table 3.11. We first focus on the top and middle parts of the table. We notice that i). When the iteration number N does not change, as

# vectors K	# iteration N	# chunks m_n	Accuracy
7	2	6, 1	78.4
8	2	6, 2	78.3
9	2	6, 3	77.7
9	3	6, 2, 1	75.4
9	2	8, 1	77.4
10	2	8, 2	77.1
12	2	8, 4	76.7
14	3	8, 4, 2	76.5
15	4	8, 4, 2, 1	75.7
17	4	8, 6, 2, 1	75.1
17	2	16, 1	76.9
18	2	16, 2	76.5
20	2	16, 4	76.1
21	2	16, 5	76
21	3	16, 4, 1	76.6
24	2	16, 8	74.8
28	3	16, 8, 4	75.9

Table 3.11: Average accuracy (%) on the UCF101 split 1 (the experiments are repeated 10 times and each time 10 training videos are uniformly chosen for each class) of SLDG with different parameters. The short-long term fusion is applied.

the value of K increases, the accuracy actually decreases. For example, the first three rows in the top and middle parts of Table 3.11 indicate the accuracy drops along with an increase of K. This is because the long-term information is not highlighted (when $m_n = 1$ or 2). ii). There is a negative correlation between the iteration number N and the accuracy when the number of SLDG K remains the same. For example, row 3 and row 4 show the accuracy decreases by 2.3% as N goes up to 3 from 2 when $K = 9$. Then we move to the bottom part of Table 3.11. However, the second observation is not valid when $m_1 = 16$. The fourth and fifth rows in the bottom part of the table show an increase in accuracy as the iteration number N goes up when $K = 21$. This is because the SLDG vectors with chunk numbers $(16, 5)$ only highlight the short-term temporal information but ignore the long-term one.

3.11.5 Analysis of Different Initialized TAVs with Different Fusion Approaches

In Section 3.6 and Section 3.7.2, we discuss four initialization approaches of TAVs to encode the frame features and three two-stream fusion approaches. The goal here is, given few labeled training data (*e.g.* 10 training video per class), finding the best-performed combination of the TAVs number and the fusion approach for the TAVs initialization approaches. Table 3.12 lists the fusion approaches (early concatenation, late fusion or short-long term fusion), the TAVs initialization approaches (Random, SSD, DG or SLDG) and the number of the TAVs that is used. The value of θ is set to 2 for DG TAVs and we sort the SSD TAVs with ascending order since we find the order does not impact their performance. Note that, the short-long fusion is only applicable to the SLDG TAVs.

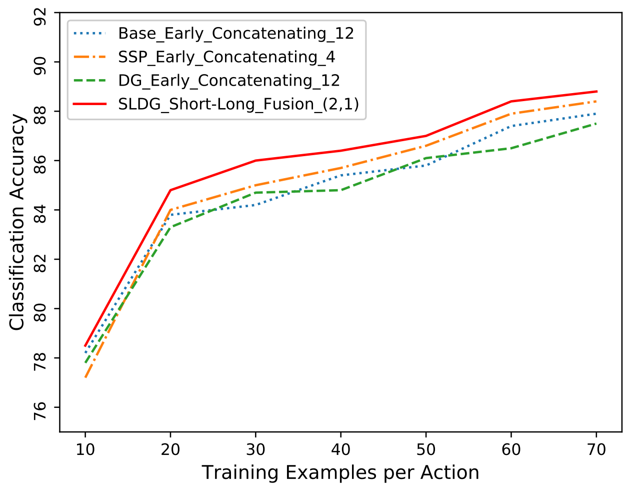
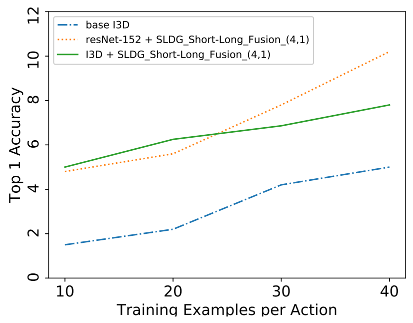


Figure 3.7: Accuracies (%) on UCF101 split 1 with change in the size of the labeled training set. The experiments are repeated 10 times and each time the training videos are uniformly chosen for each class.

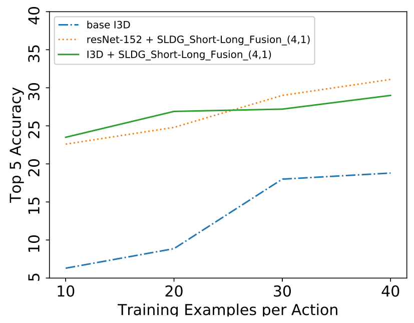
Fusion \ Temp. Vectors	Base			SSD			DG			SLDG		
	K=4	K=8	K=12	K=4	K=8	K=12	K=4	K=8	K=12	$K=3$, m=(2,1)	$K=6$, m=(4,2)	$K=12$, m=(10,2)
Early Concat.	77.5	77.8	**78.1**	**77.2**	76.9	76.2	74.2	73.1	**77.8**	77.6	77.3	77.2
Late Fusion	75.4	71.4	72.4	74.3	68.2	74.2	72	71	72.4	72.8	72.9	72.7
Short-Long term Fusion		-			-			-		**78.5**	78.4	76.8

Table 3.12: Average accuracy (%) on the UCF101 split 1 (the experiments are repeated 10 times and each time 10 training videos are uniformly chosen for each class) of different combinations of TAVs with different numbers and fusion approaches.

We first focus on the performance of the TAVs with different fusion approaches. Not surprisingly, the simple early concatenation outperforms the late fusion over all TAVs initialization approaches since the feature extractor is only trained with RGB images (*e.g.* ImageNet), the imprecise optical flow features result misprediction. For the SLDG initialization, the short-long term fusion provides marginal improvements (about 1%) than the early concatenation when $m = (2, 1)$ and $m = (4, 2)$. Then we investigate the effectiveness of using different number of the TAVs. Increasing the



(a) Top 1 Accuracy, RGB



(b) Top 5 Accuracy, RGB

Figure 3.8: Comparisons of TAVs using different backbones with I3D for action recognition with change in the size of the labeled training set on Diving48. The experiments are repeated 10 times and each time the training videos are uniformly chosen for each action class. The average accuracy (%) are reported.

number of them leads to a small improvement (0.6%) for the random initialized TAVs and a significant boost for the DG TAVs (3.6%). However, as the number of TAVs increases, the accuracy slightly drop for the SSD and SLDG TAVs. We also perform the same experiments on UCF101 split 1 with 20 training video per class and we get the same results.

We choose the best-performed combination of the TAVs number K and fusion approaches for each type of the TAVs and evaluate their performance with various size of the training set. The results are shown in Figure 3.7. Using SLDG TAVs with $m = (2, 1)$ and short-long term fusion outperforms other combinations over all training sets.

3.11.6 Evaluation of TAVs with Very Few Training Data of Diving48

In this section, we compare TAVs using resNet-152 and I3D as backbones with base I3D on Diving48 with change in the size of the labeled training set. All experiments are done with the same training sets. The top 1 and top 5 average accuracy are shown in Figure 3.8a and Figure 3.8b, respectively. Clearly, the accuracy has a significant gap between the base I3D and the I3D+TAVs. For example, with only 10 labeled training video per class, the top 1 accuracy of I3D+TAVs achieves 5.5% on Diving48 which are 3.3% higher than the accuracy of base I3D network. If we compare the top 5 accuracy, the gap even becomes larger (17.13% with 10 labeled training videos). Overall, I3D+TAVs outperforms the base I3D network with very few training videos. The interesting thing is, even we use a less powerful backbone (resNet-152), the TAVs still outperform the base I3D network with very few training data. Also, the performances of resNet-152+TAVs and I3D+TAVs are really close which shows the TAVs could effectively encode the temporal information of videos with both 2D or 3D backbones.

3.11.7 Comparison with the state-of-the-art on Full Diving48

Finally, we compare TAVs to the state-of-the-art action recognition results on Diving48 datasets. The performance is summarized in Table 3.13. The accuracy shows significant improvement when using TAVs with 2D framework. For example, the Temporal Segment Network(TSN)+TAVs outperform 5.7% than the end-to-end fine-tuning TSN network when using only RGB as input, even the backbone network never see the video in Diving48. The end-to-end fine-tuning seems much important for 3D

Methods	Framework	Input	Pre-train	Full-FT	Accuracy
TSN (Wang *et al.*, 2016)	2D	RGB	ImageNet (objects)	Yes	16.8
TSN (Wang *et al.*, 2016)	2D	RGB+FLOW	ImageNet (objects)	Yes	20.3
TSN+TAVs (Ours)	2D	RGB	ImageNet (objects)	No	22.1
I3D (Carreira and Zisserman, 2017)	3D	RGB	Kinetics (actions)	No	12.2
R(2+1)D (Tran *et al.*, 2018)	3D	RGB	Kinetics (actions)	Yes	28.9
I3D+TAVs (Ours)	3D	RGB	Kinetics (actions)	No	20.5

Table 3.13: Comparison with the state-of-the-art methods on the Diving48 dataset. The full-FT: "yes" indicates end-to-end fine-tuning on Diving 48, "no" means only train the last layer for I3D or the importance score learner of TAVs.

framework than 2D framework. For example, the end-to-end fine-tuning R(2+1)D (Tran *et al.*, 2018) network gives 28.9% classification accuracy which is higher than I3D+TAVs. We believe the reason is that the I3D backbone are not trained on Diving48 since the I3D and R(2+1)D networks show similar performance on other video action recognition datasets (*e.g.* Kinetics). However, when using TAVs on top of I3D, 8.3% accuracy improvement is still achieved comparing with the base I3D network.

3.11.8 Conclusions

In this supplementary material, we provide the detailed evaluation of TAVs on Diving48 dataset. We build the TAVs on both 2D (resNet-152) and 3D (I3D) backbones. The significant improvement is achieved on both backbones. The TAVs show the best performance with very few training videos. Also, compared to other state-of-the-art frameworks on full diving48 dataset, the TAVs still show competitive results, especially with 2D backbone.

Bibliography

Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *European conference on computer vision*, pages 428–441. Springer.

Girdhar, R. and Ramanan, D. (2017). Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems*, pages 34–45.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.

Jain, M., van Gemert, J. C., Mensink, T., and Snoek, C. G. (2015). Objects2action: Classifying and localizing actions without any video example. In *Proceedings of the IEEE international conference on computer vision*, pages 4588–4596.

Jiang, Y.-G., Dai, Q., Xue, X., Liu, W., and Ngo, C.-W. (2012). Trajectory-based modeling of human actions with motion reference points. In *European Conference on Computer Vision*, pages 425–438. Springer.

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

Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

Li, Y., Li, Y., and Vasconcelos, N. (2018). Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528.

Matikainen, P., Hebert, M., and Sukthankar, R. (2009). Trajectons: Action recognition through the motion analysis of tracked features. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 514–521. IEEE.

Mettes, P. and Snoek, C. G. (2017). Spatial-aware object embeddings for zero-shot localization and classification of actions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4443–4452.

Piergiovanni, A. and Ryoo, M. S. (2018a). Learning latent super-events to detect multiple activities in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5304–5313.

Piergiovanni, A. and Ryoo, M. S. (2018b). Temporal gaussian mixture layer for videos. *arXiv preprint arXiv:1803.06316*.

Piergiovanni, A., Fan, C., and Ryoo, M. S. (2017). Learning latent subevents in activity videos using temporal attention filters. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Sun, C., Shetty, S., Sukthankar, R., and Nevatia, R. (2015a). Temporal localization of fine-grained actions in videos by domain transfer from web images. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 371–380. ACM.

Sun, J., Wu, X., Yan, S., Cheong, L.-F., Chua, T.-S., and Li, J. (2009). Hierarchical spatio-temporal context modeling for action recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2004–2011. IEEE.

Sun, L., Jia, K., Yeung, D.-Y., and Shi, B. E. (2015b). Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605.

Wu, Z., Wang, X., Jiang, Y.-G., Ye, H., and Xue, X. (2015). Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 461–470. ACM.

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

Zhu, Y., Long, Y., Guan, Y., Newsam, S., and Shao, L. (2018). Towards universal representation for unseen action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9436–9445.

Chapter 4

CLTA: Contents and Length based Temporal Attention for Few-shot Video Classification

4.1 Citation and Main Contributor

Bo, Yang, Yangdi Lu, Wenbo He, Bohua Qiu and Muheng Wei. "CLTA: Contents and Length based Temporal Attention for Few-shot Video Classification." Submitted to CVPR2021.

The main contributor to this paper is the first author - Yang Bo (contributes more than 80%).

4.2 Abstract

Few-shot video classification has been attracting increasing attention due to the difficulty in acquiring the properly labeled training samples. In the few-shot video classification, the temporal correlation among frames of a video is effective and important to capture the video descriptor. In this paper, we propose a Contents and Length based Temporal Attention (CLTA) model, which learns customized temporal attention for the individual video to tackle the few-shot video classification problem. CLTA utilizes the Gaussian likelihood function as the template to generate temporal attention and trains the learning matrices to study the mean and standard deviation based on both frame contents and length. We show that with precisely captured temporal attention, even a simple linear classifier can achieve competitive results for few-shot video classification. Furthermore, by adopting the metric learning procedure, CLTA achieves comparable performance compared to the state-of-the-art few-shot video classification approaches.

4.3 Introduction

The performance of deep learning models on visual recognition tasks heavily relies on abundant labeled training instances. However, it is error-prone and labor-intensive to obtain the labeled training samples. Consequently, the problem of classifying unseen classes with few examples, known as few-shot classification, has attracted considerable attention. The majority of recent few-shot learning efforts focus on image classification. Examples include model initialization based methods (Ravi and Larochelle, 2016; Finn *et al.*, 2017), metric learning based methods (Vinyals *et al.*, 2016; Snell *et al.*, 2017; Sung *et al.*, 2018), gradient based methods (Rusu *et al.*, 2018; Simon *et al.*, 2020; Zintgraf *et al.*, 2019) and hallucination based methods (Hariharan and Girshick, 2017; Zhang *et al.*, 2019). For video data, the few-short classification is more required but more challenging. It is even harder to obtain correct labeled videos than images, especially for the tasks which need precise boundaries (e.g. multi-activities detection). It is hard to say the action starts and ends at a specific frame.

The video classification needs to consider both spatial information in the frame and the temporal correlation among frames. Directly apply the deep 3D Convolution Neural Networks (CNNs) (Carreira and Zisserman, 2017; Tran *et al.*, 2015, 2018) or CNNs + Recurrent Neural Networks (RNNs) (Ballas *et al.*, 2015; Donahue *et al.*, 2015) with few data lead to severe overfitting by training a complex model with deficient data. Some researchers proposed to use Generative Adversarial Networks (GANs) (Kumar Dwivedi *et al.*, 2019) or data augmentation with a self-learning manner (Zhang *et al.*, 2020) to increase training data. Another branch of works focuses on capturing temporal correlation among frames (Zhu and Yang, 2018; Bishay *et al.*, 2019; Cao *et al.*, 2019; Bo *et al.*, 2020), while the spatial information in a frame is extracted by a pre-trained CNN. Hence, the model is much simpler than deep 3D CNNs or CNN+RNNs. Therefore, they are more efficient and less likely to incur overfitting. In this paper, we focus on learning the temporal attention of videos to address the few-shot video classification problem.

One of the existing few-shot video classification approaches preserves temporal correlation among frames via a self-attention manner (Zhu and Yang, 2018). However, this work is based on memory networks that require extra computational and space resources, and the temporal weights generated by self-attention is a “rough” curve.

We consider that in a regular video, if a frame is important (e.g. contains the action), the frames near that frame are very likely also important since the action is continuous. Therefore, the temporal weights for these frames should be similar. (Bishay *et al.*, 2019; Cao *et al.*, 2019) propose alignment based approaches, which cause huge computational overhead, to preserve temporal information of videos. (Bo *et al.*, 2020) apply the same temporal weights to different videos to generate video descriptor. Thus these temporal weights work as filters over temporal dimension (same as the kernels in the convolutional layer). The important scenes of various videos usually occur at different frames. Considering two videos with “run” and “long jump” actions, respectively. The latter contains “run” and then “jump” actions, hence applying the temporal filters of “run” to the novel action “long jump” may cause the wrong prediction. Even two videos have the same labels and lengths, the important scenes may still occur in different temporal positions. For example, given two videos are both labeled blasting sand and have the same length, if both videos zoom in on the subject which is important for classification, the zoom-in operations are not necessarily happening at the same time slot.

With these insights, we propose a Contents and Length based Temporal Attention (CLTA) to address the few-shot video classification task. CLTA leaves the spatial information preserving task to an ImageNet pre-trained CNN and only learns to tailor the temporal attention based on both video length and frame representations. Therefore, CLTA is simple and is easy to be trained with few videos. To be specific, CLTA utilizes Gaussian likelihood functions to provide the temporal weights for individual frames. Different from previous work (Bo *et al.*, 2020), CLTA trains two learning matrices to study a “contribution” scores of mean and standard deviation from each frame representation, respectively. It then uses the sum of these scores to define the Gaussian. During the training, both video contents and length are considered. In another word, CLTA study the ability of learning mean and standard deviation based on the video contents and length. Since the frame representations and length of various videos are usually different, CLTA is able to provide customized temporal attention to different videos.

We evaluate our approach on three datasets, UCF101: many samples with the same class are taken from the same video, HMDB51: samples with the same class are taken from different videos, Kinetics: samples with the same class are taken from different videos and lots of the samples have the same length (videos are all 10 seconds long but some of them are recorded with 30 fps, some are recorded with 25 or 15 fps). Even using a linear classifier, CLTA achieves competitive performance for the few-shot video classification task. By adopting the cosine distance classifier which is widely used in few-shot learning, CLTA gives comparable or better performance compared to the state-of-the-art methods.

4.4 Related Work

Action Classification The CNN-based approaches have been widely applied to the video classification area. Some works extend the CNNs to three-dimensional (Carreira and Zisserman, 2017; Ji *et al.*, 2013; Tran *et al.*, 2015, 2018; Varol *et al.*, 2018) to capture the spatio-temporal information of video. Other works still use 2D CNNs but process color and optical flow information in parallel for the subsequent late fusion of their separate classification scores (Simonyan and Zisserman, 2014; Wang *et al.*, 2015; Feichtenhofer *et al.*, 2017). An alternative solution focuses on temporal rather than spatial information. The examples include modeling the temporal structure of video by various temporal pooling approaches (Yue-Hei Ng *et al.*, 2015; Girdhar and Ramanan, 2017), rank functions (Fernando *et al.*, 2015), k-means clustering (Girdhar *et al.*, 2017) and distribution functions (Piergiovanni and Ryoo, 2018a). Recurrent Neural Networks have also been used to encode the temporal information for learning video representations (Ballas *et al.*, 2015; Donahue *et al.*, 2015; Srivastava *et al.*, 2015; Sun *et al.*, 2015a,b; Wu *et al.*, 2015).

Few-shot Learning of Image Classification Many efforts have been devoted to overcome the few-shot image classification problem. Some of the recent works address this by focusing on good model initialization (Finn *et al.*, 2017, 2018; Nichol *et al.*, 2018; Rusu *et al.*, 2018). Therefore, when applying the classifier to predict novel classes, it can be learned with a limited number of labeled examples and a small number of gradient update steps. Another line of work focuses on learning an optimizer (Ravi and Larochelle, 2016; Munkhdalai and Yu, 2017). Examples include using the LSTM-based meta-learner to replace the stochastic gradient descent optimizer (Ravi and Larochelle, 2016) and applying a weight-update mechanism with external memory (Munkhdalai and Yu, 2017). These initialization based methods can achieve rapid adaption with a limited number of training examples for novel classes. Another category focuses on similarity comparison. Researchers adopt component-wise distance (Koch *et al.*, 2015), cosine similarity (Vinyals *et al.*, 2016; Gidaris and Komodakis, 2018; Qi *et al.*, 2018), Euclidean distance to class-mean representation (Snell *et al.*, 2017) and Graph Neural Network (Garcia and Bruna, 2017) to measure the similarity between images.

Few-shot Learning of Video Classification There are only few works to address few-shot video classification problem. Zhu and Yang (Zhu and Yang, 2018) propose the Compound Memory Network (CMN) which embeds frame features by multi-saliency function and predicts the class by comparing the dot product similarity of inputs. Bishay et al. (Bishay *et al.*, 2019) calculate the relation between the query and support videos by measuring the similarity between aligned segments. Zhang et al (Zhang *et al.*, 2020) train a 3D CNN with the self-supervised spatio-temporal mechanism to improve the robustness of the model and prevent over-fitting. Cao et al. (Cao *et al.*, 2019) train the frame feature extractor by minimizing the frame-wise cosine distance between the support and query videos.

Our work is similar to (Piergiovanni and Ryoo, 2018a; Bo *et al.*, 2020), both works learn the temporal weights to aggregate frame representations. (Piergiovanni and Ryoo, 2018a) utilizes Cauchy distributions as the template to generate temporal weights for frames. The centers and width parameters are defined by trainable parameters. (Bo *et al.*, 2020) applied Gaussian distributions to generate the temporal weights for action recognition. The mean and standard deviation are manually defined based on the length of videos. For each Gaussian, they introduced a trainable scale parameter. Both approaches share the temporal weights to various videos thus may cause misprediction especially only few training samples are available. In contrast, CLTA train two learning matrices to study the center and width parameters based on frame contents and video length instead of directly learning them. Therefore, our approach avoids sharing the same temporal weights to various videos. We discuss the detailed difference between CLTA with (Piergiovanni and Ryoo, 2018a; Bo *et al.*, 2020) in Section 4.6 and compare their performance in Section 4.7

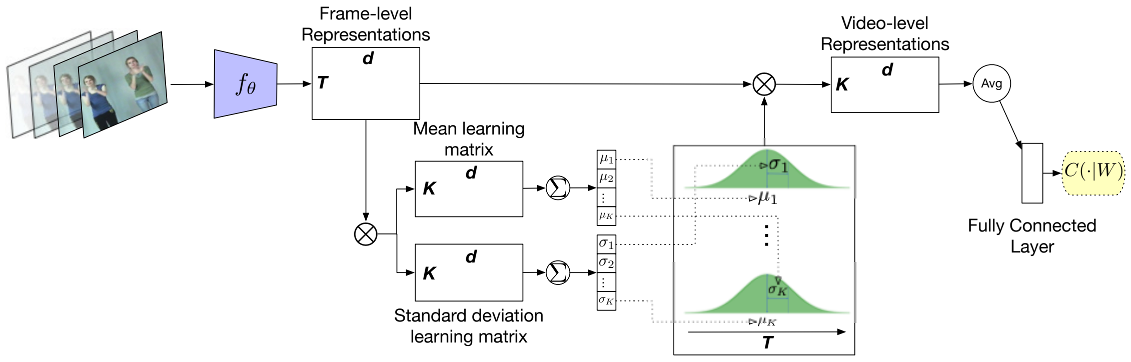


Figure 4.8: Overview of CLTA. f_θ is a 2D CNN which is used to extract frame representations. "$\otimes$" denotes matrix multiplication and the proper matrix transpose is applied. "Σ" denotes the addition operation according to the time dimension. $C(\cdot|W)$ is the classifier. Here, we show our approach with averaging video-level representations.

4.5 Our Approach

Given abundant labeled videos X_b of base classes and few labeled videos X_n of novel classes. Our goal is training a model on X_b, which also could generate distinctive representations for novel classes in X_n (unseen during training). Therefore, a classifier is able to classify them with only few labeled videos. We achieve this by proposing a Contents and Length based Temporal Attention (CLTA), the outline is shown in Figure 4.8. CLTA utilizes the Gaussian likelihood function to generate temporal attention for video frames. Since videos may have multiple crucial periods for classification, we apply multiple Gaussians to capture them. Instead of studying the mean and standard deviation directly, CLTA trains two learning matrices to study them based on the frame representations and video length. Therefore, CLTA is able to generate distinctive video representation for novel classes.

4.5.1 Establish Frame-level Correlations via CLTA

Given the frame feature extractor f_θ (e.g. resNet), the frame-level representation sequence of a video is written as $F = \{f_\theta(x_1), f_\theta(x_2), \ldots, f_\theta(x_T)\}$, where $f_\theta(x_t) \in \mathbb{R}^d$ is the representation of the t^{th} frame. The k^{th} temporal weights for the t^{th} frame $a_{k,t}$ is defined by a Gaussian likelihood function as follows,

$$a_{k,t} = exp(-\frac{1}{2}(\frac{t/Z - \mu_k}{\sigma_k})^2), \tag{4.5.1}$$

$$\mu_k = \frac{1}{Z}\sum_{t=1}^{T} Sigmoid(f_\theta(x_t) \cdot w_k^m), \tag{4.5.2}$$

$$\sigma_k = \frac{1}{Z}\sum_{t=1}^{T} Sigmoid(f_\theta(x_t) \cdot w_k^s). \tag{4.5.3}$$

Here, the mean μ_k and standard deviation σ_k are learned by adding the dot products of hidden variables w_k^m and w_k^s, where w_k^m and w_k^s are the row vectors of W^m and W^s, respectively. $W^m, W^s \in \mathbb{R}^{K \times d}$ are the mean and standard deviation learning matrices, with each frame-level representation $f_\theta(x_t)$. We apply the Sigmoid function to the dot products to make sure the contribution from each frame is between 0 to 1. Both μ_k and σ_k are then normalized by the maximum length of videos in the dataset Z to preserve the video length difference. Since CLTA encode both video content (via dot product) and length (via summation) during studying the mean and standard deviation of Gaussian, it is able to generate customized temporal attention for different videos (frame representations or length are different). The temporal attention $a_{k,t}$ is then normalized by a softmax function and used to aggregate the frame representations as the video-level representations. This could be formalized as

$$e_{k,t} = exp(a_{k,t}) / \sum_{t=1}^{T} exp(a_{k,t}), \tag{4.5.4}$$

$$v_k = \sum_{t=1}^{T} e_{k,t} f_\theta(x_t), \tag{4.5.5}$$

where v_k stands for the k^{th} video-level representation and $v_k \in \mathbb{R}^d$.

4.5.2 Design Choices

Video-level Representations Embedding

After we get K video-level representations, each of them focuses on one important scene of the video. Before we make the prediction, the video-level representations need to be aggregated to form a single video descriptor which describe the entire video. The next question is how to aggregate them? To be specific, we could treat every video-level representation equally important during the prediction by averaging them to get the video descriptor. Another way is learning a weight for each of video-level representations then calculate the weighted sum of them, $V = \sum_{k=1}^{K} s_k v_k,$, where s_k represents the soft weight for the k^{th} video-level representation. We compared the performance of embedding the video-level representation by averaging and soft weight in experiment and find that CLTA achieves the similar performance for few-shot video classification.

Classifier

Linear Classifier. Current Deep Neural Networks widely apply a linear layer followed by a softmax function as the classifier. The linear classifier $C(\cdot|W)$ makes prediction by calculating $W^T V \in \mathbb{R}^c$, where V stands for video descriptor and c is the number of classes need to predict.

Cosine Distance Classifier The importance of reducing intra-class variations of features has been highlighted in (Hu *et al.*, 2015; Gidaris and Komodakis, 2018). Training the model with a cosine distance classifier explicitly could reduce the intra-class variations of data (Qi *et al.*, 2018; Chen *et al.*, 2019). The cosine distance classifier $C(\cdot|W')$ makes predictions based on the cosine similarity scores $[s_1, s_2, \ldots, s_c]$, where c stands for the number of classes, $s_i = V \cdot w'_i / \|V\| \|w'_i\|$ and $W' = [w'_1, w'_2, \ldots, w'_c]$. The prediction probability for each class is obtained by normalizing these similarity scores with a softmax function. Intuitively, the learned weight vectors $[w'_1, \ldots, w'_c]$ can be interpreted as prototypes (similar to (Snell *et al.*, 2017; Vinyals *et al.*, 2016)) for each class and the classification is based on the cosine distance of the video descriptor to these learned prototypes.

CLTA is applicable for both linear and cosine distance classifiers, We notice that adopting cosine distance classifier, CLTA gain about 3% performance improvements, the detailed comparisons are shown in Section 4.7.

4.6 Discussion

Why use Gaussian? Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. Self-attention has been used successfully in a variety of natural language tasks (Parikh *et al.*, 2016; Paulus *et al.*, 2017; Vaswani *et al.*, 2017). A natural question is why using the Gaussian likelihood function to generate the temporal weights instead of directly learning them by the learning matrices. We consider that in a regular video, if a frame is important (e.g. contains the action), the frames near that frame are very likely also important since the action is continuous. Therefore, the temporal weights for these frames should be similar. Self-attention does guarantee that the temporal weights for adjacent frames are similar. Consequently, some temporal information might be neglected. In contrast, using the Gaussian likelihood function could generate smooth temporal attention weights. We found that a smooth attention curve (generate by Gaussian) could better represent the action than a "rough" curve (generate by learning matrices directly). We implement the self-attention as a baseline and the detailed comparisons are shown in Section 4.7 and Section 4.8.

Comparison to Similar Temporal Attention Approaches. Some recent works also use probability likelihood functions to generate the temporal attention for video understanding tasks (Piergiovanni and Ryoo, 2018a; Bo *et al.*, 2020). Piergiovanni et al. (Piergiovanni and Ryoo, 2018a) propose a Temporal Structure Filter (TSF) which trains multiple Cauchy likelihood functions as the templates to generate the temporal weights used to aggregate frame representations then use soft-attention to embed the aggregated frame representations. The centers and width of Cauchy are initialized by trainable and uniformly selected "seeds" between -1 to 1 then scaled according to the length of videos and exponential function, respectively. TSF adjusts the Cauchy distributions by training the "seeds". Therefore, the Cauchy distributions are shared with various videos which may cause it fails to capture some important scenes when applied to novel classes.

Bo et al. (Bo *et al.*, 2020) propose a Short-long Range Dynamic Gaussian (SLDG) for the few-shot action recognition task. They apply the Gaussian likelihood function to generate the temporal weights for each frame. The mean and standard deviation are manually defined based on the length of videos. Then they introduce an importance score learner to study the soft weights to embed the aggregated frame representations. SLDG still shares the same temporal weights to the video with same length since the mean and standard deviation are defined based on the video length. The soft weights only re-scale the Gaussian but since the weights are still shared with different videos, the videos with the same length still have the final temporal weights.

In few-shot learning, the aim is to classify novel classes with few training samples. Applying learned temporal weights for seen classes to novel class may cause temporal information loss and making the wrong prediction. In contrast to (Piergiovanni and Ryoo, 2018a; Bo *et al.*, 2020), CLTA train two learning matrices W^m and W^s to study the mean and standard deviation, respectively. During the study, both video contents and length are considered. In another word, CLTA study the ability of learning mean and standard deviation based on video contents and length during training. Therefore, it can be easily generalized to novel class. We implement both TSF and SLDG, the detailed comparisons are shown in Section 4.7 and Section 4.8

	Video-level Rep Embd	
num of Gaussian	Average	Weighted Sum
3	55.9	56.1
6	57.9	57.5
9	56.5	56.2

Table 4.8: We report 5-way 1-shot mean accuracy of CLTA with different number of Gaussian and video-level representations embedding methods on HMBD51. We adopt linear classifier for this experiment. The video-level Rep Embd indicates how the video-level representations are embedded as the video descriptor.

4.7 Experiments

4.7.1 Datasets

We evaluate our approach on three popular datasets. First, UCF101 (Soomro *et al.*, 2012) which consists of 13320 action videos in 101 categories. The second dataset is HMDB51 (Kuehne *et al.*, 2011) which contains 6766 videos that have been annotated for 51 actions. For UCF101 and HMDB51 datasets, we follow the split as in (Zhang *et al.*, 2020). We select 70 classes as the training set, 10 classes as the validating set and last 21 classes as the testing set for UCF101 and 31 actions as the training set, 10 actions as the validating set and 10 actions as the testing set for HMDB51. We also use the Kinetics dataset (Carreira and Zisserman, 2017) and follow the same split as in (Zhu and Yang, 2018) which samples 64 classes for training, 12 classes for validation, and 24 classes for testing. Since some of the video are not available, we select other videos in the same class to guarantee each class will have 100 videos. There are no overlap classes between the training, validating and testing sets for all these three datasets.

Video-level Rep Embd	Classifier	1-shot	5-shot
Averaging	Linear	57.9	80.1
Averaging	Cosine	60.1	82.4
Weighted Sum	Linear	57.5	79.8
Weighted Sum	Cosine	59.8	82.5

Table 4.9: We report 5-way mean accuracy of CLTA with different video-level representations embedding methods and classifiers on HMDB51. The number of Gaussian is set to 6. The video-level Rep Embd indicates how the video-level representations are embedded as the video descriptor.

4.7.2 Implementation Details

Following the data augmentation as in (Wang *et al.*, 2016), we randomly crop from four corners and the center of input frames and sample the width and height of each crop randomly from $\{256, 224, 192, 168\}$, followed by re-sizing to 224×224. The argumentation is applied for both original and horizontal flipped frames. We use ImageNet pre-trained 152 layers ResNet (He *et al.*, 2016) as the backbone for the experiments on UCF101 and HMDB51 and 50 layers ResNet for the experiments on Kinetics. For all experiments, we use RGB frames as the input for the backbone.

We follow the procedure in (Chen *et al.*, 2019). CLTA is first trained on the training set X_b and we choose the epoch which achieves the highest accuracy on the validating set. During testing, we construct support set by random selecting n classes from the testing set X_n, each of them contains k randomly selected samples, called n-way k-shot learning. The query set contains one sample from each of the n classes. Therefore, each episode has a total of $n(k + 1)$ examples. Beside of support and query set, we also uniformly select 50 samples from each of the n classes as the query validation set. Samples in the query set, query validation set and support set have no overlap with each other. We use the support set to re-train a classifier (linear or

	Linear		Cosine distance	
Way	Averaging	Weighted Sum	Averaging	Weighted Sum
5-way	57.9	57.5	60.1	59.8
6-way	53.5	54.2	56.5	56.9
7-way	51.2	52.9	52.0	53.2
8-way	48.4	49.5	50.7	51.7

Table 4.10: We report high way 1-shot mean accuracy of CLTA with different video-level representations embedding methods and classifiers on HMDB51. The number of Gaussian is set to 6.

cosine distance) and stop training when the accuracy on query validation set is about to decrease. Then the classifier is used to classify the samples in the query set. The mean accuracies are reported by random sampling 10000 episodes for all experiments.

The fully connected layer has dimension 1024, followed by a ReLU function and a batch normalization layer. Adopting these additional layers speed up the training also gives 1% improvements of CLTA. During the training phase, we fix the parameters of the backbone and only train CLTA by minimizing the standard cross-entropy classification loss using Adam optimizer with initial learning rate 0.001. The learning rate decays every 20 epochs by 0.1. We train our model at most 60 epochs with batch size 64. In the testing phase, we fix the parameters of both backbone and CLTA then use the videos in support set to retrain a new classifier 100 epochs with batch size 64 and learning rate 0.001 in each episode. The softmax scores of augmentations are averaged as the final score.

	Linear		Cosine distance	
Way	Averaging	Weighted Sum	Averaging	Weighted Sum
5-way	80.1	79.8	82.4	82.5
6-way	76.3	76.8	78.1	78.7
7-way	72.6	73.4	74.1	75.0
8-way	68.6	70.0	70.9	72.7

Table 4.11: We report high way 5-shot mean accuracy of CLTA with different video-level representations embedding methods and classifiers on HMDB51. The number of Gaussian is set to 6.

				UCF101		HMDB51	
Methods	Backbone	Fine-tune	Frame-level Rep Embd	1-shot	5-shot	1-shot	5-shot
3D Prototypical Net (Zhang *et al.*, 2020)	3D Conv-4	Y	3D Conv	57.1	78.3	38.1	53.2
3D RelationNet (Zhang *et al.*, 2020)	3D Conv-4	Y	3D Conv	58.2	78.4	38.2	53.2
3D SoSN (Zhang *et al.*, 2020)	3D Conv-4	Y	3D Conv	62.6	81.5	40.8	55.2
ARN (Zhang *et al.*, 2020)	3D Conv-4	Y	3D Conv	66.3	83.1	45.5	60.6
Self-attention	ResNet-152	N	trainable matrices	73.8	80.7	52.6	70.8
TSF (Piergiovanni and Ryoo, 2018a)	ResNet-152	N	Gaussian	74.7	84.3	52.9	72.1
SLDG (Bo *et al.*, 2020)	ResNet-152	N	Gaussian	76.0	84.5	53.1	73.6
CLTA+linear (ours)	ResNet-152	N	Gaussian	78.0	88.4	57.5	79.8
CLTA+cosine (ours)	ResNet-152	N	Gaussian	**80.3**	**90.7**	**59.8**	**82.5**

Table 4.12: Mean accuracy of 5-way video classification on UCF101 and HMDB51. The Frame-level Rep Embd indicates how the frame-level representations are embedded as video-level representation.

4.7.3 Comparison to Different Design Choices

The Number of Gaussian

We evaluate CLTA with different numbers of Gaussian and video-level representation embedding approaches on HMDB51. We adopt linear classifier for this experiment and report the 5 way 1-shot classification results in Table 4.8. Increasing the number of Gaussian used in CLTA does not necessary improve the performance. When the number is set to 6, CLTA gives the best performances. Also, averaging the video-level representation achieves the same level of performance as applying weighted sum for 1-shot classification. We set the number of Gaussian to 6 for the following experiments.

Classifiers

Next, we show the performance of CLTA with different classifiers on HMDB51 in Table 4.9. In contrast with cosine distance classifier which a great increase (around 5% on mini-ImageNet dataset) of the performance for few-shot image classification (Chen *et al.*, 2019), directly adopting the cosine distance classifier gives a relative small increase (around 2%) in the performance of CLTA comparing to just using linear classifier. One reason is that the backbone is not fine-tuned which limits the ability of cosine distance classifier to reduce the intra-class variation. On the other hand, applying soft-attention to embed the video-level representation does not necessary improves the performance of CLTA. We believe that is because CLTA has already learned a good enough temporal attention which no need to re-weight.

High Way Classification

We further evaluate CLTA in high way 1-shot and 5-shot classification on HMDB51, the results are shown in Table 4.10 and Table 4.11. Overall, as the increasing of the number of way, the classification become harder and the performance of CLTA decreased for both 1-shot and 5-shot classification. Although the performances of CLTA are not shown much difference by adopting soft attention to embed video-level representation compared to averaging them for 5-way 1-shot classification. As the problem became harder (high way classification), adopting soft attention outperforms averaging the video-level representation for CLTA. Compared the performance of CLTA with linear and cosine distance classifiers in high way scenarios, we can conclude that applying linear classifier achieves the same level of performance as using cosine distance classifier if embed the video-level representations by averaging or soft

Methods	Backbone	Fine-tune	Frame-level Rep Embd	1-shot	5-shot
Matching Net (Zhu and Yang, 2018)	ResNet-50	N	Averaging	53.3	74.6
MAML (Zhu and Yang, 2018)	ResNet-50	N	Averaging	54.2	75.3
CMN (Zhu and Yang, 2018)	ResNet-50	N	Multi-saliency	60.5	78.9
TARN (Bishay *et al.*, 2019)	C3D	N	Alignment	66.6	80.7
TSN++ (Cao *et al.*, 2019)	ResNet-50	Y	Averaging	64.5	77.9
CMN++ (Cao *et al.*, 2019)	ResNet-50	Y	Multi-saliency	65.4	78.8
TRN++ (Cao *et al.*, 2019)	ResNet-50	Y	Multilayer Perceptron	68.4	82.0
TAM (Cao *et al.*, 2019)	ResNet-50	Y	Alignment	**73.0**	**85.8**
Self-attention	ResNet-50	N	Trainable Matrices	65.7	79.5
TSF (Piergiovanni and Ryoo, 2018a)	ResNet-50	N	Gaussian	63.9	77.2
SLDG (Bo *et al.*, 2020)	ResNet-50	N	Gaussian	64.2	78.1
CLTA+linear (ours)	ResNet-50	N	Gaussian	69.6	82.7
CLTA+cosine (ours)	ResNet-50	N	Gaussian	71.9	84.1

Table 4.13: Mean accuracy of 5-way video classification on Kinetics. The Frame-level Rep Embd indicates how the frame-level representations are embedded as video-level representation. The "++" sign indicates that the model using cosine distance classifier and episode-base training procedure

attention approaches.

In the following experiments, we use the soft weights as the default video-level representation embedding approach for CLTA with both linear and cosine distance classifiers and the number of Gaussian is set to 6.

4.7.4 Evaluation on UCF101 and HMDB51

We compared CLTA to other approaches that use 3D CNNs as the backbone and the approaches we mentioned in Section 4.6, the results are shown in Table 4.12. Zhang et. al (Zhang *et al.*, 2020) report the performance of Prototypical Network (3D Prototypical Net (Snell *et al.*, 2017)), Relation Network (3D RelationNet (Sung *et al.*, 2018)), Second-order Similarity Network (SoSN) (Zhang and Koniusz, 2019) and Action Relation Network (ARN) (Zhang *et al.*, 2020) by using a 4 layers trained from scratch 3D CNN as the backbone. We implement the soft-attention baseline, TSF (Piergiovanni and Ryoo, 2018a) and SLDG (Bo *et al.*, 2020). For the self-attention baseline, the only difference compared to CLTA is that we use two learning matrices to generate the temporal weights for each frame directly rather than using Gaussian likelihood function. The dimension of learning matrices are also $6 \times d$. The temporal attention are then also normalized by softmax function. We calculate the weighted sum of frame representations to generate video-level representations. The video-level representations are then averaged as the video descriptor. For TSF, we adopt the Gaussian likelihood function instead of Cauchy to achieve fair comparison and follow all other setting mentioned in (Piergiovanni and Ryoo, 2018a). For SLDG, we use the default setting as described in the paper (Bo *et al.*, 2020).

The models focusing on temporal information preserving (second part of Table 4.12) show great improvements on UCF101 and HMDB51 for both 1-shot and 5-shot classification compared to the approaches using 3D CNNs backbones. It shows a large image dataset pre-trained deep 2D CNNs also have the potential as the backbone for the few-shot video classification even without fine-tuning. Also, the temporal information may play a more important role than spatial information in few-shot video classification. Compared to the self-attention baseline with SLDG and TSF, even they almost achieve the same level performance for 1-shot classification but the gap is become bigger for 5-shot classification. We may conclude that using the distribution likelihood function generates better temporal weights for actions. The CLTA outperform all these methods on UCF101 and HMDB51. By comparing the result of our approach with SLDG and TSF which also use distribution likelihood function to generate temporal weights for video frames, we can conclude that customizing the temporal attention for individual video (CLTA) gives better generalization to novel classes than sharing the same temporal weights over different videos (TSF and SLDG).

4.7.5 Evaluation on Kinetics

We compared our approach to other state-of-the-art approaches on Kinetics as shown in Table 4.13. The "++" sign indicates that the model using cosine distance classifier as described in (Chen *et al.*, 2019) and episode-base training procedure. Compared the models which average the frame-level representations (e.g. Matching Net, MAML, TSN++) to the temporal patterns preserved models (e.g. CMN, TARN, TRN++), the methods which consider the temporal correlation between frames outperform those models which averaging the frame features. We can conclude that the averaging operation does not preserves the video temporal information well in few-shot classification. This is consistent with video classification on large datasets. Also, training model with proper distance function (e.g. cosine distance) have the potential to improve the model generalization on novel classes when comparing CMN with CMN++. Even for the model is not designed for few-shot video classification (TSN, TRN), training them to reduce intra-class variations could let them achieve comparable or better performance for few-shot classification. The TSF and SLDG show the similar performance comparing to the TRN++ for 5 shot learning but the accuracy is about 5% lower for 1 shot learning. However, in contrast with the experiment on UCF101 and HMDB51, Alignment model outperforms both TSF and SLDG. We believe that is because the Kinetics dataset contains lots of different labeled video with the same length. Therefore, the approaches which share temporal weights (e.g. TSF, SLDG) cannot preserve well for the temporal information.

For CLTA, applying the cosine distance classifier gives 2.3% improvements for 5-way 1-shot classification (69.6% versus 71.9%) and 1.4% improvement for 5-way 5-shot classification (82.7% versus 84.1%). We believe the reason is that the not fine-tuned backbone limit the potential of cosine distance classifier to reduce the intra-class variations between videos. By adopting cosine distance classifier, CLTA achieves the same level performance compared to TAM which is the current best approach for few-shot video classification. However, TAM introduce huge computational overhead since it utilizes alignment based approach, which compared the cosine distance frame-by-frame between the support and query videos, to preserve temporal correlation among frames. CLTA is much simpler, and not use a fine-tuned backbone.

4.8 Qualitative Results and Visualizations

We visualize the learned temporal weights of different videos (Figure 4.9) by Alignment model (first row in each part), TSF (second row in each part), SLDG (third row in each part) and CLTA (last row in each part). The top two are the learned temporal attention of video with various lengths but the same label. The top left and bottom left are the learned temporal attention of videos with the same length and the same labels. The bottom two are the learned temporal attention of videos with the same length but the different labels. We map the video length to the range $(0, 1]$ (x-axis). For example, if the length of the video is 300 (maximum video length in the dataset), the x-axis has the range $(0, 1]$. The y-axis indicates the value of temporal attention weights.

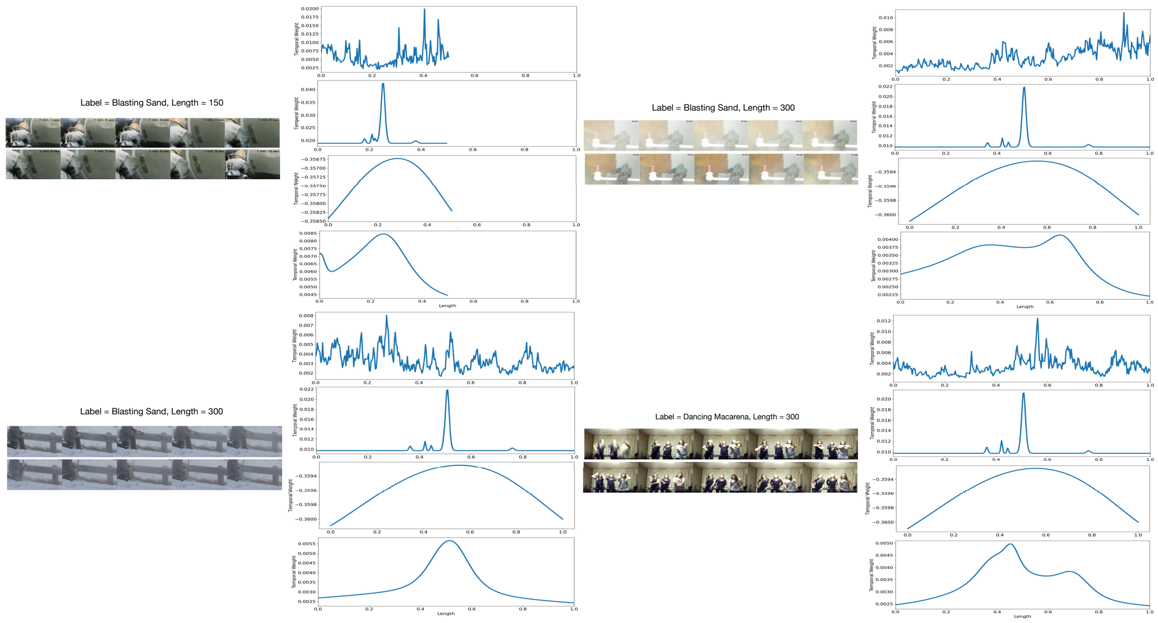


Figure 4.9: The visualization of learned temporal attentions by Alignment Model (first row in each part), TSF (second row in each part), SLDG (third row in each part) and CLTA (last row in each part). The length of video is mapped in range (0,1]. We choose four sample videos from the Kinetics dataset, video has label blasting sand and length 150 frames (top left), video has label blasting sand and length 300 frames (top right), another video has label blasting sand and length 300 frames (bottom left), and video has label dancing macarena and length 300 frames (bottom right). We extract the frames from the three videos with rate 1 fps and place them on top of each column.

We observe that all approaches could change the temporal attention weights according to the length of videos. However, if we compare the learned temporal attentions (second and third rows in top right and bottom left parts in Figure 4.9) of the videos with the same length and same label by TSF and SLDG, we observe that both approaches fail to adjust the temporal attention based on the content of videos as we mentioned before. Although TSF studies different temporal attention weights for the videos which have different labels (second row in the bottom left and bottom right parts in Figure 4.9), the center of the Cauchy distributions still remain the same. The self-attention baseline adjusts the temporal attention based on both length and content of video, but it gives a sharp temporal weights which may not correctly represent the temporal information of video. In contrast with them, CLTA is able to provide customized temporal attention weights for various videos based on their length and contents. Aslo, CLTA gives smooth temporal attention curves compared to alignment model.

4.9 Conclusion

We propose a Contents and Length based Temporal Attention (CLTA) for the few-shot video classification task based on the idea different videos have various temporal patterns. CLTA trains two learning matrices to study the mean and standard deviation based on both video contents and length. Therefore, CLTA is easily generalized to novel classes. We evaluate CLTA with different video-level embedding approaches and classifiers. Overall, using cosine distance classifier achieves the better performance compared to linear classifier. Also, we report the performance of CLTA for high way video classification. We evaluate CLTA on three datasets, CLTA outperforms other temporal attention methods (alignment, TSF and SLDG) for few-shot video classification. We show that CLTA can achieves competitive results for few-shot video classification even with a simple linear classifier. Furthermore, by adopting the metric learning procedure, CLTA achieves comparable performance compared to the state-of-the-art few-shot video classification approaches.

Bibliography

Bo, Y., Lu, Y., and He, W. (2020). Few-shot learning of video action recognition only based on video contents. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 595–604.

Finn, C., Xu, K., and Levine, S. (2018). Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527.

Hu, J., Lu, J., and Tan, Y.-P. (2015). Deep transfer metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 325–333.

Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille.

Munkhdalai, T. and Yu, H. (2017). Meta networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2554–2563. JMLR. org.

Nichol, A., Achiam, J., and Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Parikh, A. P., Täckström, O., Das, D., and Uszkoreit, J. (2016). A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Paulus, R., Xiong, C., and Socher, R. (2017). A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Qi, H., Brown, M., and Lowe, D. G. (2018). Low-shot learning with imprinted weights.

In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5822–5830.

Ravi, S. and Larochelle, H. (2016). Optimization as a model for few-shot learning.

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., and Hadsell, R. (2018). Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*.

Simon, C., Koniusz, P., Nock, R., and Harandi, M. (2020). On modulating the gradient for meta-learning. In *European Conference on Computer Vision*, pages 556–572. Springer.

Sun, C., Shetty, S., Sukthankar, R., and Nevatia, R. (2015a). Temporal localization of fine-grained actions in videos by domain transfer from web images. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 371–380. ACM.

Sun, L., Jia, K., Yeung, D.-Y., and Shi, B. E. (2015b). Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zhang, H. and Koniusz, P. (2019). Power normalizing second-order similarity network for few-shot learning. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1185–1193. IEEE.

Zhang, H., Zhang, J., and Koniusz, P. (2019). Few-shot learning via saliency-guided hallucination of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2770–2779.

Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., and Whiteson, S. (2019). Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pages 7693–7702. PMLR.

Chapter 5

Conclusion and Future Work

5.1 Conclusions

The core problem for solving the few-shot human action recognition problem is learning the discriminative video descriptor with only few training videos. The descriptor should preserve both spatial and temporal information thus can be easily classified. To generate the discriminative video descriptor given few training videos, the following questions should be answered. 1. How to preserve the entire temporal information of videos with various lengths. 2. How to prevent overfitting during training. 3. How to classify the video descriptors. In this thesis, we try to answer these questions and propose three temporal aggregation methods that can preserve the temporal information of the entire video. Meanwhile, our approaches does not or only introduce few training parameters and thus can easily prevent the overfitting problem. Finally, our approaches can adopt both linear classifiers and the clustering approaches.

We first proposed the DVD which recursively convolves the frame features with the normally chosen basis for a lower-dimensional space over the temporal direction.

By properly choosing the dimensions of the basis (e.g. 2), the temporal dimension of the frame feature sequence could be eventually reduced to 1. This approach could take various length frame sequences as input so that the temporal information over the whole video can be preserved. We evaluate the DVD with both hand-craft based frame features (SIFT, SURF, and 3D-SIFT) as well as deep learning-based features (VGGNet16 and VGGNet19) for the near-duplicate video detection and human action recognition tasks. The DVD shows a 14% improvement for near-duplicate video detection and 5.37% improvement for human action recognition tasks compared to the baselines, respectively.

Although the DVD can take various length frame sequences as input and involve the temporal information of the entire video, it brings additional computation overhead. Also, temporal attention is not considered. We then proposed the TAVs which also adapt the various length videos. The TAVs adopt multiple Gaussian distribution functions to generate the temporal attention for each frame/segment. The mean and standard deviation of each Gaussian is manually defined based on the length of the video to highlight a certain period of the video. Also, each TAV has a trainable important score that enables the most distinguishable period of the videos to have higher temporal attention. We evaluate the TAVs with various backbones (resNet152 and I3D) on two human action recognition datasets with the regular and few-shot scenario. The TAVs outperform the state-of-the-art video action recognition benchmarks with very few labelled training videos (92% on UCF101 and 59% on HMDB51, with 10 and 8 training videos per class, respectively). Furthermore, our approach can still achieve competitive results on full datasets without fully fine-tuning (97.1% on UCF101 and 77% on HMDB51).

Since the TAVs define the Gaussian based on the video length and share the important scores of TAVs to all videos. The videos with the same length will share the same TAVs even they belong to different action classes. To solve this problem, we further proposed the CLTA. Different than TAVs, CLTA studies the mean and standard deviation of Gaussian from the contents of each video frame called contribution scores. These scores are added together to form the mean and standard deviation of the Gaussian. Since the frames of different videos are various, CLTA can customize temporal attention for individual video. We evaluate CLTA on three human action recognition datasets, UCF101, HMDB51 and Kinetics in the few-shot scenario. Without fine-tuning the backbone (resNet50 and resNet152), CLTA achieves comparable or better results compared to the state-of-the-art few-shot approaches on all three datasets.

5.2 Future Work

The first future work is evaluating the CLTA for few-shot high-way classification scenario (similar to the TAVs). Current few-shot approaches usually evaluate the algorithm under 8-way classification. However, in real-life practice, there are much more actions for the recognition task. Thus, it is worth evaluating the performance of few-shot action recognition approaches on the full dataset. Another advantage of evaluating with the full dataset is saving the evaluation time. Current approaches uniformly select the fine-tuning and testing class from a larger set during each episode, therefore it needs to repeat multiple times to get a reliable performance of the methods on the dataset. Evaluating the performance on the full dataset could avoid that since the testing set is fixed. Another interesting future work is related to evaluate CLTA on reversed actions in the few-shot scenario. For example, if we play the action opening the door backward, the action becomes closing the door. Although CLTA could study the Gaussian from each frame, it may fail to distinguish this kind of action. Current approaches (e.g. bi-directional RNNs) used in natural language processing may not be suitable for the few-shot scenario. Moreover, all of our three approaches achieve similar or better results compared to the previous state-of-the-art methods, but they are not compared to each other. More specifically, comparing the performance between using hierarchical architecture (e.g. DVD) or temporal attention mechanism (e.g. TAVs and CLTA) to preserve the temporal information of videos. In summary, we will evaluate our algorithms by incorporating more complex real-life actions and improve them. We believe our approaches could benefit more video-related tasks.

Bibliography

Avci, A., Bosch, S., Marin-Perianu, M., Marin-Perianu, R., and Havinga, P. (2010). Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *23th International conference on architecture of computing systems 2010*, pages 1–10. VDE.

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., and Baskurt, A. (2011). Sequential deep learning for human action recognition. In *International Workshop on Human Behavior Understanding*, pages 29–39. Springer.

Bachman, P., Sordoni, A., and Trischler, A. (2017). Learning algorithms for active learning. *arXiv preprint arXiv:1708.00088*.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ballas, N., Yao, L., Pal, C., and Courville, A. (2015). Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*.

Bart, E. and Ullman, S. (2005). Cross-generalization: Learning novel classes from a single example by feature replacement. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 672–679. IEEE.

Bishay, M., Zoumpourlis, G., and Patras, I. (2019). Tarn: Temporal attentive relation network for few-shot and zero-shot action recognition. *arXiv preprint arXiv:1907.09021*.

Cao, K., Ji, J., Cao, Z., Chang, C.-Y., and Niebles, J. C. (2019). Few-shot video classification via temporal alignment. *arXiv preprint arXiv:1906.11415*.

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

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. (2019). A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*.

Choi, J., Krishnamurthy, J., Kembhavi, A., and Farhadi, A. (2018). Structured set matching networks for one-shot part labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3627–3636.

Coppola, C., Faria, D. R., Nunes, U., and Bellotto, N. (2016). Social activity recognition based on probabilistic merging of skeleton features with proximity priors from rgb-d data. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5055–5061. IEEE.

Coppola, C., Cosar, S., Faria, D. R., and Bellotto, N. (2019). Social activity recognition on continuous rgb-d video sequences. *International Journal of Social Robotics*, pages 1–15.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee.

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

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

Douze, M., Szlam, A., Hariharan, B., and Jégou, H. (2018). Low-shot learning with large-scale diffusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3349–3358.

Fei-Fei, L., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, **28**(4), 594–611.

Feichtenhofer, C., Pinz, A., and Zisserman, A. (2016a). Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941.

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

Feichtenhofer, C., Pinz, A., and Wildes, R. P. (2017). Spatiotemporal multiplier networks for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4768–4777.

Fernando, B. and Gould, S. (2016). Learning end-to-end video classification with rank-pooling. In *International Conference on Machine Learning*, pages 1187–1196.

Fernando, B., Gavves, E., Oramas, J. M., Ghodrati, A., and Tuytelaars, T. (2015). Modeling video evolution for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5378–5387.

Fink, M. (2005). Object classification from a single example utilizing class relevance metrics. In *Advances in neural information processing systems*, pages 449–456.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org.

Garcia, V. and Bruna, J. (2017). Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043*.

Gidaris, S. and Komodakis, N. (2018). Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375.

Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., and Russell, B. (2017). Actionvlad: Learning spatio-temporal aggregation for action classification. In *CVPR*, volume 2, page 3.

Goyal, R., Kahou, S. E., Michalski, V., Materzyńska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., Hoppe, F., Thurau, C., Bax, I., and Memisevic, R. (2017). The "something something" video database for learning and evaluating visual common sense.

Hariharan, B. and Girshick, R. (2017). Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3018–3027.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, **9**(8), 1735–1780.

Hoffman, J., Tzeng, E., Donahue, J., Jia, Y., Saenko, K., and Darrell, T. (2013). One-shot adaptation of supervised deep convolutional models. *arXiv preprint arXiv:1312.6204*.

Ji, S., Xu, W., Yang, M., and Yu, K. (2012). 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, **35**(1), 221–231.

Ji, S., Xu, W., Yang, M., and Yu, K. (2013). 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, **35**(1), 221–231.

Kar, A., Rai, N., Sikka, K., and Sharma, G. (2017). Adascan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3376–3385.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In

Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 1725–1732.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., *et al.* (2017). The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Kozerawski, J. and Turk, M. (2018). Clear: Cumulative learning for one-shot one-class image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3446–3455.

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

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

Kumar Dwivedi, S., Gupta, V., Mitra, R., Ahmed, S., and Jain, A. (2019). Protogan: Towards few shot learning for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0.

Lan, Z., Lin, M., Li, X., Hauptmann, A. G., and Raj, B. (2015). Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 204–212.

Lan, Z., Zhu, Y., Hauptmann, A. G., and Newsam, S. (2017). Deep local video feature for action recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1219–1225. IEEE.

Leo, M., D'Orazio, T., and Spagnolo, P. (2004). Human activity recognition for automatic visual surveillance of wide areas. In *Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*, pages 124–130.

Li, C., Zhong, Q., Xie, D., and Pu, S. (2019). Collaborative spatiotemporal feature learning for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7872–7881.

Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S. J., and Yang, Y. (2018). Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*.

Luo, Z., Zou, Y., Hoffman, J., and Fei-Fei, L. F. (2017). Label efficient learning of transferable representations acrosss domains and tasks. In *Advances in Neural Information Processing Systems*, pages 165–177.

Marszalek, M., Laptev, I., and Schmid, C. (2009). Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936. IEEE.

Miller, E. G., Matsakis, N. E., and Viola, P. A. (2000). Learning from one example through shared densities on transforms. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 464–471. IEEE.

Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **37**(3), 311–324.

Motiian, S., Jones, Q., Iranmanesh, S., and Doretto, G. (2017). Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 6670–6680.

Mulroy, S., Gronley, J., Weiss, W., Newsam, C., and Perry, J. (2003). Use of cluster analysis for gait pattern classification of patients in the early and late recovery phases following stroke. *Gait & posture*, **18**(1), 114–125.

Pfister, T., Charles, J., and Zisserman, A. (2014). Domain-adaptive discriminative one-shot learning of gestures. In *European Conference on Computer Vision*, pages 814–829. Springer.

Rautaray, S. S. and Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial intelligence review*, **43**(1), 1–54.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*.

Rodriguez, M. D., Ahmed, J., and Shah, M. (2008). Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

Salakhutdinov, R. and Hinton, G. (2009). Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455.

Salakhutdinov, R., Tenenbaum, J. B., and Torralba, A. (2011). Learning to learn with compound hd models. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, pages 2061–2069.

Salakhutdinov, R., Tenenbaum, J., and Torralba, A. (2012). One-shot learning with a hierarchical nonparametric bayesian model. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 195–206.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850.

Schuldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE.

Shyam, P., Gupta, S., and Dukkipati, A. (2017). Attentive recurrent comparators. *arXiv preprint arXiv:1703.00767*.

Sigurdsson, G. A., Varol, G., Wang, X., Farhadi, A., Laptev, I., and Gupta, A. (2016). Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., *et al.* (2016).

Mastering the game of go with deep neural networks and tree search. *nature*, **529**(7587), 484–489.

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

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

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

Srivastava, N., Mansimov, E., and Salakhudinov, R. (2015). Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852.

Sun, L., Jia, K., Yeung, D.-Y., and Shi, B. E. (2015). Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208.

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

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. (2018). A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6450–6459.

Varol, G., Laptev, I., and Schmid, C. (2018). Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, **40**(6), 1510–1517.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., *et al.* (2016). Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638.

Vishwakarma, S. and Agrawal, A. (2013). A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, **29**(10), 983–1009.

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

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

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

Wang, L., Li, W., Li, W., and Van Gool, L. (2017a). Appearance-and-relation networks for video classification. *arXiv preprint arXiv:1711.09125*.

Wang, X., Girshick, R., Gupta, A., and He, K. (2018a). Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Wang, Y.-X. and Hebert, M. (2016a). Learning from small sample sets by combining unsupervised meta-training with cnns. In *Advances in Neural Information Processing Systems*, pages 244–252.

Wang, Y.-X. and Hebert, M. (2016b). Learning to learn: Model regression networks for easy small sample learning. In *European Conference on Computer Vision*, pages 616–634. Springer.

Wang, Y.-X., Girshick, R., Hebert, M., and Hariharan, B. (2018b). Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7278–7286.

Wu, Y., Lin, Y., Dong, X., Yan, Y., Ouyang, W., and Yang, Y. (2018). Exploit the unknown gradually: One-shot video-based person re-identification by stepwise learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5177–5186.

Xu, Z., Zhu, L., and Yang, Y. (2017). Few-shot object recognition from machine-labeled web images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1164–1172.

Yan, W., Yap, J., and Mori, G. (2015). Multi-task transfer methods to improve one-shot learning for multimedia event detection. In *BMVC*, pages 37–1.

Yoo, D., Fan, H., Boddeti, V. N., and Kitani, K. M. (2017). Efficient k-shot learning with regularized deep networks. *arXiv preprint arXiv:1710.02277*.

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

Zhang, H., Zhang, L., Qui, X., Li, H., Torr, P. H., and Koniusz, P. (2020). Few-shot action recognition with permutation-invariant attention. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Zhang, Y., Tang, H., and Jia, K. (2018). Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. In *Proceedings of the european conference on computer vision (ECCV)*, pages 233–248.

Zhu, L. and Yang, Y. (2018). Compound memory networks for few-shot video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 751–766.

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

Zolfaghari, M., Singh, K., and Brox, T. (2018). Eco: Efficient convolutional network for online video understanding. *arXiv preprint arXiv:1804.09066*.