Attention Clusters: Purely Attention Based Local Feature Integration for Video Classification

Xiang Long[1], Chuang Gan[2]*, Gerard de Melo[3], Jiajun Wu[4] , Xiao Liu[5] , Shilei Wen[5]

[1]Tsinghua University , [2]MIT-IBM Watson AI Lab , [3]Rutgers University , [4]MIT, [5]Baidu IDL

{longx13, ganc13}@mails.tsinghua.edu.cn, gdm@demelo.org

jiajunwu@mit.edu, {liuxiao12, wenshilei}@baidu.com

Abstract

Recently, substantial research effort has focused on how to apply CNNs or RNNs to better capture temporal patterns in videos, so as to improve the accuracy of video classification. In this paper, however, we show that temporal information, especially longer-term patterns, may not be necessary to achieve competitive results on common trimmed video classification datasets. We investigate the potential of a purely attention based local feature integration. Accounting for the characteristics of such features in video classification, we propose a local feature integration framework based on attention clusters, and introduce a shifting operation to capture more diverse signals. We carefully analyze and compare the effect of different attention mechanisms, cluster sizes, and the use of the shifting operation, and also investigate the combination of attention clusters for multimodal integration. We demonstrate the effectiveness of our framework on three real-world video classification datasets. Our model achieves competitive results across all of these. In particular, on the large-scale Kinetics dataset, our framework obtains an excellent single model accuracy of 79.4% in terms of the top-1 and 94.0% in terms of the top-5 accuracy on the validation set.

1. Introduction

Video classification remains one of the prime challenges in computer vision as well as machine learning. It has received a substantial amount of attention in recent years, owing not least to its numerous potential use cases, such as video tagging, surveillance, autonomous driving, and stock footage search. Thanks to recent large datasets, e.g. YouTube-8M [1] and Kinetics [4], the recognition accuracy in video classification has advanced considerably, although the current state-of-the-art remains subpar in comparison with human performance.
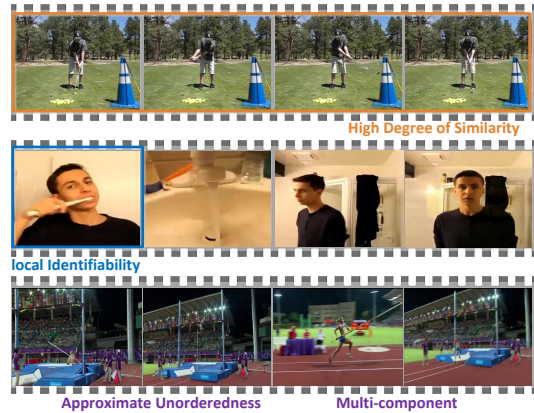
---

*Corresponding author.

Figure 1. Example of RGB frames sampled from videos. The bottom ones are shown in a randomly permuted temporal order. We observe several important characteristics of local features of a video: high degree of similarity, local identifiability, approximate unorderedness, and multi-component inputs.

Most of the existing effective video classification methods are based on convolutional neural networks (CNNs). CNNs have shown their powerful representation learning abilities in various image classification tasks. Convolutional and pooling layers together essentially act as potent feature extractors, which are able to mine local features from different regions of an image. Unsurprisingly, CNNs can also be used as a feature extractor for local feature extraction from videos, extracting a sequence of features for relevant video frames in accordance with their temporal order.

Many existing methods use convolutional neural networks (CNNs) or recurrent neural networks (RNNs) based on such local feature sequences to capture the temporal interactions within a video. The latter are particularly often considered for their ability to capture longer-term temporal patterns by retaining pertinent state information across time. In this paper, however, we cast some doubt on whether temporal patterns, especially long-term ones, are truly indispensable for common trimmed video classification tasks.

This is motivated by the following observations regarding the characteristics of such local features for video classification.

First, local features within a video tend to have a *high degree of similarity* across frames. In short video clips, the changes between RGB frames tend to be small. Especially during phases of slow movement, adjacent RGB frames exhibit substantial redundancy, sometimes differing in just minuscule ways. For instance, in the short example clip portraying *golf chipping* in Figure 1 (top), the frames are almost identical except for gradual changes in the position of the club. For classification, it may suffice to view these similar features holistically, packaged as a whole, while disregarding the particular details of their evolution over time.

Second, the local features of a video often possess the property of *local identifiability*. When people watch videos, they are frequently able to classify them based on just a few, occasionally even just a single frame. For example, in the video for *brushing teeth* in Figure 1 (middle), we can infer the class having observed just the initial frame. That is to say, even a tiny fraction of local features may single-handedly provide exhaustive classification information. For classification, the key is to spot the relevant local features of these most informative frames, without needing to ponder over their temporal patterns.

Third, the local features of a video may be regarded as *approximately unordered* in many classification settings. Of course, the input video itself is ordered, and a thorough semantic interpretation of the portrayed narrative requires some level of understanding of the temporal progression. For classification problems, however, we conjecture that the order may not be crucial. Even if the local features are permuted, a correct classification of the video may remain achievable. For example, in the *pole vault* video in Figure 1 (bottom), the frames have been reordered, showing first the landing, then the jump, and finally the run up. Yet, humans can still easily categorize it. Hence, the order of local features may not need to be preserved in video classification.

Accounting for the above considerations, we investigate an approach that completely abandons temporal cues. Instead, we explore the potential of purely attention based local feature integration methods to generate a global representation. This is because attention mechanisms naturally possess the following properties. First of all, attention outputs are essentially weighted averages, which implies that repeated local features will automatically be aggregated to reduce their redundancy. Secondly, an attention model may assign higher weights to significant local features so as to focus on a small number of key signals, and their local identifiability determines to what extent the classification results on a small number of key frames can be taken as a class label for the entire video. Finally, the inputs to an attention model are naturally unordered sets of varying sizes, which

fits the properties of the local features, and also facilitates a generalization ability to varying numbers of local features.

We also observe a further important property, the *multi-component* nature of local features of a video. Multiple cues in the signal may simultaneously make important contributions towards enabling the classification of a given video. For instance, in the *pole vault* video (Figure 1 bottom), the landing, jump, and run up all may yield useful information. Combining the signals across these different aspects ought to be better than focusing on just one of them.

A single attention unit can be viewed as focusing on just one aspect of the video, hence discarding a considerable amount of information. It turns out that it is near-impossible to achieve our aims with a single attention unit. Thus, we propose using multiple attention units to construct an *attention cluster* that constitutes a global representation of the video. Furthermore, we find that attention clusters resulting from a simple concatenation of the outputs of attention units only lead to weak gains, making them an inefficient choice. Instead, we propose a very simple and efficient procedure, the *shifting operation*, which effectively increases the diversity between attention units, speeding up the training efficiency and improving the classification accuracy.

In the following, we first review pertinent related work in Section 2. Then, in Section 3, we present our proposed attention clusters approach with the shifting operation, as well as our overall architecture for video classification. In Section 4, in order to analyze the effect of various attention cluster approaches and visualize the inner workings of the attention mechanism, we propose Flash–MNIST as a new toy dataset, and conduct various comparative experiments on it. Finally, we show the results of using attention clusters on challenging real-world video classification datasets in Section 5.

2. Related Work

2.1. Attention Mechanisms

Attention networks were originally proposed on the basis of the REINFORCE algorithm. In particular, Mnih et al. [20] and Ba et al. [2] proposed attention for object recognition with recurrent neural networks. These attention networks select regions by making hard binary choices, which may face difficulties in training.

Soft attention mechanisms were proposed by using weighted averages instead of hard selections. Bahdanau et al. [3] apply soft attention to machine translation with the aim of capturing soft alignments between source and target words. Sharma et al. [24] proposed a Soft-Attention LSTM model built on top of multi-layered RNNs to selectively focus on parts of the video frames and classify videos after taking a few glimpses. Wang et al. proposed an attention-based method for weakly supervised action

recognition [37], but it mainly targets untrimmed videos and is not ideal when directly applied to trimmed videos. Li et al. proposed an end-to-end sequence learning model called VideoLSTM [17], which hardwires convolutions in the Soft-Attention LSTM. These soft attention models require the introduction of supplementary sources of information to guide the weighted averages, which incur a substantial computational cost while failing to yield sufficient improvements in classification tasks.

To address the problem of attention on single sequences, many self-attentive models have been proposed for a variety of tasks, such as reading comprehension [5] and abstractive summarization [22]. Lin et al. [18] applied multiple attention units to learn task-independent sentence representations, relying on a penalization term to force each attention to attend to different parts. However, penalty functions forcing each weight vector to be different are too restrictive for video classification. Due to highly similar features between frames, many videos lack sufficient diversity, so this method fails to obtain good results. Our proposed attention clusters are another form of self-attentive architecture, which introduces a shifting operation to learn diversified attention units.

2.2. Video Classification

Since CNNs enjoy great success in image classification [15, 31, 26, 11], they have also been applied to video classification tasks. Karpathy et al. [14] studied multiple fusion methods based on pooling local spatio-temporal features extracted by 2D CNNs from RGB frames. This can be viewed as a preliminary exploration of the idea of integrating local feature sets, although simple pooling methods do not yield significant gains.

Many architectures have been proposed for modeling spatio-temporal information. The optical flow method [39] captures temporally local information by considering the variation in the surrounding frames. Simonyan et al. [25] devised a method that uses both RGB and stacked optical flow frames as appearance and motion signals, respectively. The accuracy is significantly boosted even by simply fusing probability scores, which indicates that optical flow can contribute useful short-term motion information. Feichtenhofer et al. [8, 7] compared a number of ways of fusing CNNs both spatially and temporally and combined them with ResNets [11] to extract better spatio-temporal information. C3D [33] extends 2D CNNs by using 3D convolution kernels to capture spatio-temporal information. Varol et al. [34] found that better results could be achieved by expanding the temporal length of inputs and using optical flows instead of RGB inputs for 3D CNNs. Carreira et al. [4] incorporated the Inception architecture [31] into 3D CNNs.

To model long-term temporal interactions in video classification, recurrent neural networks (RNN), particularly long short-term memory (LSTM) [13] have been applied in numerous papers. Ng et al. [21] devised two-stream LSTMs. Donahue et al. [6] proposed an end-to-end architecture based on LSTMs. Srivastava et al. [28] attempted to improve the representation ability of LSTMs by first pretraining them in an unsupervised manner to reconstruct the input. However, the accuracy on video classification with these RNN-based methods has been unsatisfactory, which may indicate that long-term temporal interactions are not crucial for existing video classification datasets. Our proposed method explores the potential of local feature integration without any recourse to long-term order information.

3. Approach

We now describe our approach of using attention clusters with a shifting operation, and show how to apply it to the task of video classification. We broadly consider three major parts: local feature extraction, local feature integration, and global feature classification. Each of these is addressed by suitable neural networks. Among them, the local feature extraction uses existing CNNs, and the global feature classification invokes fully connected and softmax layers. The main contribution lies in the local feature integration step, that is, our investigation of how to generate global representations given a set of local features.

3.1. Local Feature Set

In neural network settings, we often obtain local features of a video, since CNNs can naturally be used as a feature extractor.

The local feature set is defined as a set of unordered local features corresponding to different parts of the same video. Here, for convenience, we use a $M \times L$ matrix $\mathbf{X}$ to represent a set containing L local features, each column of which is a separate local feature vector $\mathbf{x}_i$:

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_L). \tag{1}$$

Note that, in fact, the set of local features is unordered, and hence permuting the columns of the matrix should not affect the results. Also, the number of local features L can vary across different objects. The challenge we seek to address at this point is how to generate fixed-length global vectors $\mathbf{g}$ to classify objects based on pertinent information from the local feature sets as given by $\mathbf{X}$.

3.2. Attention

We rely on an attention mechanism to obtain such global features. In classification settings, the attention is static, and the input contains only the local feature vector set itself. Its responsibility is to first analyze the importance of each local feature and then to bestow the global feature with as

much useful information as possible, while ignoring irrelevant signals and noise. Such attention outputs can essentially be regarded as weighted averages on a vector set:

$$\mathbf{v} = \mathbf{X}\mathbf{a}, \tag{2}$$

where $\mathbf{a}$ is a weight vector of dimension L, which is determined by a weighting function.

The choice of weighting function is the most crucial design decision to be made. Its input is the local feature set $\mathbf{X}$, while its output is the weight vector $\mathbf{a}$, whose ℓ_1 norm is 1. Each dimension of the weight vector corresponds to a local feature.

There are many methods to compute the weights of local features. For instance, global averages can be considered as a degenerate form of attention, and the corresponding weighting function can be expressed as:

$$\mathbf{a} = \frac{1}{L}\mathbf{1}, \tag{3}$$

where $\mathbf{1}$ is a vector of dimensionality L with all elements equal to 1. For a more malleable attention weighting function, we can use a single fully-connected layer that has only one cell (FC1), such as:

$$\mathbf{a} = \text{softmax}(\mathbf{w}^\mathsf{T}\mathbf{X} + \mathbf{b}), \tag{4}$$

where $\mathbf{w}$ and $\mathbf{b}$ are parameter vectors of dimensionality M and L, respectively. Similarly, we may use two successive fully-connected layers of size H and one hidden cell (FC2):

$$\mathbf{a} = \text{softmax}\left(\mathbf{w}_2^\mathsf{T} \tanh(\mathbf{W}_1^\mathsf{T}\mathbf{X} + \mathbf{b}_1) + \mathbf{b}_2\right), \tag{5}$$

where $\mathbf{W}_1$ is a parameter matrix of dimensionality $M \times H$, $\mathbf{b}_1$, $\mathbf{w}_2$ are parameter vectors of dimensionality H, and $\mathbf{b}_2$ are parameter vectors of size L.

In the experiments in Section 4, we compare the effects of these different weighting functions.

3.3. Attention Clusters

The output of one such attention unit typically focuses on a specific part of the video, e.g. a particular set of related frames or similar sounds. Normally, a single attention unit can only be expected to reflect one aspect of the video. However, there can be multiple pertinent parts in a video that together describe the overall event portrayed in the entire video. Therefore, to be able to represent multiple components, we need multiple attention units that focus on different parts of local features. We refer to a group of attention units that operate on the same input but have independent parameters as an *attention cluster*. The size N of an attention cluster is defined by the number of independent attention units in it. The global feature $\mathbf{g}$ resulting from an attention cluster is a vector of dimensionality NM which
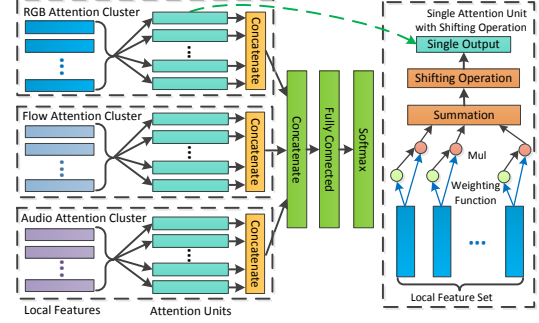

Figure 2. Multimodal Attention Clusters with Shifting Operation: The overall architecture for video classification. Separate attention clusters are applied for different feature sets and then the outputs are concatenated for classification.

can be obtained by concatenating the outputs of all involved attention units:

$$\mathbf{g} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N], \tag{6}$$

where $\mathbf{v}_k$ is output of k-th attention unit.

3.4. Shifting Operation

Although we expect attention clusters to be able to focus on different components, through experiments, which we describe in Section 4.4, we found that simply concatenating the outputs of attention units yields unsatisfactory results, since they tend to focus on similar signals. In order to address this problem, we propose a *shifting operation*, which is added onto each attention unit. This is achieved by adapting Eq. 2 as follows:

$$\mathbf{v} = \frac{\alpha \cdot \mathbf{X}\mathbf{a} + \beta}{\sqrt{N}\left\|\alpha \cdot \mathbf{X}\mathbf{a} + \beta\right\|_2}, \tag{7}$$

where α and β are learnable scalars, which act as a linear transformation in the feature space. After the linear transformation, we ℓ_2-normalize each attention unit separately. The factor $1/\sqrt{N}$ finally acts as a global ℓ_2-normalization on the cluster. Combining the linear transformation and normalization, the shifting operation shifts the weighted sum in the feature space and at the same time ensures scale-invariance. The shift operation efficiently enables different attention units to flexibly diverge from each other and have different distributions, and the scale-invariance facilitates the optimization of the entire network.

3.5. Overall Architecture for Video Classification

In order to collect multimodal information from videos, we extract a variety of different local feature sets, such as appearance (RGB), motion (flow), and audio signals. However, it is unrealistic to process all feature sets simultaneously within the same attention cluster, because features of

A: 0123456789 B: 0123456789 C: 0123456789 D: 0123456789 E: 0123456789

Figure 3. Frames of 5 videos in Flash-MNIST and the corresponding label. The **purple bold digits** in label are digits flashing in the video.

N	Average	Without Shifting		With Shifting	
		FC1	FC2	FC1	FC2
1	0.2	0.2	0.2	51.2	53.7
2	0.4	0.4	0.5	64.8	66.8
4	0.6	2.2	2.3	75.9	76.8
8	0.9	31.7	22.5	80.6	83.1
16	1.2	82.3	82.0	86.9	84.9
32	2.4	**83.3**	**83.2**	**87.1**	85.6
64	5.0	83.2	82.4	**87.1**	85.6
128	**8.9**	81.8	80.9	**87.1**	**85.7**

Table 1. Accuracy (%) on Flash-MNIST to show the effect of different weighting functions , various cluster sizes N, and aggregation with or without the shifting operation.

different modalities have different distributions, dimensionalities, and scales. Instead, we propose multimodal attention clusters with the shifting operation to train attention clusters for different modalities simultaneously. The layout of the proposed overall architecture is illustrated in Figure 2.

First, we extract multiple feature sets from the video. For each feature set, we apply independent attention clusters with shifting operation to obtain a modality-specific representation vector. Next, the output of all attention clusters are concatenated to form a global representation vector of the video. Finally, the global representation vector is used for classification through a fully-connected layer.

4. Analysis and Visualization

Because real-world video classification datasets conflate many different forms of variation, they are not easy to analyze and visualize directly. We first propose a new toy video classification dataset, Flash–MNIST, which we synthetically generate from the MNIST handwritten digit dataset. The Flash–MNIST dataset has fewer irrelevant factors of influence and requires only modest amount of computation. Nevertheless, its local feature set shares many properties with real video classification, which is convenient for analysis and visualization. This allows us to observe the behavior of the model under simplified conditions and achieve a deeper understanding of how the model works. We make our code for generating the dataset and training on it publicly available[1].

4.1. Flash–MNIST Dataset

In the well-known MNIST dataset, the goal is classify 28×28 pixel images of handwritten digits into 10 classes for the respective digits. Flash–MNIST extends this image classification task to video. The videos in Flash–MNIST consist of 25 frames with noisy backgrounds, on which various MNIST digits briefly flash up. The goal is to identify the specific set of digits that appear in the video, which entails choosing from a total of $2^{10} = 1024$ categories. Figure 3 shows 5 samples and their corresponding labels. Specifically, in order to generate training samples, we first randomly generate 25 different 28×28 noise frames, sample a

possible set of digits, and then randomly select corresponding digit images from the MNIST training set and overlay them on the random frames. For the test set samples, a similar process is used, except that images are selected from the MNIST test data. We randomly generate 102,400 samples for training, and 10,240 samples for testing.

We pretrain CNNs on MNIST with noisy backgrounds to extract local features. The CNNs consist of 2 successive convolutional layers with 5×5 kernels, 10/20 filters followed by *relu* activations and max-pooling with stride 2, and one fully connected layer with 50 hidden units. Through the CNNs, we can obtain 25 local features with a dimensionality of 50, each local feature corresponding to a frame in the video. These 25 local features consist of the local feature set, and we apply a variety of attention cluster alternatives, as described in Section 3, to induce a global representation. Finally, this is passed through one fully connected layer for classification. The accuracy scores for different settings for the attention clusters are given in Table 1. We describe and analyze the results in the following. For more details of the dataset generation and network training, please refer to the supplementary material.

4.2. Effect of Weighting Function

First of all, we analyze the effect of the choice of weighting function. We consider Average as described in Eq. 3, and two different attention weighting functions, FC1 as described in Eq. 4, and FC2 with 10 hidden units as described in Eq. 5. As shown in Table 1, we observe a significant gap between the results of using Average and the other attention weighting functions, which means that attention can play an effective role in this situation to focus on the parts that merit consideration. We also observe that FC2 fares slightly better than FC1 when the cluster size is small, but FC2 performs worse than FC1 when the cluster size is large. This, we speculate, may stem from the expressive power of attention clusters saturating as the size increases, even if the form of attention itself is simple enough. Considering that FC2 contains more parameters and requires more computation, but is unable to yield any benefits, we rely on the FC1 weighting function as the default in all subsequent experiments.
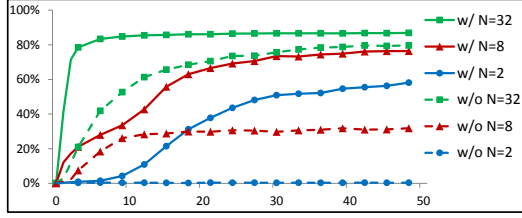
---

[1] https://github.com/longxiang92/Flash-MNIST

Figure 4. The accuracy on Flash–MNIST in each epoch, learned with different cluster sizes, with (w/) or without (w/o) the shifting operation.

4.3. Effect of Attention Cluster Size

Next, we consider the effect of different cluster sizes N. Because a video may include a variety of digits, these should not be attended to by the same unit. Consider an ideal situation, in which 10 attention units each pay attention to whether a specific digit occurs. This is obviously more reasonable than using a single attention function. Of course, during training, we lack control over which attention unit learns information about which digit. Still, we may hope that using multiple attention units may have the potential to learn more beneficial information.

In order to verify our idea, for fairness of comparison, we ensure that the numbers of network parameters are completely identical, except for parameters contained in the weighting functions, by also replicating the output vectors for the Average method N times. As shown in Table 1 and Figure 4, we find that with an increase in the cluster size N, the classification results increase significantly when the size is small and subsequently almost remain unchanged until reaching a certain level. Furthermore, the gap between using attention and Average becomes larger as N increases, indicating that this improvement is not due to an increase in the number of network parameters, but that the model genuinely pays attention to different aspects of local features.

Besides, the convergence speed also increases for increasing cluster sizes. Although a larger cluster size requires more computation, a smaller overall training time is required for reasonably large cluster sizes, since the computation of the attention and shifting operations is very efficient.

We visualize the attention weight maps for attention clusters with 8 units in Figure 5. We observe that each attention unit learns about different kinds of information. For instance, with shifting operation (middle, bottom), the first attention unit learns to attend to the digit 4, and the fifth attention unit learns to attend to digits 6 and 7.

4.4. Effect of Shifting Operation

Finally, we consider the effect of the shifting operation. As shown in Table 1, we find that applying attention clusters with shifting yields substantial improvements, as the
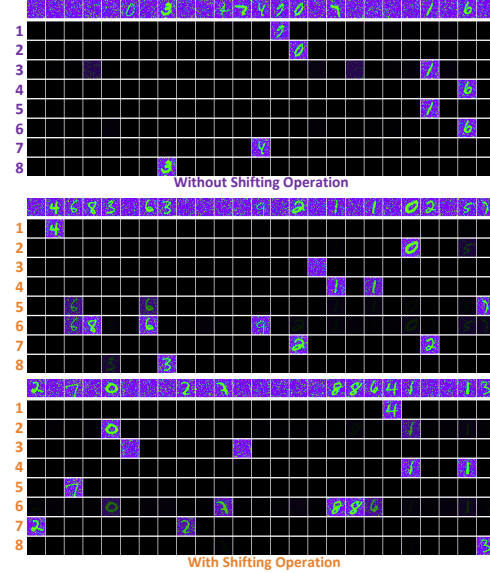

Figure 5. Visualization of attention weight maps of 8-unit attention clusters with shifting operation (middle, bottom), and without (top). We show frames in HSV space, in which larger values indicate a larger weight. The first row for each sample provides the video frames, and the respective following 8 rows correspond to the 8 weights of 8 attention units.

accuracy increases from 83.3% to 87.1%. Inspecting the attention weight maps in Figure 5, we find that the attention weights diverge entirely when using shifting, while the third and fourth attention weights match the fifth and sixth when not using the shifting operation. This indicates that the shifting operation can help us learn more diversified information for better generalization, and, ultimately, a higher accuracy. Simultaneously, the shifting operation can also help the model converge more rapidly. As plotted in Figure 4, at the same cluster size, the approach with shifting converges much more rapidly than when forgoing the shifting operation.

5. Experiment on Real Video Classification

In this section, we proceed to evaluate and compare our proposed methods on real-world video classification tasks.

5.1. Datasets

Specifically, we evaluate our methods on three popular trimmed video classification datasets.

UCF101 [27] contains 13,320 web video clips with heterogeneous forms of camera motion and illumination. Each clip contains about 180 frames and is labeled with one of 101 action classes, ranging from daily life activities to unusual sports.

HMDB51 [16] consists of 6,766 video clips from movies and web videos. Each clip is labeled with one of 51 action

N	RGB			Flow			Audio		
	Average	w/o	w/	Average	w/o	w/	Average	w/o	w/
1	72.5	73.1	73.4	63.5	63.7	65.2	20.6	21.3	21.5
2	73.0	73.6	73.9	64.5	64.6	66.1	21.4	22.3	22.3
4	**73.2**	73.8	74.2	65.0	65.0	66.5	22.2	22.8	22.9
8	**73.2**	**74.1**	74.4	65.4	65.5	66.8	**22.5**	**23.3**	23.4
16	73.1	74.0	74.6	65.2	**65.9**	67.1	22.4	**23.3**	23.7
32	73.0	73.6	74.7	64.9	65.5	67.4	22.3	23.1	23.9
64	72.5	73.2	74.9	64.4	65.1	**67.5**	22.2	22.7	24.1
128	72.6	73.2	**75.0**	65.1	63.9	**67.5**	21.8	22.5	**24.2**

Table 2. Top-1 accuracy (%) on Kinetics to show the effect of different cluster sizes and training with (w/) or without (w/o) shifting operation for RGB, flow, and audio.

N RGB	N Flow	N Audio	Top-1 (%)	Top-5 (%)
1	1	1	77.9	93.6
4	4	4	78.7	94.0
16	16	16	79.1	94.0
32	16	16	79.2	94.0
32	32	32	79.3	93.9
64	**32**	**32**	**79.4**	**94.0**
64	64	64	79.3	94.0
128	128	128	79.3	93.9

Table 3. Top-1 and top-5 accuracy (%) of multimodal integration of different cluster sizes for different modality on Kinetics.

categories. For UCF101 and HMDB51, we report the average accuracy over three training/testing splits, following the original evaluation scheme.

Kinetics [4] is a large-scale trimmed video dataset with more than 300K video clips in total, in which 246,535 serve as training data, and 19,907 for validation. Each video clip is taken from a different YouTube video and lasts around 10s. The clips are labeled using a set of 400 human action classes. We report experimental results on the validation split, due to the unavailability of the test split at the time of conducting these experiments. Since Kinetics is larger-scale and has more categories, it is convenient for stable and reliable experimental analysis. We mainly perform comparative experiments on Kinetics.

5.2. Local Feature Extraction

Considering that video is inherently multimodal, we extract three kinds of local features – RGB, flow, and audio – to represent the video. We rely on CNNs to extract these features.

RGB and flow features are extracted from RGB video frames or optical flow images, which are created by stacking the two channels for the horizontal and vertical vector fields [25]. For UCF101 and HMDB51, we initialize ResNet-152 [11] with a pre-trained ImageNet model and fine-tune it using the frames from training videos and then apply it to extract RGB and flow features. For Kinetics, we rely on Inception-ResNet-v2 [30] to extract these features. The RGB model is initialized with a pre-trained ImageNet model and fine-tuned using the training split based on the temporal segment network framework [38] with 7 segments. Then the flow model is initialized by the RGB model and also fine-tuned the same way. After training, we can extract local RGB and flow features for every frame.

To extract audio features, we generate audio spectrogram patches first. For every 10ms, we decompose the signal with a short-time Fourier transform and then rely on aggregated, logarithm-transformed 64 mel-spaced frequency bins following [12]. Each 96 consecutive bins yield one log-mel 96×64 spectrogram patch, which can be processed just like an image. After this, we can extract audio features using VGG-16 [26] on Kinetics just as for RGB and flow

features. For videos without audio, we feed in the average of audio features over the training set.

5.3. Local Feature Set Augmentation

Data augmentation plays a very important role in training neural networks, making use of properties of the data to effectively reduce overfitting. Here, we can similarly exploit the properties of the local feature sets to design new data enhancement methods. Local feature sets are approximately unordered, and most of the time, we do not need to understand the video using all of the local features, since we often only need a few key frames for classification. Hence, when we train the model, we can randomly sample a part of the features from the local feature set, but use all the features during testing. This data augmentation method can reduce the amount of computation during training, effectively prevent overfitting, and allow us to make use of all information during testing.

5.4. Implementation Details

In order to reduce overfitting, we apply dropout with probability 0.9 before the final fully connected layer. For local feature set augmentation, we sample 15/15/20 local features during training on UCF101/HMDB51/Kinetics, respectively, and we extract 20/20/25 local features, respectively, at equal intervals during testing. To balance the dataset, we set the sample weight to $1/S$ if a given sample belongs to a class that contains S samples during training. We rely on the RMSPROP algorithm [32] to update parameters with a learning rate of 0.001 and clip the gradient ℓ_2-norm of all parameters to 5 for better convergence.

5.5. Result of Single Modality

We explore how many attention units we need to use for a single modality and whether we should use the shifting operation or not to achieve the best results on Kinetics. Based on the previous experiment, we use the weighting function FC1 as default. Table 2 describes the relationship between the Top-1 accuracy of RGB, flow, and audio as single modalities, for different attention cluster sizes N, and with or without the shifting operation. We also show the Average results as the baseline.

We find that often with increasing N, the accuracy increases first and then decreases without shifting, or remains

	Method	Top-1(%)	Top-5(%)
	C3D [33]	55.6	79.1
	3D ResNet [10]	58.0	81.3
	Two-Stream I3D* [4]	74.2	91.3
	RGB+Flow TSN Inception V3 [38]	76.6	92.4
RGB	TSN[38]	73.0	90.9
	TS-LSTM (5 seg) [19]	73.2	90.9
	Temporal-Inception [19]	73.5	91.2
	Bi-directional LSTM	74.0	91.6
	Attention Cluster	**75.0**	**91.9**
Flow	TSN[38]	66.0	86.9
	TS-LSTM (5 seg) [19]	65.3	86.2
	Temporal-Inception [19]	65.4	86.2
	Bi-directional LSTM	66.4	86.9
	Attention Cluster	**67.5**	**87.3**
Audio	TSN[38]	21.6	39.4
	TS-LSTM (5 seg) [19]	22.6	40.6
	Temporal-Inception [19]	22.7	40.7
	Bi-directional LSTM	23.4	41.3
	Attention Cluster	**24.2**	**42.2**
RGB + Flow + Audio	Average	76.5	92.7
	Flatten	76.7	92.7
	TS-LSTM (5 seg) [19]	77.3	93.0
	Temporal-Inception [19]	77.6	93.3
	Bi-directional LSTM	78.2	93.5
	Attention Cluster	**79.4**	**94.0**

Table 4. Kinetics top-1 and top-5 accuracy (%) on the validation set, except for results marked with '∗', which were reported based on the test set.

Method	UCF101(%)	HMDB51 (%)
iDT + FV [35]	85.9	57.2
iDT + HSV [23]	87.9	61.1
EMV-CNN [40]	86.4	-
VideoLSTM[17]	89.2	-
FSTCN [29]	88.1	59.1
TDD+FV [36]	90.3	63.2
TSN (2 modalities) [38]	94.0	68.5
Two Stream [25]	88.0	59.4
Temporal-Inception [19]	93.9	67.5
TS-LSTM [19]	94.1	69.0
Fusion [8]	92.5	65.4
ST-ResNet [7]	93.4	66.4
ActionVLAD [9]	92.7	66.9
Attention Cluster RGB+Flow	**94.6**	**69.2**

Table 5. Mean classification accuracy (%) comparing with State-of-the-Art methods on UCF101 and HMDB51.

same local features (see supplementary material for details). As shown in Table 4, our approach improved over them by a large margin.

5.7. Comparison with State-of-the-Art

Finally, we compare our method against the state-of-the-art methods.

On UCF101 and HMDB, our approach obtains robust improvements over the two-stream fusion results for CNNs. As shown in Table 5, our approach can achieve competitive results in comparison with existing published methods [35, 23, 40, 25, 29, 17, 36, 8, 33, 38, 7, 9, 19].

On Kinetics, as shown in Table 4, we compare our method against many published results [33, 10, 4, 38]. Since the local feature extractors are trained using TSN [38], the results with CNNs are already very strong. The implemented three stream fusion methods also act as a strong baseline. Our approach again enjoys great improvements over all of them and obtains the start-of-the-art result.

6. Conclusion

To explore the potential of pure attention networks for video classification, a new architecture based on attention clusters with a shifting operation is proposed to integrate local feature sets. We analyze and visualize attention on the proposed Flash–MNIST data to get a better understanding of how our attention clusters work. We also have conducted experiments on three well-known video classification datasets and find that this architecture can achieve excellent results for a single modality or integrating multiple modalities, while also accelerating the training phase.

In terms of future work, we hope to apply this architecture to low-level local features and assess to what extent it can uncover relationships between features in different spatial coordinates. We further hope to integrate it into end-to-end-trained networks.

Acknowledgements

unchanged with the shifting operation. When N is small, due to limitations of the expressive power of attention models, the accuracy increases with increasing N both with and without the shifting operation. When N is sufficiently large, the expressive power is adequate. Without the shifting operation, for increases in N, the number of parameters also increases, and harmful overfitting becomes a serious issue, while the training is also more difficult. This leads to a decrease in accuracy. With the shifting operation, the training remains stable and reliable even for large cluster sizes.

We observe that the accuracy while using the shifting operation is universally better than without it. This suggests that the shifting operation can increase the diversity of the attention mechanism effectively, to improve the accuracy. We have also observed that attention clusters with shifting operation converged more quickly than without, similar to the observations on Flash–MNIST.

As shown in Table 4, comparing to the pretrained TSN models used for feature extraction, our attention clusters achieve excellent improvements of 2.0% for RGB, 1.5% for flow, and 2.6% for audio, in terms of top-1 accuracy. We also find that our results can beat other fusion methods using the same local features.

5.6. Result of Multimodal Integration

We investigate the effects of various combinations of different attention cluster sizes for multimodal integration. The results are shown in Table 3. We find that we can use smaller cluster sizes for multimodal integration rather than for a single modality. We can achieve the best top-1 accuracy (79.4%) and the best top-5 accuracy (94.0%) with 64 attention units for RGB, and 32 for flow and audio. We also implement a series of three stream fusion methods using the

References

[1] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. YouTube-8M: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 1

[2] J. Ba, V. Mnih, and K. Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014. 2

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 2

[4] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *arXiv preprint arXiv:1705.07750*, 2017. 1, 3, 7, 8

[5] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016. 3

[6] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015. 3

[7] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016. 3, 8

[8] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 3, 8

[9] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. ActionVLAD: Learning spatio-temporal aggregation for action classification. In *CVPR*, 2017. 8

[10] K. Hara, H. Kataoka, and Y. Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. *arXiv preprint arXiv:1708.07632*, 2017. 8

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 7

[12] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. Channing Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. CNN architectures for large-scale audio classification. *arXiv preprint arXiv:1609.09430*, 2016. 7

[13] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 3

[14] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 3

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 3

[16] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. In *ICCV*, 2011. 6

[17] Z. Li, E. Gavves, M. Jain, and C. G. M. Snoek. Videolstm convolves, attends and flows for action recognition. *arXiv preprint arXiv:1607.01794*, 2016. 3, 8

[18] Z. Lin, M. Feng, C. Nogueira dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017. 3

[19] C.-Y. Ma, M.-H. Chen, Z. Kira, and G. AlRegib. TS-LSTM and temporal-inception: Exploiting spatiotemporal dynamics for activity recognition. *arXiv preprint arXiv:1703.10667*, 2017. 8

[20] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *NIPS*, 2014. 2

[21] Y. H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015. 3

[22] R. Paulus, C. Xiong, and R. Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017. 3

[23] X. Peng, L. Wang, X. Wang, and Y. Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *Computer Vision and Image Understanding*, 150(C):109–125, 2016. 8

[24] S. Sharma, R. Kiros, and R. Salakhutdinov. Action recognition using visual attention. *arXiv preprint arXiv:1511.04119*, 2015. 2

[25] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. *NIPS*, 2014. 3, 7, 8

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 7

[27] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 6

[28] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015. 3

[29] L. Sun, K. Jia, D. Y. Yeung, and B. E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*, 2015. 8

[30] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, Inception-ResNet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016. 7

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 3

[32] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012. 7

[33] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *ICCV*, 2015. 3, 8

[34] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2017. 3

[35] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 8

[36] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015. 8

[37] L. Wang, Y. Xiong, D. Lin, and L. Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017. 3

[38] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 7, 8

[39] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. *Annual Symp. German Association Pattern Recognition*, 4713(5):214–223, 2007. 3

[40] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang. Real-time action recognition with enhanced motion vector cnns. In *CVPR*, 2016. 8