DEEP FEATURE FUSION FOR VIDEO-BASED ACTION RECOGNITION

A Thesis

by

CHENG CHENG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2020

Major Subject: Computer Engineering

ABSTRACT

3-Dimensional Convolutional Neural Networks (3D ConvNets) have been adopted for video-based action recognition task recently. Many 3D ConvNets, such as C3D, I3D, and Res3D, have been proposed and achieved great success. The model ensemble techniques have been very successful in achieving better performance over a single model. But model ensemble could not be adopted in this case given that the single 3D ConvNets model as a base learner is unrealistic. It remains an open question about how to achieve better performance by leveraging multiple 3D ConvNets models. To solve the problem, we present a two-stage framework to combine multiple 3D ConvNets models at the feature level. In the first stage, we treat each pretrained 3D ConvNets model as a feature extractor to extract features from raw videos. We fuse the extracted features of different 3D ConvNets models to form the new video representation and then train a classifier based on the new video representation in the second stage. We explore several widely-used feature fusion methods for deep features learned from different models, to learn more robust action representations from raw videos. We show that our framework outperforms any single 3D ConvNets model by a large margin and exhibits comparable performance to the state-of-the-art model on two video action recognition benchmarks.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to Professor Hu and Professor Qian, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this work. I would also like to thank Professor Shi and Professor Duffield for their advice and being the committee members. My grateful thanks are also extended to the AutoML group of DATA lab directed by Dr. Hu for their insightful suggestions.

I would also like to extend my thanks to all the staff of ECEN department for their help in offering me the resources and advice throughout my master program.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis (or) dissertation committee consisting of Professor Qian and Professor Shi and Professor Duffield of the Department of Electrical & Computer Engineering and Professor Hu of the Department of Computer Science & Engineering.

Some pre-trained models were provided by Kensho Hara on Github.

All other work conducted for the thesis (or) dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by a fellowship from Texas A&M University and a dissertation research fellowship from Texas A&M Engineering Experiment Station (TEES).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Video-based action recognition has been widely studied by researchers in the computer vision community, due to wide applications in many areas such as intelligent human-computer interaction, video surveillance and video anomaly detection. The goal of action recognition is to identify the actions from raw videos. Conventionally, descriptors, which are based on some efficient hand-crafted features, are proposed to catch the action information, such as HOG3D (Histogram of Oriented Gradient) [1] , SIFT3D (Scale-invariant Feature Transform) [2] and extended SURF [3]. These hand-crafted features are representative enough and could achieve good results. However, these hand-crafted features are very hard and time-consuming to design. Nowadays, Convolutional Neural Networks (ConvNets) [4] has become the standard for video action recognition. Different from classifying images of scenes and objects, video-based action recognition need to capture both spatial appearance and temporal motion. Conventional 2-Dimensional Convolutional Neural Networks (2D Convnets) has achieved great success for spatial appearance modeling, but it can't be directly used for temporal structure modeling. To mitigate the temporal modeling problem, Tran et al. [5, 6] propose to use the 3D ConvNet model as a feature extractor that model appearance and motion simultaneously. Carreira et al. [7] propose a new Two-Stream Inflated 3D ConvNet (I3D) that is based on 2D ConvNet inflation and achieve the state-of-the-art accuracy. Hara et al. [8] examine the architectures of various 3D ConvNets on a large-scale video datasets and conclude that using 3D ConvNets together with large-scale datasets will retrace the successful history of 2D ConvNets and ImageNet. Interestingly, the deep features learned by these models with a simple linear classifier, e.g. Support Vector Machine (SVM), can yield good performance on other video analysis tasks, such as video object detection and dynamic scene recognition. Since these models could learn efficient and compact spatio-temporal features for videos, can we do better if we fuse these deep features?

In this work, we explore several widely-used feature fusion methods for deep features learned from different models, to learn more robust action representations from raw videos. More specifically, we firstly train several state-of-the-art models on large labeled video datasets such as Kinetics, and we treat these trained models as video feature extractors. Then we explore different widely-used feature fusion methods for the deep features extracted from trained models, to find the best feature fusion method of deep features for the action recognition task. Also, we compare the feature fusion method with some popular model ensemble methods, which prove the effectiveness of our method. Our method also achieves comparable accuracy to the state-of-the-art models.

To summarize, our contributions in this work are three-fold. First, we design and conduct systematic and thorough experiments to investigate the best feature fusion techniques to generate video representation based on deep features learned by 3D ConvNets for the action recognition task. Second, we empirically find the best Convnets layer activations combination as a feature extractor. Finally, we show the effectiveness of our work by comparing the result with some common model ensemble methods on some benchmarks, e.g. UCF-101 and HMDB-51.

2. RELATED WORK

Video-based action recognition has been studied by researchers for decades. Previous works related to ours can be classified a s t wo c ategories: (1) c onvolutional n etworks f or video-based action recognition. (2) feature fusion.

2.1 Convolutional Neural Networks for Video-based Action Recognition

Inspired by the 2D Convnets breakthroughs in image tasks such as image classification and image object detection, conventional 2D Convnets are directly adopted for video feature learning, which can not achieve a significant a dvantage o ver t raditional h and-crafted f eatures f or the action recognition task. Then 3D Convnets, the extension of 2D ConvNets, are proposed to learn spatio-temporal feature from raw videos, and achieve great success and become the de facto standard for video recognition tasks. So we will review ConvNets related work in the following two subsections: (1) 2D ConvNets. (2) 3D ConvNets.

2.1.1 Models Based on 2D ConvNets

Inspired by the successful application of 2D ConvNets in image tasks, using the 2D ConvNets is a straightforward way for the video-based action recognition task. For example, Simonyan et al. [9] design two-stream ConvNets, RGB ConvNet and optical flow C onvNet, to m odel appearance and motion separately and fuse two streams together at last. They demonstrate that the two-stream ConvNets architecture can achieve very good performance despite limited training data. Temporal Segment Networks (TSN) [10] propose a sparse temporal sampling strategy based on two-stream ConvNets architecture. Karpathy et al. [11] trained deep ConvNets on a large weakly labeled dataset and achieve moderate success using the network as a feature extractor for other video classification t asks. M ore r ecently, L in e t a l. [12] p ropose a n ovel Temporal S hift M odule that facilitates information exchanged among neighboring frames, which get better performance than 3D ConvNets but maintain 2D ConvNets' complexity.

2.1.2 Models Based on 3D ConvNets

3D ConvNets can jointly learn spatial and temporal features simultaneously. 3D ConvNets is first proposed for action recognition by Ji et al. [13]. Tran et al. [5] conduct a systematic study for 3D ConvNets and train 3D Convnets (C3D) on large-scale datasets, which can model appearance and motion information simultaneously. In another study, inception based 3D Convnets (I3D) is proposed by Carreira et al. [7], which achieves state-of-the-art performance. Hara et al. [8] systematically examine the architectures of various resnet based 3D ConvNets on Kinetics datasets, and get a state-of-the-art result even trained from scratch.

2.2 Feature Fusion

Feature fusion, also known as feature encoding and descriptor/feature aggregation, has also been studied by many researchers for decades. We review related work in two directions: (1) feature fusion for other tasks. (2) feature fusion for action recognition.

2.2.1 Feature Fusion for Other Tasks

Feature fusion is widely used for image classification and retrieval tasks. Before the onset of deep learning, handcrafted features such as SIFT [14], combined with aggregation method such as Bag-of-Words (BoW) [15], Fisher Vectors (FV) [16] and Vector of Locally Aggregated Descriptors (VLAD) [17], are the most common methods for image classification and retrieval. Nowadays, Convnets are used to replace the previously hand-tuned feature extraction stage, where intermediate or higher layer activations of pre-trained Convnets models are used as features. For instance, Feng et al. [18] use Convnets as feature extractor so that it can learn more discriminative visual vocabularies from the geotagging images. Their resultant method achieves better performance than BoVW for geographical image classification. Gong et al. [19] extract Convnets activations at multiple scale levels, then perform orderless VLAD pooling of these activations at each level separately and concatenate them as new image representation. The resultant representation is more robust for image classification and retrieval. Ng et al. [20] propose an approach for extracting

Convnets features from different layers of the networks, and encode features into a single vector for each image using VLAD encoding. Their work also demonstrates that intermediate layers with finer scales produce better results for image retrieval than the last layer. Mohedano et al. [21] propose a simple image instance retrieval approach based on encoding the convolutional features of Convnets using the BOW encoding. Cao et al. [22] build an effective BoW model using ConvNets features.

2.2.2 Feature Fusion for Action Recogntion

As for video-based action recognition, early works focus on high-dimensional encodings of hand-crafted local spatio-temporal features. For example, Laptev et al. [23] propose an algorithm to detect sparse spatio-temporal interest points, which are then described using Histogram of Oriented Gradients (HOG) [24] and Histogram of Optical Flow (HOF). Finally, the features are then pooled over several spatio-temporal grids to encode into the BOW representation and combined with an SVM classifier. Nowadays, those hand-crafted local spatio-temporal features are replaced by deep convolutional features. Diba et al. [25] propose a bilinear model, which pools the activations of the last convolutional layers of pre-trained networks. Qiu et al. [26] propose a new quantization method and achieve a comparable result to the state-of-the-art model. Girdhar et al. propose a learnable spatio-temporal feature aggregation layer to learn a new video representation in an end-to-end way, where the feature aggregation layer is the variation of VLAD. Lan et al. [27] propose to train the deep convolutional networks on local inputs and treat the trained model as a local feature extractor, then aggregate the local deep features as video-level representation to classify videos in a second stage.

3. METHODOLOGY

Different from end-to-end models, our framework consists of two stages. We show an overview of the framework in Figure 3.1. In the first stage, different Convnets models, e.g. C3D [5], Res3D [8, 6], are trained on large labeled video datasets, Kinetics. These trained models are used as feature extractors to produce features for unseen datasets such as UCF-101 and HMDB-51. Then the features extracted from different models are aggregated together to produce more robust video representation. In the second stage, we learn a linear classifier that maps the video representation to the label.

As shown in Figure 3.1, there are several design choices for our framework: (1) Which layer of

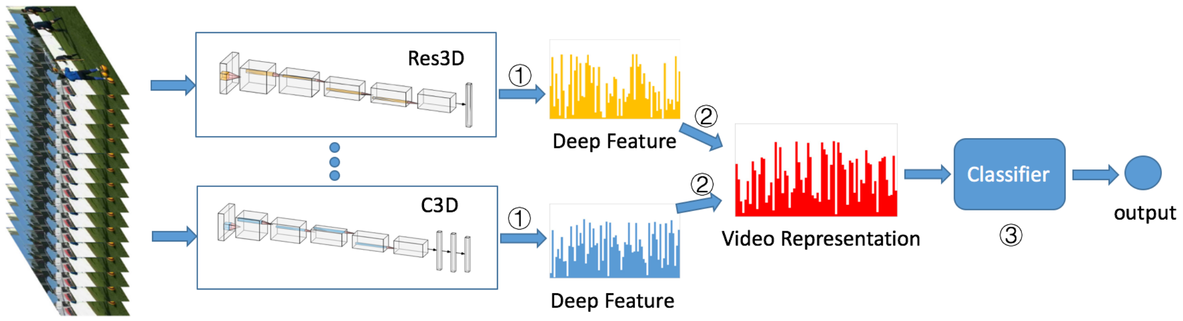
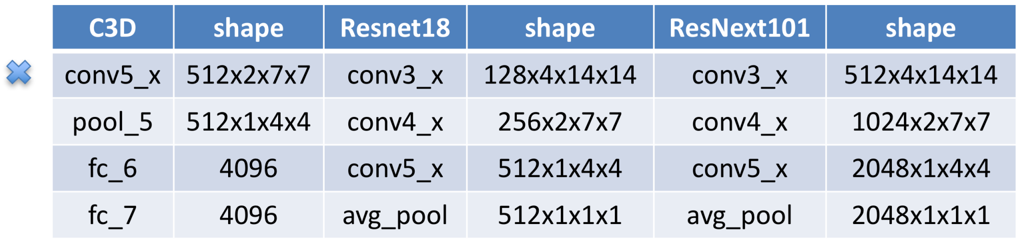


Figure 3.1: Overview of the Proposed Framework

the ConvNets should the features be extracted from? (2) What is the best feature fusion method? (3) What classifier to use in the second stage? The answers to these questions are crucial to our framework. Also, we could empirically find the answers by running control experiments. So in the following subsections, we describe the methodologies from the experimental perspective.

3.1 Exploring Layer Activation

There are two principles of extracting the layer activations. First, the dimension of the layer activation should not be too large. Because the layer activation will be used to train the classifier later, it is very difficult to train a classifier if the feature dimension of the sample is too large. Second, previous work has proved that the feature learned by the low layer is local. We are more interested in global features, so we test with middle or high layer activation in our experiments. We list the shape of layer activations of different 3D ConvNets models in Figure 3.2. Given the speed-accuracy tradeoff, we test with layers after the last convolutional layer for C3D, and layers after the third convolutional layer for ResNet-18 and ResNext-101.



C3D	shape	Resnet18	shape	ResNext101	shape
conv5_x	512x2x7x7	conv3_x	128x4x14x14	conv3_x	512x4x14x14
pool_5	512x1x4x4	conv4_x	256x2x7x7	conv4_x	1024x2x7x7
fc_6	4096	conv5_x	512x1x4x4	conv5_x	2048x1x4x4
fc_7	4096	avg_pool	512x1x1x1	avg_pool	2048x1x1x1

Figure 3.2: Shape of Layer Activations of Different 3D ConvNets Models

3.2 Exploring Feature Fusion

These are some straightforward feature fusion methods: element-wise addition and concatenation. Also, some more complex feature fusion methods are widely used in image classification and retrievals, such as Bag of words (BoW), Vector of Locally Aggregated Descriptors (VLAD) and Fisher Vectors (FV). So we test all of them to find the best one.

3.3 Exploring Classifier

Since the video representation is acquired through the aggregation of deep features, we assume that some non-deep classifier is enough in the second stage. So we only examine some non-deep classifiers, such as SVM, logistic regression (LR) and multinomial logistic regression (MLR), etc.

4. EXPERIMENTS

In Chapter 3, we describe several design choices of our framework, we empirically find the answers by running control experiments in this section.

4.1 Datasets

We evaluate our framework on three trimmed video classification benchmarks: Kinetics [28], UCF-101 [29] and HMDB-51 [30]. The large-scale Kinetics dataset is mainly used to train 3D ConvNets. The middle-size UCF-101 and HMDB-51 datasets are used to conduct the control experiments after the 3D ConvNets are trained. For a fair comparison, we follow the common evaluation scheme, that is, we run the experiments on three training/testing splits and report average accuracy over the three splits.

4.1.1 UCF-101

The UCF-101 [29] dataset consists of realistic action videos, collected from YouTube. It contains 101 action classes and 13320 video clips.

4.1.2 HMDB-51

The HMDB-51 [30] dataset is collected from various sources, mostly from movies. The dataset contains 6849 clips divided into 51 action categories, each containing a minimum of 101 clips.

4.1.3 Kinetics

Kinetics [28] dataset is a huge scope, top-notch dataset of URL connects to roughly 650,000 video cuts that spread 400 human activity classes, including human-object interactions such as playing instruments, as well as human-human interactions such as shaking hands and embracing. Every video is transiently cut and keeps going around 10 seconds. The quantity of training, validation, and testing sets are around 240000, 20000, 40000, separately.

4.2 What Architectures

In our framework, we only consider 3D ConvNets. To the best of our knowledge, C3D [5], Res3D [6, 8] and I3D [7] are the three most common reported 3D ConvNets for action recogntion task. For fair comparison, we also run experiments based on the three architectures.

4.2.1 Architectures Detail

4.2.1.1 C3D

C3D [5] is the 3D version of VggNet [31]. Table 4.1 lists the architecture details of C3D. We train the model on Kinetics dataset following the guidance of [5].

Table 4.1: C3D Architecture

layer name	**output size**	**C3D**
$conv\_1$	$16 \times 112 \times 112$	$3 \times 3 \times 3$, 64, stride $1 \times 1 \times 1$
$maxpool$	$16 \times 56 \times 56$	$1 \times 2 \times 2$ stride $1 \times 2 \times 2$
$conv2\_x$	$16 \times 56 \times 56$	$3 \times 3 \times 3, 128$
$maxpool$	$8 \times 28 \times 28$	$2 \times 2 \times 2$ stride $2 \times 2 \times 2$
$conv3\_x$	$8 \times 28 \times 28$	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix}$
$maxpool$	$4 \times 14 \times 14$	$2 \times 2 \times 2$ stride $2 \times 2 \times 2$
$conv4\_x$	$4 \times 14 \times 14$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix}$
$maxpool$	$2 \times 7 \times 7$	$2 \times 2 \times 2$ stride $2 \times 2 \times 2$
$conv5\_x$	$2 \times 7 \times 7$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix}$
$maxpool$	$1 \times 4 \times 4$	$2 \times 2 \times 2$ stride $2 \times 2 \times 2$

4.2.1.2 Res3D

Res3D [6, 8] is the 3D version of ResNet/ResNext [32, 33]. Table 4.2 lists the architecture details of Res3D. In our experiments, we use two variations of ResNet: ResNet-18 and ResNext-101. We train the model on Kinetics dataset following the guidance of [33], but the accuracies is slightly inferior to that reported in the paper.

Table 4.2: Res3D Architecture

layer name	output size	ResNet-18	ResNext-101
$conv\_1$	$16 \times 56 \times 56$	$7 \times 7 \times 7$, 64, stride $1 \times 2 \times 2$	
$maxpool$	$8 \times 28 \times 28$	$3 \times 3 \times 3$ stride $2 \times 2 \times 2$	
$conv2\_x$	$8 \times 28 \times 28$	$\begin{bmatrix} 3 \times 3 \times 3, 64 \\ 3 \times 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ 3 \times 3 \times 3, 128 \end{bmatrix} \times 3$
$conv3\_x$	$4 \times 14 \times 14$	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ 3 \times 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix} \times 4$
$conv4\_x$	$2 \times 7 \times 7$	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 23$
$conv5\_x$	$1 \times 4 \times 4$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 1024 \\ 3 \times 3 \times 3, 1024 \end{bmatrix} \times 3$
-	$1 \times 1 \times 1$	average pool, 101-d fc, softmax	

4.2.1.3 I3D

I3D [7] is the 3D version of GoogleNet [34]. Figure 4.1 shows the architecture details of I3D. We train the model on Kinetics dataset following the guidance of [7], but the accuracies is inferior to that reported in the paper.

4.2.2 Fine-tuned Accuracies on UCF-101 and HMBD-51

3D ConvNets trained on small datasets, such as UCF-101 and HMDB-51, proved to do not achieve high accuracy in previous work [8, 7], whereas those trained on big datasets, such as Kinetics, work well. So in our experiments, all the 3D ConvNets are trained on Kinetics [2], and then fine-tuned on UCF-101 and HMDB-51 as a baseline. Table 4.3 lists the top-1 accuracies of the fine-tuned models on UCF-101 and HMDB-51.

[1]reprinted from [7]

[2]some pre-trained model are provided by authors of the papers

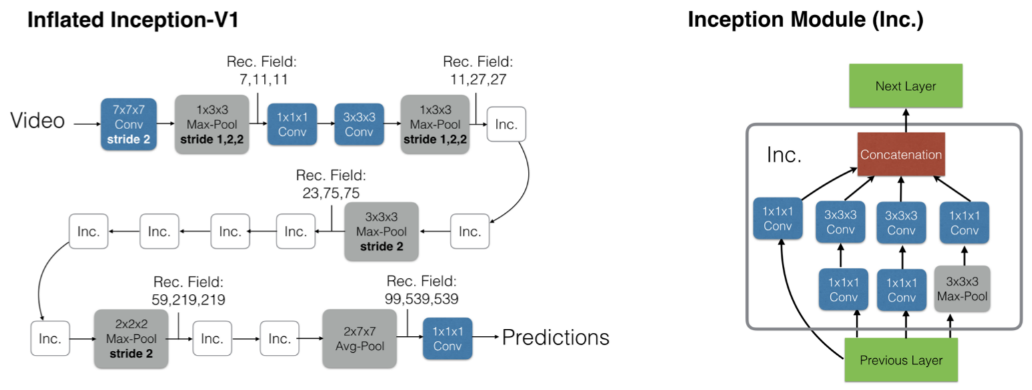


Figure 4.1: I3D Architecture[1]

Table 4.3: Fine-tuned Top-1 Accuracies on UCF-101 and HMDB-51

Model	**UCF-101**	**HMDB-51**
C3D	76.74	49.74
I3D	86.53	68.30
ResNet-18	84.06	54.12
ResNext-101-16f	90.14	64.05
ResNext-101-64f [3]	94.00	68.18

4.3 Which Layer

After training, the layer activations of 3D ConvNets could be used as generic video features/descriptors. Previous works [5, 6] utilized high convolution layer or fully-connected layer activation. In our experiment, we examine the effects of different layers activation of C3D, ResNet-18, and ResNext-101 on the UCF-101 dataset. More specifically, a video is split into several consecutive 16 frame long clips, and these clips are passed to 3D Convnets to extract layer activations. These clip layer activation of the same video are averaged to form the fixed-length video feature. Finally, the video feature is treated as input to linear SVM to classify the video. We list the accuracies of different layer activation as video descriptor in Table 4.4, Table 4.5 and Table 4.6. Here, we can see that the high convolutional/pool layer activations are better than low convolutional layer activations in terms of accuracy. Also, the higher layer activations generally have smaller feature dimensions, which make it make suitable for real applications. So considering the speed-accuracy tradeoff, we use the last pool layer activations as features in later experiments.

Table 4.4: Linear SVM Accuracies of C3D on UCF-101 and HMDB-51

Layer activation	**UCF-101**	**HMDB-51**	**Length**
$pool\_5$	78.98	48.89	8192
fc_6	77.50	48.82	4096
fc_7	74.07	46.14	4096

Table 4.5: Linear SVM Accuracies of ResNet-18 on UCF-101 and HMDB-51

Layer activation	**UCF-101**	**HMDB-51**	**Length**
$conv4\_x$	80.31	49.80	25088
$conv5\_x$	84.91	57.39	8192
avg_pool	83.98	56.99	512

Table 4.6: Linear SVM Accuracies of ResNext-101 on UCF-101 and HMDB-51

Layer activation	**UCF-101**	**HMDB-51**	**Length**
$conv4\_x$	88.55	60.46	100352
$conv5\_x$	88.92	61.18	32768
avg_pool	88.16	61.37	2048
$avg\_pool(f64)$	91.12	64.70	2048

4.4 Feature Fusion

In this section, we explore various feature fusion/encoding methods for the deep features learned by ConvNets. Specifically, we use the Kinetics trained models to produce features for the unseen videos of the UCF-101 and HMDB-51 datasets and produce the new fused feature using the feature fusion methods. We finally utilize the new fused feature to train a multi-class Linear SVM classifier for the classes of UCF-101 and HMDB-51 (using their training data) and evaluate on their test sets.

4.4.1 Feature Concatenation

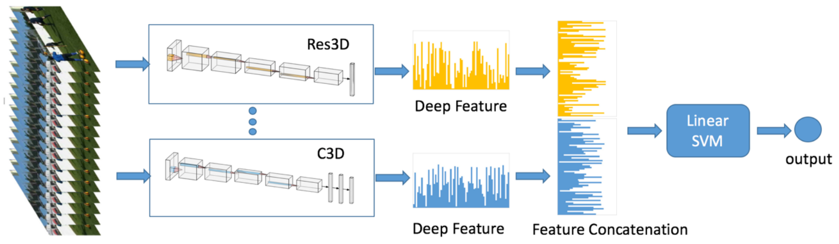


Figure 4.2: Feature Concatenation

Feature concatenation is one of the most straightforward feature fusion methods. In our experiments, as shown in Figure 4.2, we firstly extract the video feature of every single model, then concatenate the video features of different models to form the new video representation. Finally, we input the new video representation to a multi-class linear SVM for training models. For a fair comparison, we first list accuracies of the single model without feature fusion on UCF-101 and HMDB-51 datasets in Table 4.7. Table 4.8 lists the feature concatenation accuracies of two different models on UCF-101 and HMDB-51 datasets. Table 4.9 lists the feature concatenation accuracies of three or more different models on UCF-101 and HMDB-51 datasets. As shown in the table, C3D gets the lowest accuracy (77.50/48.82). But it (92.68/67.19) is superior to all other two models concatenation when combined with ResNet-101f64, which shows the features learned by different 3D ConvNets are complementary to each other. Notably, the four models concatenation achieves the best accuracy, which surpasses any single model by a large margin.

Table 4.7: Accuracies of Single Model on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**	**Length**
C3D	77.50	48.82	4096
I3D	81.21	55.03	4096
ResNet-18	83.98	56.99	512
ResNext-101	88.16	61.37	2048
ResNext-101f64	91.12	64.70	2048

Table 4.8: Feature Concatenation Accuracies of Two Models on UCF-101 and HMDB-51

models	**UCF-101**	**HMDB-51**	**Length**
models trained on 16-frame clips			
C3D+ResNet-18	87.58	59.67	4096+512
C3D+ResNext-101	90.88	62.16	4096+2048
ResNet-18+ResNext-101	89.14	61.50	512+2048
models trained on 64-frame clip			
I3D+ResNext-101f64	92.15	55.23	4096+2048
models trained on 16-frame or 64-frame clips			
C3D+I3D	84.51	58.69	4096+4096
C3D+ResNext-101f64	**92.68**	**67.19**	4096+2048
ResNext-101+I3D	90.56	63.53	4096+2048
ResNext-101+ResNext-101f64	92.10	66.60	2048+2048
ResNet-18+I3D	86.84	63.59	512+4096
ResNet-18+ResNext-101f64	91.59	66.73	512+2048

4.4.2 Feature Element-wise Addition

Feature element-wise addition, similar to feature concatenation, is one of the most common feature fusion methods. But different from feature concatenation, feature addition requires that the

Table 4.9: Feature Concatenation Accuracies of Multiple Models on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**	**Length**
C3D+ResNet-18+ResNext-101	90.88	63.40	4096+512+2048
C3D+ResNext-101+ResNext-101f64	93.28	67.71	4096+2048+2048
C3D+I3D+ResNext-101+ResNext-101f64	**95.27**	**67.91**	4096+7168+2048+204

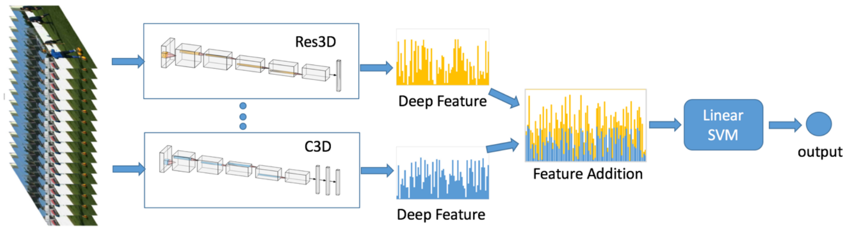


Figure 4.3: Feature Addition

Table 4.10: Feature Addition Accuracies on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**	**Length**
C3D	77.50	48.82	4096
I3D	81.21	55.03	4096
C3D+I3D	**84.27**	**58.24**	4096
ResNext-101	88.16	61.37	2048
ResNext-101f64	91.12	64.70	2048
ResNext-101+ResNext-101f64	**92.10**	**66.27**	2048

feature to be added must have the same dimension as shown in Figure 4.3. Table 4.10 lists the feature element-wise addition accuracies on UCF-101 and HMDB-51 dataset.

4.4.3 BOVW

Bag of Visual Words (BOVW) is commonly used in the image classification task. Its concept is adapted from bag of words (BOW) of information retrieval and natural language processing. In our experiments, as shown in Figure 4.4, we use the clip feature of videos to construct vocabularies and represent each video as a frequency histogram of features that are in the video. Different 3D

ConvNets are treated as different clip feature extractors. Table 4.11 lists the BOVW accuracies on UCF-101 and HMDB-51 datasets. Here, we can see that the BOVW of two models achieves the best accuracy.

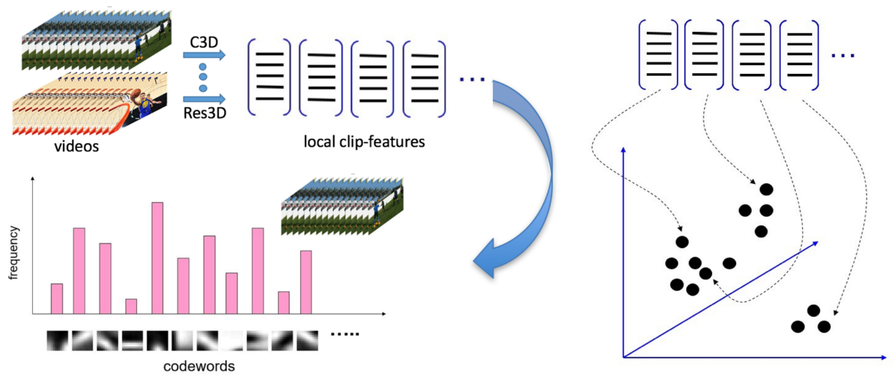


Figure 4.4: BOVW

Table 4.11: BOVW Accuracies on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**
ResNext-101	82.81	51.50
ResNext-101f64	83.58	55.49
ResNext-101+ResNext-101f64	**85.38**	**55.82**

4.4.4 VLAD

Vector of Locally Aggregated Descriptors (VLAD) [35] is an extension of BOVW. Different from BOVW which counts the number of local features associated with each feature cluster in a codebook, VLAD accumulates the residual of these local features corresponding to its assigned cluster. Table 4.12 lists the VLAD accuracies on UCF-101 and HMDB-51 datasets. As shown in the table, the VLAD of two models achieves the best accuracy.

Table 4.12: VLAD Accuracies on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**
ResNext-101	88.42	59.02
ResNext-101f64	90.91	61.96
ResNext-101+ResNext-101f64	**90.99**	**65.95**

4.4.5 FV

Fisher Vector (FV) [36, 16] is also an extension of BOVW. Different from BOVW, FV learns vocabulary with a Gaussian Mixture Model (GMM). More specifically, FV uses the likelihood a feature belongs to certain gaussian to measure the expectation of the distance between features and each Gaussian distribution, which is formalized as a feature vector. Then it concatenates the resulting vector for each Gaussian distribution into one large feature vector. Table 4.13 lists the FV accuracies on UCF-101 and HMDB-51 datasets. The FV of two models achieves the best accuracy.

Table 4.13: FV Accuracies on UCF-101 and HMDB-51

Models	**UCF-101**	**HMDB-51**
ResNext-101	85.28	59.02
ResNext-101f64	90.38	62.68
ResNext-101+ResNext-101f64	**91.67**	**64.12**

4.5 Classifier

4.5.1 Linear SVM

All results reported in previous sections use multi-class linear SVM as a classifier. Regard to linear SVM implementation, we use the SVM module of the scikit-learn library. For a fair comparison, we use all default parameter settings in all experiments. We also tried to tune the parameter to find better parameter settings and report the results of ResNext-101 (64f) in Tabel 4.14.

Table 4.14: Accuracies of Different Linear SVM Parameter Setting on UCF-101 and HMDB-51

C (Regularization)	UCF-101	HMDB-51
0.1	88.42	63.07
1	**91.12**	**64.71**
10	90.83	61.31
100	90.11	59.87

4.5.2 Logistic Regression (LR)

LR is a statistical model that uses a logistic function to represent a binary dependent variable. In our experiments, we use the multi-class version implemented by the scikit-learn library. We report the accuracies of logistic regression in Table 4.15. As shown in the table, linear SVM consistently performs better than LR.

Table 4.15: Accuracies of LR and Linear SVM on UCF-101

Model	Linear SVM	LR	Length
C3D+ResNet-18	87.58	86.52	4096+512
C3D+ResNext-101	90.88	89.80	4096+2048
ResNet-18+ResNext-101	89.14	87.97	512+2048
I3D+ResNext-101f64	92.15	92.97	4096+2048

4.5.3 Multilayer Perceptron (MLP)

Multilayer Perception (MLP) is another common classifier. In our experiments, we also tried replacing linear SVM with MLP and reported the accuracies in Table 4.17. As shown in the table, linear SVM consistently performs better than MLP. Table 4.16 lists the architecture details of MLP. We use categorical cross-entropy loss function and RMSprop optimizer to train the MLP model.

Table 4.16: MLP Architecture Details for UCF-101

Layer type	**Output shape**	**Param**
Dense	(None, 1024)	6292480
Dropout	(None, 1024)	0
Dense	(None, 101)	103525

Table 4.17: Accuracies of MLP and Linear SVM on UCF-101

Model	**Linear SVM**	**MLP**	**Length**
C3D+ResNet-18	87.58	78.44	4096+512
C3D+ResNext-101	90.88	84.33	4096+2048
ResNet-18+ResNext-101	89.14	86.72	512+2048
I3D+ResNext-101f64	92.15	91.39	4096+2048

4.6 Comparison

4.6.1 Model Ensemble

To prove the effectiveness of our framework, we report the accuracies comparison with the model ensemble on UCF-101 and HMDB-51 in Table 4.18. Voting is a widely-used ensemble method. So the accuracies report in the table uses the voting ensemble methods. More specifically, every single model predicts each test video and the final output prediction is the one that receives more than half of the votes (Majority Voting). As shown in the table, our method consistently performs much better than the model ensemble method.

Table 4.18: Accuracies Comparison with Model Ensemble on UCF-101 and HMDB-51

Models	**Ours**	**Model ensemble**
C3D+ResNet-18+ResNext-101	90.88/63.40	80.49/55.62
C3D+ResNext-101+ResNext-101f64	93.28/67.71	90.90/64.90
C3D+I3D+ResNext-101+ResNext-101f64	95.27/67.91	90.61/65.29

4.6.2 State-of-the-art

We show the comparison of our results with state-of-the-art methods in Table 4.19. As shown in the table, our method achieve higher accuracies than TDD [37], TSN [10] and P3D [38]. The two-stream I3D [7] achieve the best accuracies, which utilizes computationally expensive two-stream I3D architectures pre-trained on Kinetics. Our method achieves comparable accuracies to two-stream I3D on UCF-101 without using optical flow information.

Table 4.19: Top-1 Accuracies on UCF-101 and HMDB-51 Compared with the State-of-the-art Methods

Method	**UCF-101**	**HMDB-51**
Two-stream CNN [9]	88.0	59.4
TDD [37]	90.3	63.2
TSN [10]	94.2	69.4
P3D [38]	88.6	–
Two-stream I3D [7]	98.0	80.7
Ours	95.3	67.9

4.7 Discussion

We have presented the results of different feature fusion/encoding methods above. To verify why some feature fusion methods are better, we visualize the new fused feature using t-SNE [39]. More specifically, we extract and fuse features for each video, and those features are then projected to 2-dimensional space using t-SNE. We visualize the feature embedding of different models on

UCF-101 dataset below: C3D (77.50), I3D (81.21), ResNet-18 (83.98), ResNext-101 (88.16). As we can see from Figure 4.5, the better model has more separated feature embedding, which proves the model has learned more discriminative features. We also visualize the new fused feature embedding of ResNext-101s in Figure 4.6. The new fused feature embedding (c) is more separated than the raw feature (a and b). Notably, the new fused feature embedding of fine-tuned models (d) is incredibly separated.

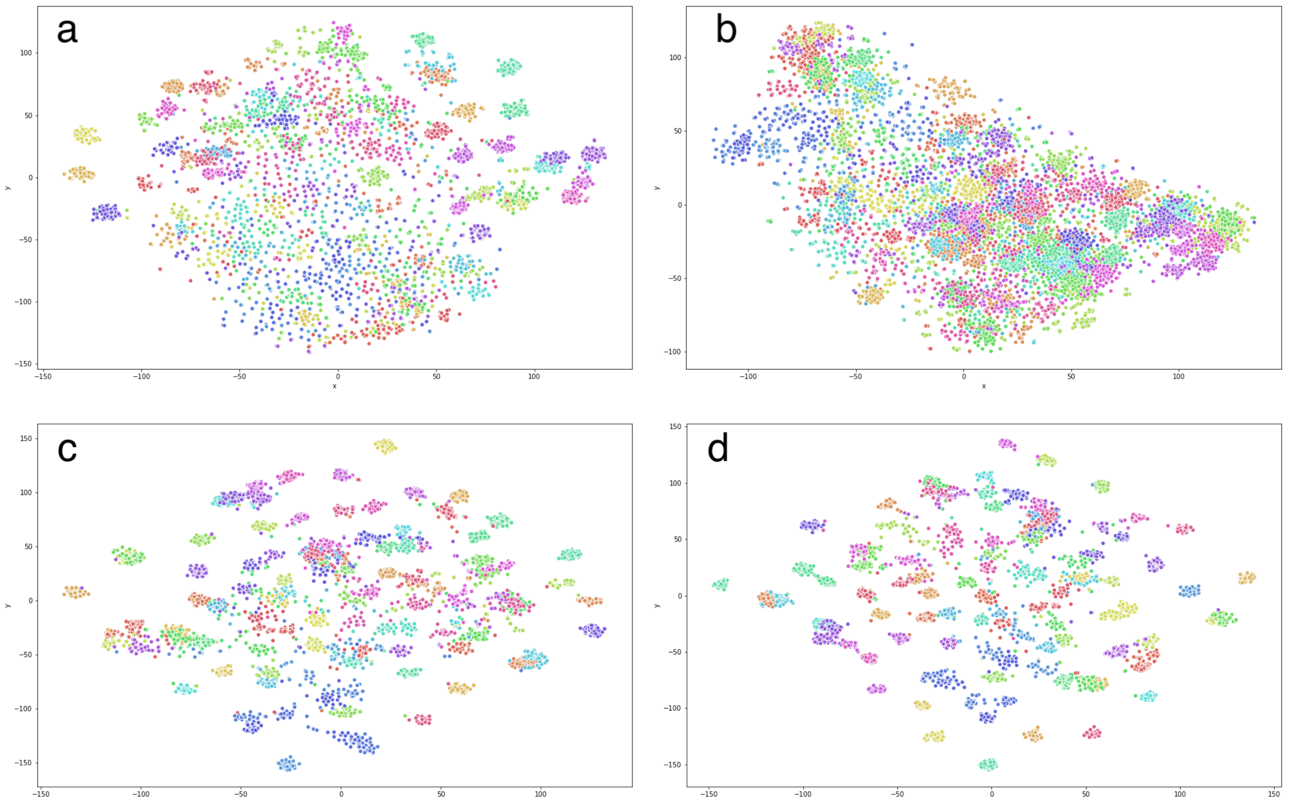


Figure 4.5: Feature Embedding of Models: (a). C3D (b). I3D (c). ResNet-18 (d). ResNext-101

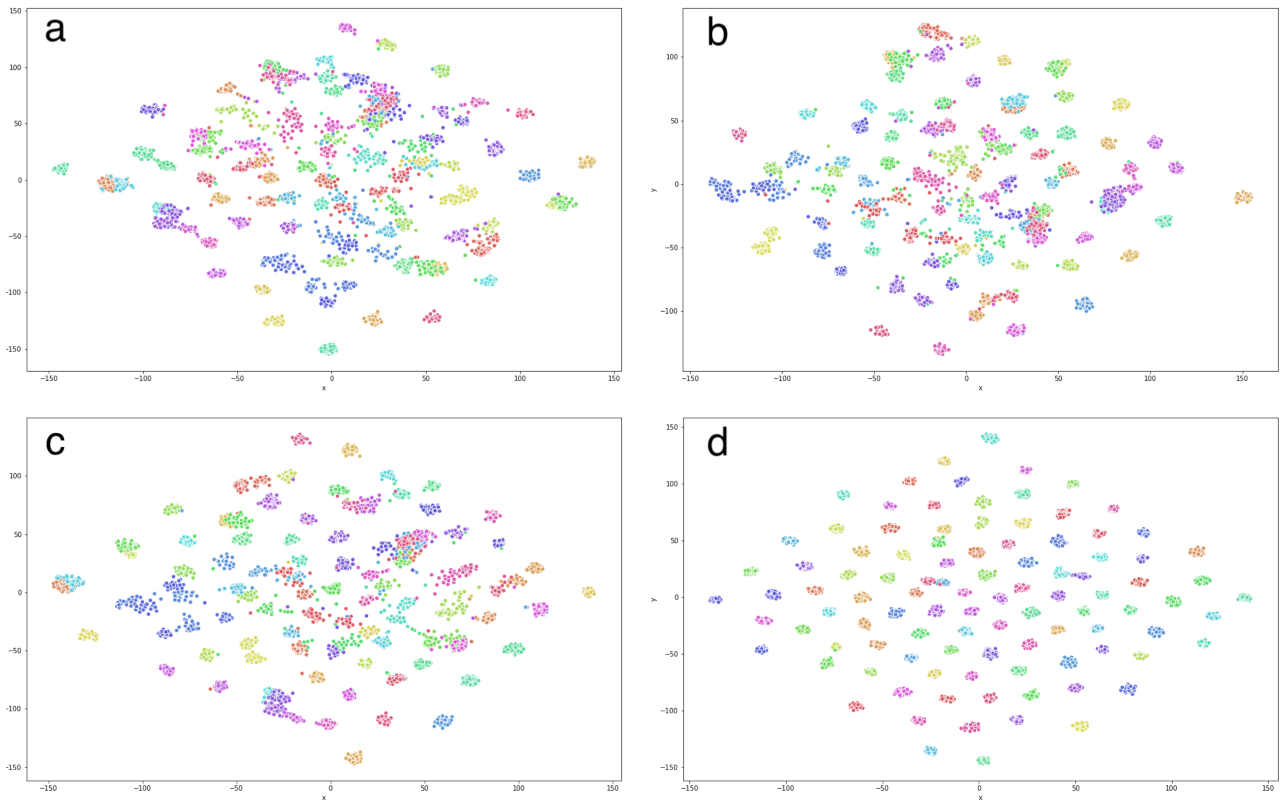


Figure 4.6: Feature Embedding of ResNets: (a). ResNext-101 (b). ResNext-101f64 (c). Feature Addition of ResNext-101 and ResNext-101f64 (d). Feature Addition of Fine-tuned ResNext-101 and ResNext-101f64

5. CONCLUSION

In this study, we explore various feature fusion/encoding of deep features learned by 3D ConvNets, such as C3D, I3D, and Res3D (ResNet and ResNext), for the video-based action recognition task. We empirically make good design choices for our framework by running lots of control experiments: (1). High layer activations are better than low layer activation. We mostly use the last convolutional layer or pooling layer activations as features. (2) Most feature fusion/encoding methods could boost the robustness and discrimination of features. Simple feature fusion methods, such as concatenation and addition, are better than complicated ones for deep features learned by 3D ConvNets. (3). Linear SVM is a better classifier than Logistic Regression and MLP in our case. By visualizing the feature embedding, we demonstrate that the new fused feature is more discriminative and robust. Finally, we compare our method with the model ensemble method (Majority Voting) to prove the effectiveness of our new framework. The new fused features with a linear classifier can outperform or approach current best methods on different video action recognition benchmarks. The improvement of recognition accuracy also proves that the new fused features are better spatio-temporal features for the video-based action recognition task.

REFERENCES

[1] A. Klaser, M. Marszałek, and C. Schmid, “A spatio-temporal descriptor based on 3d-gradients,” 2008.

[2] P. Scovanner, S. Ali, and M. Shah, “A 3-dimensional sift descriptor and its application to action recognition,” in *Proceedings of the 15th ACM international conference on Multimedia*, pp. 357–360, ACM, 2007.

[3] G. Willems, T. Tuytelaars, and L. Van Gool, “An efficient dense and scale-invariant spatio-temporal interest point detector,” in *European conference on computer vision*, pp. 650–663, Springer, 2008.

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 675–678, ACM, 2014.

[5] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

[6] D. Tran, J. Ray, Z. Shou, S.-F. Chang, and M. Paluri, “Convnet architecture search for spatiotemporal feature learning,” *arXiv preprint arXiv:1708.05038*, 2017.

[7] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.

[8] K. Hara, H. Kataoka, and Y. Satoh, “Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 6546–6555, 2018.

[9] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *Advances in neural information processing systems*, pp. 568–576, 2014.

[10] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, “Temporal segment networks: Towards good practices for deep action recognition,” in *European conference on computer vision*, pp. 20–36, Springer, 2016.

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

[12] J. Lin, C. Gan, and S. Han, “Tsm: Temporal shift module for efficient video understanding,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7083–7093, 2019.

[13] S. Ji, W. Xu, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[14] D. G. Lowe *et al.*, “Object recognition from local scale-invariant features.,” in *iccv*, vol. 99, pp. 1150–1157, 1999.

[15] J. Sivic and A. Zisserman, “Video google: A text retrieval approach to object matching in videos,” in *null*, p. 1470, IEEE, 2003.

[16] F. Perronnin and C. Dance, “Fisher kernels on visual vocabularies for image categorization,” in *2007 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2007.

[17] H. Jégou, M. Douze, C. Schmid, and P. Pérez, “Aggregating local descriptors into a compact image representation,” in *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*, pp. 3304–3311, IEEE Computer Society, 2010.

[18] J. Feng, Y. Liu, and L. Wu, “Bag of visual words model with deep spatial features for geographical scene classification,” *Computational intelligence and neuroscience*, vol. 2017, 2017.

[19] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *European conference on computer vision*, pp. 392–407, Springer, 2014.

[20] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 53–61, 2015.

[21] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giro-i Nieto, "Bags of local convolutional features for scalable instance search," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, pp. 327–331, ACM, 2016.

[22] J. Cao, Z. Huang, and H. T. Shen, "Local deep descriptors in bag-of-words for image retrieval," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pp. 52–58, ACM, 2017.

[23] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," 2008.

[24] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005.

[25] A. Diba, V. Sharma, and L. Van Gool, "Deep temporal linear encoding networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2329–2338, 2017.

[26] Z. Qiu, T. Yao, and T. Mei, "Deep quantization: Encoding convolutional activations with deep generative model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6759–6768, 2017.

[27] Z. Lan, Y. Zhu, A. G. Hauptmann, and S. Newsam, "Deep local video feature for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–7, 2017.

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

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

[30] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, “Hmdb: a large video database for human motion recognition,” in *2011 International Conference on Computer Vision*, pp. 2556–2563, IEEE, 2011.

[31] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[33] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[35] R. Arandjelovic and A. Zisserman, “All about vlad,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1578–1585, 2013.

[36] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, “Image classification with the fisher vector: Theory and practice,” *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[37] L. Wang, Y. Qiao, and X. Tang, “Action recognition with trajectory-pooled deep-convolutional descriptors,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4305–4314, 2015.

[38] Z. Qiu, T. Yao, and T. Mei, “Learning spatio-temporal representation with pseudo-3d residual networks,” in *proceedings of the IEEE International Conference on Computer Vision*, pp. 5533–5541, 2017.

[39] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.