GCM: Efficient Video Recognition with Glance and Combine Module

Yichen Zhou[a], Ziyuan Huang[a], Xulei Yang*[,b], Marcelo Ang[a], Teck Khim Ng[a]

[a] *National University of Singapore (NUS), Singapore*
[b] *Institute for Infocomm Research (I2R), A*STAR, Singapore*

Abstract

In this work, we present an efficient and powerful building block for video action recognition, dubbed *Glance and Combine Module* (GCM). In order to obtain a broader perspective of the video features, GCM introduces an extra glancing operation with a larger receptive field over both the spatial and temporal dimensions, and combines features with different receptive fields for further processing. We show in our ablation studies that the proposed GCM is much more efficient than other forms of 3D spatio-temporal convolutional blocks. We build a series of GCM networks by stacking GCM repeatedly, and train them from scratch on the target datasets directly. On the Kinetics-400 dataset which focuses more on appearance rather than action, our GCM networks can achieve similar accuracy as others without pre-training on ImageNet. For the more action-centric recognition datasets such as Something-Something (V1 & V2) and Multi-Moments in Time, the GCM networks achieve state-of-the-art performance with less than two thirds the computational complexity of other models. With only 19.2 GFLOPs of computation, our GCMNet$_{15}$ can obtain 63.9% top-1 classification accuracy on Something-Something-V2 validation set under single-crop testing. On the fine-grained action recognition dataset FineGym, we beat the previous state-of-the-art accuracy achieved with 2-stream methods by more than 6% using only RGB input.

Key words: Glance and Combine Module, Video Action Recognition,
Spatio-Temporal Convolution, Action Recognition Datasets.

*Corresponding author: yangx@i2r.a-star.edu.sg

1. Introduction

Video action recognition is one of the fundamental research topics in computer vision. Since the introduction of ImageNet [1] and AlexNet [2], deep convolutional neural network (CNN) has become the essential tool to solve many computer vision tasks. Over the last decade, the efficiency of image classification models has been increased by around two orders of magnitude through the development of works such as ResNet [3] and MobileNet [4]. CNN has also been researched and utilized to solve the video action recognition task in the past few years. The early works [5, 6] started with using 2D CNN to extract features on both RGB frames and optical flow inputs. The subsequent works focused more on 3D CNN to learn spatio-temporal features for action classification [7, 8]. More recently, several works [9, 10, 11, 12] made efforts to modify existing 2D networks to improve their temporal modeling capability for action recognition.

Despite the great research progress, deep video models are still very heavy in terms of huge number of parameters and large computational cost. This makes it hard to deploy video recognition models to real world applications, especially those requiring real-time response under certain computational resource constraints. Therefore, improving the efficiency of video recognition models is an essential research problem to be studied.

In this work, we propose Glance and Combine Module (GCM), a highly efficient and powerful module for video action recognition. In GCM, for both spatial and temporal domains, we perform an extra glancing at a higher scale to get a broader perspective of the video features, then combine the spatio-temporal features obtained at different scales for further feature processing. A lightweight bottleneck structure is also included in GCM for further improvement in efficiency. We show in our carefully designed ablation studies that the proposed GCM is much more efficient than other versions of 3D spatio-temporal convolutional blocks.

By stacking GCMs in groups following the same manner as ResNet-50 [3], we build a series of GCM Networks with various complexity and spatio-temporal modeling power. Unlike most other related works that pre-train their models on ImageNet [1]

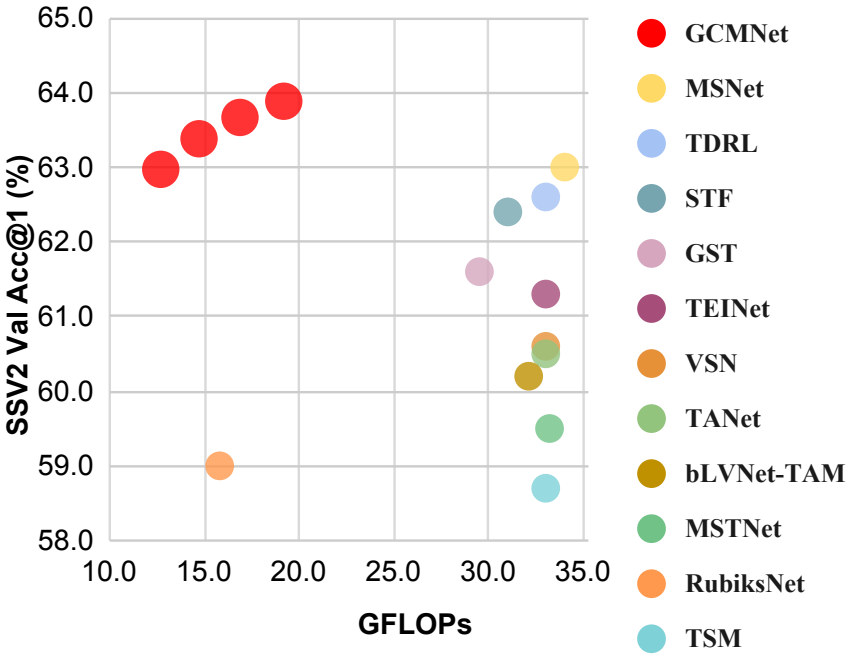


Figure 1: Comparison of GCMNets with state-of-the-arts models under 35 GFlops on Something-Something-V2

or Kinetics-400 [13], we train all our models directly on the target datasets and that demonstrates the great spatio-temporal modeling capabilities of GCM Networks. We evaluate the effectiveness and efficiency of GCM and GCM Networks by experimenting on several action recognition datasets including Something-Something (V1 [14] & V2 [15]), Multi-Moments in Time and [16] Kinetics-400 [13]. On the Kinetics-400 dataset which focuses more on appearance rather than action, our GCM network can achieve similar accuracy as others without pre-training on ImageNet. For the more action-centric recognition datasets such as Something-Something and Multi-Moments in Time, the GCM network achieves state-of-the-art performance with around two thirds the computational complexity of other recent methods. As can be seen from figure 1, our GCM Network outperforms other methods by a large margin while using

only a fraction of the computational cost. On the recently proposed fine-grained action recognition dataset FineGym, we beat the previous state-of-the-art accuracy achieved with 2-stream methods by more than 6% using only RGB input.

2. Related Works

2D CNNs in video action recognition. Before the introduction of 3D CNNs for video modelling, 2D CNNs were first used for action recognition by [17], where different naive temporal fusion strategies were explored. On top of this, [5] further introduces an extra network in parallel to extract motion information from optical flow features. Temporal segment networks [6] follow similar approach with improvements in spatio-temporal modeling ability. As it is usually expensive and troublesome to extract optical flow features, methods which does not require explicit optical flow features emerged. Spatiotemporal distilled dense-connectivity network (STDDCN) [18] implements both knowledge distillation and dense-connectivity to explore interaction strategies between appearance and motion streams along different hierarchies. Temporal Shift Module (TSM) [19] shifts some of the channels along the temporal axis to achieve information mixing among neighboring frames and that replaces the separate network for motion feature extraction. Other works [9, 10, 11, 12] further explored different ways of incorporating and enhancing temporal modeling into 2D networks. While some of these methods have been successfully proven to be effective, most of them still rely on 2D networks (ResNet in most works) pre-trained on ImageNet as a starting point and that limits their design space for greater efficiency.

3D CNNs in video action recognition. As there are 3 dimensions (2 spatial dimensions and 1 temporal dimension) for video data, spatio-temporal 3D CNN [7] naturally becomes a suitable tool for video recognition. I3D [8] introduces an effective way to inflate 2D networks to 3D networks so that the parameters learned during pre-training on ImageNet can be leveraged for better generalization. In the past few years, several works [20, 21] tried to decouple one single 3D convolution to separate spatial convolution and temporal convolution, and explored different arrangements of combining them for better efficiency. More recently, depthwise spatio-temporal 3D convolutions have

been utilized to increase the efficiency by reducing the computational complexity [22]. SlowFast networks [23] proposed to utilize two pathways for processing features at different frame rates, with lateral connections to fuse the features several times along the networks. While [24] proposed an attention based deformable module, taking into consideration the mutual correlations in both temporal and spatial domains, to effectively capture the long-range and long-distance dependencies in the video actions.

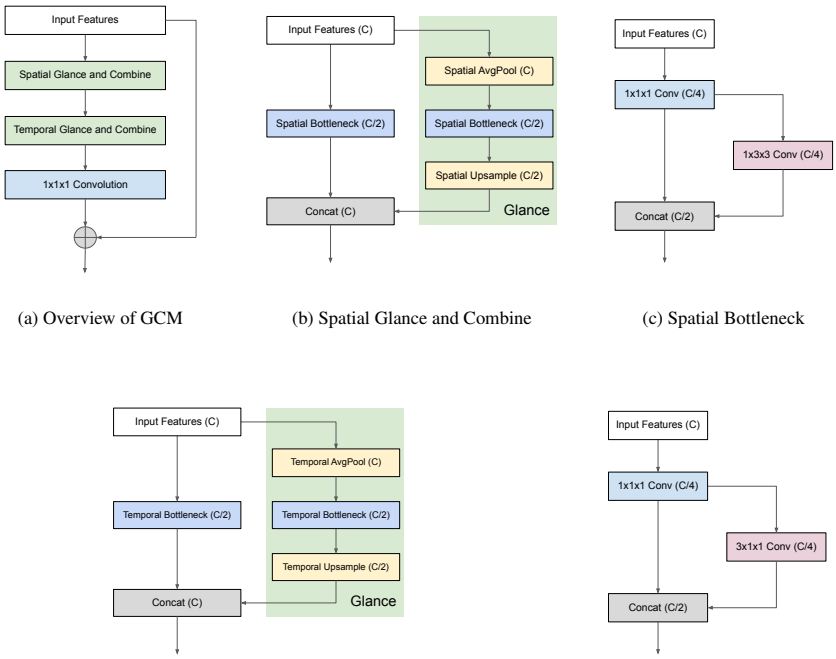
3. Methodology

The design of the Glance and Combine Module (GCM) is partly inspired by the SlowFast networks [23]. In SlowFast, two separate pathways (networks) are used to extract features at different frame rates and the features are fused after every stage containing several blocks of convolutions. We propose to extract features on two different scales at the local block level and combine them to accelerate information mixing in every block. Specifically, GCM processes features (glance) at a higher receptive level (lower frequency) and then immediately fuse (combine) with the features at the finer level. Furthermore, we extend GCM from temporal dimension to spatial dimensions and interleave them for better spatio-temporal representation learning by interacting with different spatio-temporal scales.

3.1. Glance and Combine Module (GCM)

Overview of GCM. As shown in figure 2a, GCM resembles the structure of a residual module [3]. We first apply a Spatial Glance and Combine block followed by a Temporal Glance and Combine block to process the features along the spatial domain and temporal domain respectively. A point-wise 1x1x1 convolution is then added to enhance the cross-correlation of spatio-temporal features. Finally, we apply element-wise addition with an identity pathway from the input to complete the GCM.

Design of the Glance and Combine block. Figure 2b shows the structure of the Spatial Glance and Combine (GC) block. The branch on the left is a normal Spatial Bottleneck that applies spatial convolution to process the features on the current scale. The branch on the right is the *Glance* branch that takes a glance of the features at the



(a) Overview of GCM (b) Spatial Glance and Combine (c) Spatial Bottleneck

(d) Temporal Glance and Combine (e) Temporal Bottleneck

Figure 2: **Glance and Combine Module** (Number in brackets indicates number of channels).

broader perspective and transform back. In the *Glance* branch, we apply a spatial average pooling to enlarge the receptive field and at the same time shrink the size of the feature map. We then apply the Spatial Bottleneck which has the same structure as the one on the left but with smaller input and output dimensions. A spatial (bilinear) upsampling operation is then used to upsample the feature map back to the original spatial dimensions. There is another way of realising the *Glance* branch. We can just apply a dilated convolution which can achieve the similar effect of extracting features with a larger receptive field. Our design is better in the sense that it would reduce the feature map size and therefore computational cost by first downsampling the features. We show in our ablation studies in Subsection 4.3 that our design indeed performs better than the dilation counterpart while using less computation. Finally, we combine the normal branch with the *Glance* branch by a concatenation operation. The Temporal GC block as shown in Figure 2d has the same design as the Spatial GC block except that all

the pooling, convolution and upsampling operations applied to the spatial dimensions are replaced with a temporal dimension.

Efficient bottleneck structure. To further improve the efficiency of our GCM, we design the bottleneck structure as shown in 2c. The main motivation of this design is to reduce the computations and encourage feature reuse [25] in an effort to increase efficiency. We firstly reduce the number of channels to a quarter of the original using a 1x1x1 convolution. We then apply the spatial convolution and finally concatenate with the output of the 1x1x1 convolution. The number of output channels is half of that of the input. That "bottleneck" allows us to control the number of channels when we combine two branches introduced in the previous paragraph. We later show in our ablation studies in Subsection 4.3 that this design of the bottleneck is more efficient than other similar and simpler design choices. Each convolution operation is followed by BatchNorm and ReLU following common practice. The corresponding Temporal Bottleneck has the same design except that we use a temporal convolution in place of the spatial convolution as shown in Figure 2e.

Spatial and Temporal Downsampling. In order to build a model for video recognition, we need downsampling in some of the GCMs within the network to shrink the dimensions of the feature map progressively. When downsampling is needed in GCM, the identity path is replaced by an average pooling followed by a 1x1x1 convolution. Moreover, the "normal" branch on the left of figure 2b and the spatial upsampling are removed, leaving only the average pooling followed by the bottleneck. The number of filters of the convolutions are proportionally increased such that the number of output channels of the GCM is doubled. It is a standard practice to spatially downsample the feature maps at the beginning of every stage. What differs in the various research work is in the temporal downsampling strategies. Several methods such as R(2+1)D [20] and CSN [22] apply temporal downsampling at the last three stages of the network while more recent works like SlowFast [23] and TSM [19] do not involve any temporal downsampling throughout the network. We empirically found that applying temporal downsampling only once achieves better speed-accuracy trade-off than the above strategies. The detailed experiment settings and results can be found in Subsection 4.3.

Stage	**Operation**	**Output channels**	**Output size**
input	-	3	$T \times H \times W$
conv_1	Conv 3×7^2	C	$T \times \frac{H}{2} \times \frac{W}{2}$
stage_2	GCM x 3	4C	$T \times \frac{H}{4} \times \frac{W}{4}$
stage_3	GCM x 4	8C	$T \times \frac{H}{8} \times \frac{W}{8}$
stage_4	GCM x 6	16C	$T \times \frac{H}{16} \times \frac{W}{16}$
stage_5	GCM x 3	32C	$\frac{T}{2} \times \frac{H}{32} \times \frac{W}{32}$
pool	AvgPool	32C	$1 \times 1 \times 1$
fc	FC	# classes	-

Table 1: **Structure of GCM Networks with width C**. We apply 4 stages of GCMs one by one, the number of GCMs in each stage is (3,4,6,3) which follows the configuration from ResNet-50. The first Glance and Combine Module in each stage is responsible for reducing the spatial (and temporal for stage_5) dimensions by half and doubling the number of output channels.

3.2. GCM Networks

With the efficient GCM as the building block, we can now construct a GCM Network by simply stacking GCMs repeatedly in groups. The grouping and stacking pattern follows the structure of ResNet-50 [3] where we half the spatial dimensions and double the number of channels at the start of every stage. We also downsample the temporal dimension at the last stage of the network. Moreover, we introduce a width factor C which indicates the number of output channels of the first convolution. Instead of setting a fixed value for C, we can vary the value of C to make a trade-off between speed and accuracy, creating a series of GCM networks. We denote a GCM network with width C by GCMNet_C. The structure is specified in table 1.

4. Experiments

4.1. Datasets

Something-Something-V1 (SSV1) [14] is a large collection of labeled video clips that shows humans performing pre-defined basic actions with everyday objects. The total number of action categories is 174. **Something-Something-V2 (SSV2)** [15] doubled the number of videos, improved the annotation quality and increased the resolution of videos compare to V1. These two versions of the dataset are known to be temporal-sensitive in the way that they require understanding the action more than the appearance of background or objects. Hence they are more suitable for assessing the spatio-temporal modeling capabilities. In particular, we use SSV2 as the main dataset for our ablation studies.

FineGym [26] is a recently proposed dataset for fine-grained action recognition which is built with gymnastic videos. Comparing with other datasets, it contains finely defined labels for sub-actions of a set of gymnastic events. The actions that belong to the same gymnastic event often only have very subtle differences and usually have very similar background and camera view angle. These difficulties pose great challenges to existing action recognition models.

Multi-Moments in Time (MMiT) [16] is a recently published large-scale and multi-label video recognition dataset. It contains around 1.02 million videos with 2.01 million labels from 313 action categories. All the videos are trimmed to 3 seconds long. Similar to the Something-Something datasets, MMiT also places more emphasis on temporal-relationships for determining the action rather than visual appearance.

Kinetics-400 [13] is a large-scale video action recognition dataset which contains 400 human action categories. Each category has at least 400 video clips. All the clips are from YouTube videos and are trimmed to 10 seconds in duration. For Kinetics-400, the appearance of the background contributes greatly to the process of identifying the action category in the video.

4.2. Experiment Settings

The following is the common training and testing setup for most of the experiments that will be presented. Additional specific dataset related settings (if any) are described

in the corresponding subsections. For all the experiments, we only take RGB channels of the video frames as input. No optical flow information is used as it is costly to obtain and will double the complexity of the whole model.

Training. Each model is trained for a total of 64 epochs on 8 GPUs, with 8 samples on each GPU making the total batch-size 64. The base learning rate is 0.01 per GPU. A cosine learning rate schedule is used with the first 4 epochs as the warm-up period. All models are trained from scratch and are optimized using Stochastic Gradient Descent with momentum of 0.9 and weight decay of 5e-5. We also include Dropout with rate 0.5 before the classification layer to avoid overfitting. Following the sparse segment-based sampling method [6], we uniformly divide the video into 32 segments, and randomly sample one frame from each segment to form a video clip of 32 frames. For the spatial sampling, we first resize the height to a number uniformly sampled from the range of 192 to 224 while keeping the original aspect ratio. We then randomly crop a rectangular region of height 192 and width 256 from the resized frames as the input to the network during training. The input size to the GCM Networks in terms of $T \times H \times W$ is thus $32 \times 192 \times 256$.

Testing. For inference, we select the middle frame from each of the 32 uniformly divided segments. All frames are re-scaled such that the height is equal to 192. We then perform one single center crop to obtain a region of 192×256 from the resized frames.

4.3. Ablation Studies

In this section, we conduct comprehensive experiments to compare the performance of the proposed Glance and Combine block and the bottleneck structure with various alternatives. All the ablation studies are carried out on Something-Something-V2. For all the structures used for comparison in figure 3 and figure 4, we only show the versions for the spatial domain for simplicity. The corresponding temporal version of each structure can be constructed easily by changing the operations on the spatial axes to the temporal axis whenever applicable.

Temporal Downsampling. In order to identify a suitable and efficient temporal sub-sampling strategy, we perform a simple set of experiments by applying temporal

Width	num of Temp. DS.	num of output frames	GFLOPs	SSV2 acc(%)
14	0	32	17.9	63.42
14	1	16	16.9	**63.66**
14	2	8	13.6	60.53
14	3	4	10.0	58.23
12	0	32	13.5	62.80
12	1	16	12.7	**62.97**

Table 2: **Experiments on temporal downsampling**

Structure		W	FLOPs (10^9)	SSV2 acc%	Acc Δ%
GCM	Both S.T. GC & Concatenation	12	12.7	62.97	
		13	14.7	63.38	
		14	16.9	63.66	
		15	19.2	63.88	
Ablate GC	No GC (baseline)	12	18.9	62.08	-1.8
	Only Spatial GC	12	14.6	62.28	-1.1
	Only Temporal GC	12	17.0	62.50	-1.2
	Both w. dilation	12	14.9	62.47	-0.9
Ablate Concat	No 1x1x1 (baseline)	14	19.6	62.18	-1.7
	Addition	14	13.8	62.03	-1.2
	Sequential	14	13.8	61.90	-1.3
Simple baseline	"R(2+1)D"	12	13.3	60.62	-2.4
	"R3D"	12	16.5	60.89	-2.7

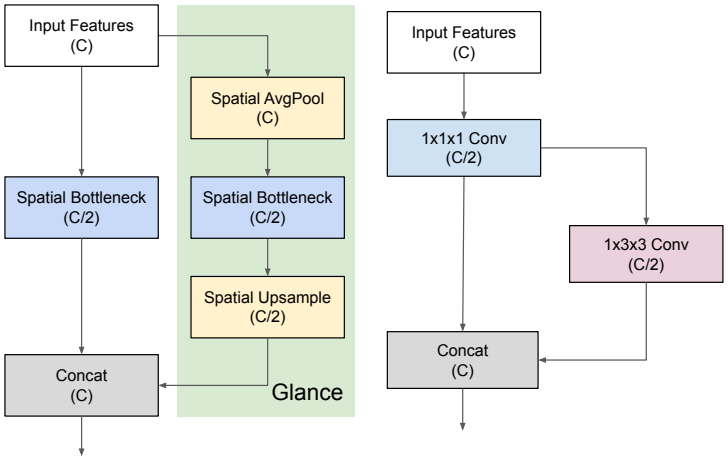
Table 3: **Ablation experiments of GCM.** The "Acc Δ%" indicate the approx. change in accuracy of ablated structure compared to the corresponding proposed GCM Network with similar computational cost.

downsampling at the beginning of the last (0, 1, 2, 3) stages in the GCM Networks. The results of top-1 validation accuracy on Something-Something-V2 are shown in

table 2. For models with the same width C=14 in which we apply one single temporal downsampling, the accuracy increases from 63.42% to 63.66% while requiring less computation. If we apply another temporal downsampling in the second last stage, the accuracy drops significantly to 60.53%. We further run experiments of models with width C=12 such that we can perform a fair comparison under similar computation cost below 14 GFlops (shown in the bottom of table 2). The results indicate that the performance drop when having more than one temporal downsampling is not due to the low computation complexity, it is rather because of the lack of temporal details to make accurate predictions. Therefore, we choose to only apply temporal sampling once at the beginning of the last stage of the GCM Networks.

Effectiveness of Glance and Combine. In our proposed Glance and Combine Module, we perform Glance and Combine along both spatial and temporal dimensions. To verify the effectiveness of each of them, we experimented with replacing the spatial and/or temporal Glance and Combine with a simple single-branch structure shown in figure 3b. The simple spatial block without Glance and Combine contains one spatial bottleneck where the number of channels after each convolution is doubled to compensate for the missing glance branch. The results are shown in table 3. As we can see, the model can only obtain an accuracy of 62.08% without any Glance and Combine (GC) on spatial and temporal dimensions. By applying GC on either spatial or temporal dimension, we can reduce the computation complexity and increase the accuracy at the same time to 62.28% and 62.50% respectively. When we apply spatial and temporal GC, we further decrease the FLOPs and improve the accuracy to 62.97%. The improvement brought about by using GCM instead of a single-branch structure at 19 GFLOPs of computation is around 1.8% (from 62.08% to 63.88%), all factors being equal.

Design of the bottleneck structure. Here we test a few alternative designs of the bottleneck structure that might be used in other related works. The structure that we proposed in GCM fuses the output of the spatial 1x3x3 convolution with the 1x1x1 convolution with concatenation (figure 4a). We explored changing the fusing operation to addition (figure 4c), or removing the fusing operation making the bottleneck purely sequential (figure 4b). The simplest variant where there is no 1x1x1 convolution (fig-



(a) Proposed Spatial GC

(b) Simple spatial block without GC

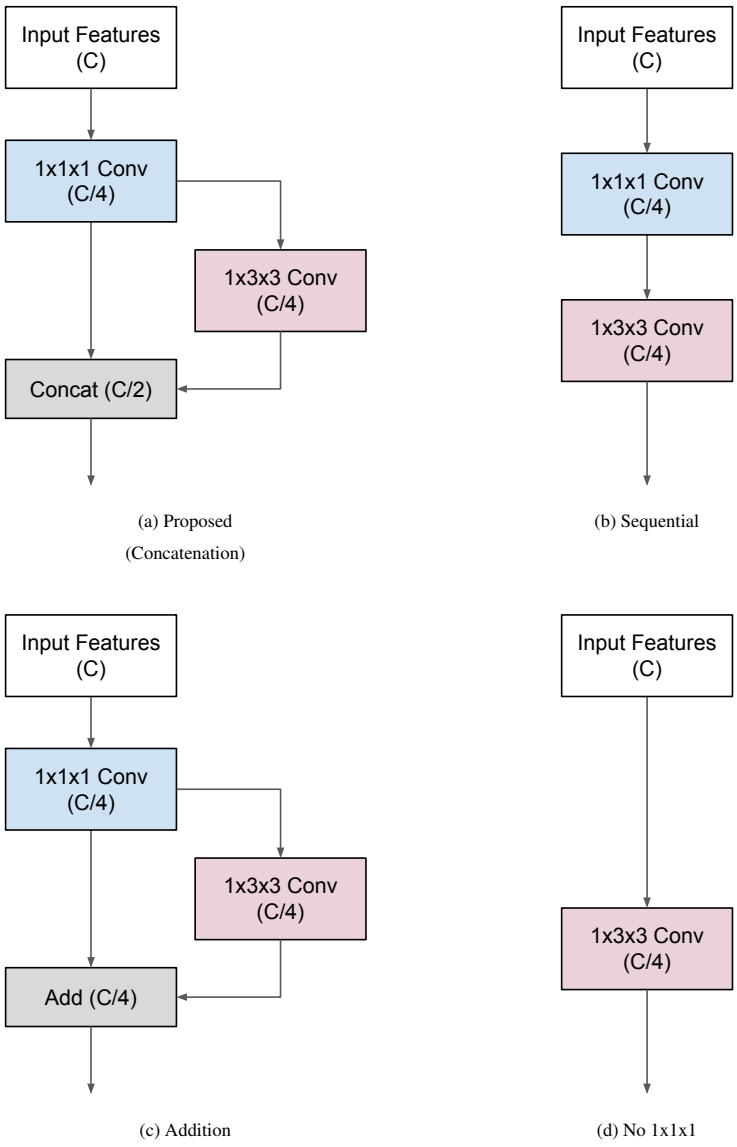
(c) Spatial GC with Dilated Spatial Bottleneck

(d) Dilated Spatial Bottleneck

Figure 3: Structures for ablation of GCM (Number in brackets indicates number of channels)

ure 4d) is also included. As we can see from table 3, all the 3 alternative designs cause the accuracy to drop by a non-trivial amount as compared to our proposed design. At 19.2 GFLOPs of computational budget, our design increases the accuracy by 1.7% to 63.88% compared to 62.18% achieved by the naive baseline design (figure 4d).

Comparing with existing simple structures. By removing both the *GC* and the *Concat* design, the module is conceptually the same as R(2+1)D where spatial and



(a) Proposed (Concatenation)

(b) Sequential

(c) Addition

(d) No 1x1x1

Figure 4: Structures for ablation of bottleneck (Number in brackets indicates number of channels)

temporal convolutions are applied separately. Furthermore, it can be reduced to R3D by fusing the spatial and temporal convolutions. With the number of output channels fixed at C, the detailed structure of the tested modules are as follows: (number in brackets indicates # of channels)

Model	Pre	Param (10^6)	FLOPs (10^9)	SSV1 acc%	SSV2 acc%
TIN [9]	Ki	24.3	34	45.8	-
TSM [19]	IN	24.3	33	45.6	58.7
STM [27]	IN	24.0	33.3	47.5	-
bLVNet-TAM [28]	IN	40.2	32.1	47.8	60.2
TANet [29]	IN	30.4	33	46.5	60.5
VSN [11]	IN	24.3	33	46.6	60.6
TEINet [10]	IN	30.4	33	47.4	61.3
GST [30]	IN	21.0	29.5	47.0	61.6
TEA [31]	IN	-	35	48.9	-
TSM+W3 [32]	IN	-	33.5	49.0	-
STF [33]	IN	21.4	31	50.2	62.4
TDRL [34]	IN	-	33	49.8	62.6
MSNet [35]	IN	24.6	34	**50.9**	63.0
GCMNet_{12}	-	**2.89**	**12.7**	49.6	63.0
GCMNet_{13}	-	3.38	14.7	50.1	63.4
GCMNet_{14}	-	3.92	16.9	50.6	63.7
GCMNet_{15}	-	**4.49**	19.2	**50.9**	**63.9**

Table 4: **Comparison of GCMNets with state-of-the-art RGB models under 35 GFLOPs on SSV1 & SSV2.** Note that some results on SSV2 are missing as the single-crop performance were not reported in the referenced work.

- "R(2+1)D": 3×1^2 (C/4) $\Rightarrow 1 \times 3^2$ (C/4) $\Rightarrow 1^3$ (C)
- "R3D": 1^3 (C/4) $\Rightarrow 3^3$ (C/4) $\Rightarrow 1^3$ (C)

As shown in the last two rows of table 3, the performance further dropped after we change the building block to simpler structures.

4.4. Results on Something-Something (V1 & V2)

In this section, we conduct experiments on Something-Something-V1 [14] and Something-Something-V2 [15] datasets. We perform experiments using a series of

GCM Networks with different widths C, and compare with other state-of-the-art methods under 35 GFLOPs of computation.

Table 4 shows the performance comparison of our GCM Networks with other recently proposed state-of-the-art methods on the validation set of SSV1 and SSV2 with single-crop testing using only RGB frames as input. Note that all other models in the table are pre-trained on either ImageNet or Kinetics400, while GCM Networks are **trained from scratch** directly on SSV1 and SSV2. Among all the "small" version (under 35 GFLOPs) of other methods, the state-of-the-art accuracy for SSV1 and SSV2 were 50.9% and 63.0% respectively by [35]. Our GCMNet$_{15}$ with only 4.49 million parameters and 19.2 GFLOPs is able to match the state-of-the-art accuracy with 50.9% on SSV1, and set a new state-of-the-art with 63.9% for SSV2. On SSV2, despite using only $2/3$ of FLOPS as the MSNet [35] model which achieved 63.0% top-1 accuracy, GCMNet$_{15}$ outperforms it by a non-trivial amount of 0.9%.

Model	Modality	Gym288		Gym99	
		Mean	Top-1	Mean	Top-1
TSN [6]	RGB	26.5	68.3	61.4	74.8
	RGB+Flow	37.6	79.9	76.4	86.0
I3D [13]	RGB	28.2	66.1	64.4	75.6
TRN [36]	RGB	33.1	73.7	68.7	79.9
	RGB+Flow	42.9	81.6	79.8	87.4
TRNms [36]	RGB	32.0	73.1	68.8	79.5
	RGB+Flow	43.3	82.0	80.2	87.8
TSM [19]	RGB	34.8	73.5	70.6	80.4
	RGB+Flow	46.5	83.1	81.2	88.4
GCMNet$_{12}$	RGB	**53.9**	**86.7**	**88.0**	**91.5**
*GCMNet$_{12}$	RGB	**59.7**	**88.4**	**88.6**	**92.2**

* trained with v1.1 of FineGym.

Table 5: **Comparison of GCMNet with state-of-the-arts on FineGym.** The accuracy numbers for all the other methods are directly referenced from [26].

.

4.5. Fine-grained Action Recognition on FineGym

FineGym [26] is a recently proposed dataset for fine-grained action recognition built with gymnastic videos. There are two evaluation metrics used for the fine-grained action classification tasks. The first is the standard *Top-1 accuracy* which is the fraction of correctly predicted samples over the total number of samples in the validation set. However, due to the unbalanced distribution, it might overfit to the more frequent action classes. Therefore, the *Mean accuracy* is proposed for a less biased view of the performance by calculating the average of per-class accuracy.

Same as the other datasets, we train our GCM Networks on the training set directly from scratch with the same protocol introduced in Subsection 4.2 without using any special techniques for fine-grained recognition. Table 5 shows the results of our GCM Networks compared to other existing methods on *Gym288* and *Gym99*. The specific computational complexities for the other methods were not explicitly mentioned in [26] but we estimate that they are at the level of at least around 50 GFLOPs for single modality (assuming ResNet-50 as backbone, and an input size of $12 \times 224 \times 224$). Our GCMNet$_{12}$ only costs 12.7 GFLOPs of computation which is less than 30% of the complexity of others. On *Gym288*, we improve the state-of-the-art mean accuracy from 46.5% obtained by the two-stream TSM [19] to 53.9% with only RGB input. Similarly, we improve the state-of-the-art mean accuracy on *Gym99* from 81.2% to 88.0%. The authors of FineGym suggested that optical flow information and temporal dynamics are critical in achieving good results on these fine-grained action recognition tasks. Despite this, GCM Networks can still achieve **more than 6% absolute increase** in mean accuracy compared to two-stream models (and **more than 17%** compared to models using only RGB input). This shows that our GCM Networks possess great spatial-temporal modelling ability.

Furthermore, the authors of FineGym have introduced an updated version 1.1 for the dataset with more samples in the training set. We therefore also perform experiments by training on the updated v1.1 train set, and test on the same validation set. The results are listed in the last row of table 5 as a reference for future works. As expected, the mean accuracy for *Gym288* and *Gym99* are further improved to 59.7% and 88.6%, respectively.

Model	Pre-train	Param	FLOPs	mAP
TSN [6]	Kinetics	~43M	~40G	58.92
TSM [19]	Kinetics	~43M	~63G	61.06
TIN [9]	Kinetics	~43M	~64G	62.22
GCMNet_{18}	-	**6.5M**	**20.3G**	**62.86**

Table 6: **Comparison of GCMNet with state-of-the-arts on Multi-Moments in Time.** The mAP numbers for other methods are referenced from the Temporal Interlacing Network [9], while the parameters and FLOPs are estimated based on the information about the ResNet-101 backbone.

4.6. Results on Multi-Moments in Time and Kinetics-400

In this section, we further verify the efficiency of our GCM Networks on two large scale video recognition datasets. When training on Multi-Moments in Time and Kinetics-400, a dropout rate of 0.2 is used during training as both datasets contain large number of categories and video clips. We randomly crop a 192x192 region from the resized frames of which the shorter side is randomly sampled in range [192, 224] for training. Again, we train our GCM Networks from scratch directly on these two target datasets unlike most other methods which rely on pre-training.

For the experiment on Multi-Moments in Time, we evaluate the mean Average Precision on the validation set under single-crop testing and compare with other methods. Due to time and resource constraints, we train and test one GCM Network with a width of 18. As we can see from table 6, the GCM Network achieves higher mAP at 62.86%, using only 20.3 GFLOPs of computation which is less than 40% of the other methods.

When experimenting with the Kinetics-400 dataset, we change the temporal sampling method as the raw videos are longer (around 10 seconds). We uniformly sample 32 frames from a trimmed clip formed by 128 consecutive raw frames from the original video. We randomly sample one trimmed clip from a video during training, and uniformly sample 4 such clips for testing. The GCM Networks are trained for a total of 128 epochs from scratch on Kinetics-400. Table 7 shows the comparison of our method to other recently proposed methods using only RGB inputs. Despite not having

Model	Pre	Param (10^6)	*total inference GFLOPs	top-1 (%)
bLVNet-TAM [28]	IN	25.5	841=93.4x9	73.5
TSM [19]	IN	24.8	650=65x10	74.7
TEINet [10]	IN	31	990=33x30	74.9
TEA [31]	IN		1050=35x30	74.7
VSN [11]	IN	43.4	630=63x10	75.4
TDRL [34]	IN		990=33x30	75.7
MSNet [35]	IN	25.1	670=67x10	76.4
TANet [29]	IN	25.6	1032=86x12	**76.9**
ir-CSN [22]	-	22.1	141=14.1x10	71.3
ip-CSN [22]	-	24.5	159=15.9x10	71.8
STF [33]	-		508=254x2	72.5
X3D-S [37]	-	**3.8**	81.8=2.72x30	73.5
SlowFast [23]	-	34.4	1083=36.1x30	75.6
X3D-M [37]	-	**3.8**	185=6.18x30	76.2
GCMNet_{18}	-	6.6	**81.0**=20.25x4	73.5
+GCMNet_{20}	-	8.1	176=43.9x4	75.8

*(total GFLOPs)=(GFLOPs for single view)x(num of views)

+using 256×256 as the spatial input size to the network

Table 7: **Comparison of GCM Networks with other methods on Kinetics-400.**

pre-training on ImageNet, our GCM Networks still achieve comparable results as other models while requiring small amount of computations and parameters.

4.7. Runtime Analysis

In addition to low theoretical computational cost, it is more important for a video model to have low latency and high throughput for real world applications. We perform runtime analysis of GCMNet_{15} on a single NVIDIA GeForce 2080 Ti GPU. In order to compare with other related works, we also perform measurements for TSM under the same protocol as mentioned in [19]. Specifically, the batch size is set as 1 and 16

Model	P100		2080 Ti		SSV2
	Lat	Thpt	Lat	Thpt	acc%
VSN [11]	28.7	47.2			61.6
TEINet [10]	49.5	24.2			62.1
TSM [19]	29.0	39.5	23.4	50.6	61.0
GCMNet_{15}			**22.1**	**348**	**63.9**

Table 8: **Comparison of latency(ms) and throughput(videos/sec) of GCMNet with other methods**.

for measuring latency and throughput respectively. Our measurements are performed using PyTorch-1.6.0 with CUDA-10.1. The results are shown in table 8. Though the latency of GCMNet_{15} is only slightly shorter than that of TSM, our GCMNet_{15} is able to achieve $7\times$ throughput at 348 video clips per second. Most of the other related works [29, 31, 32, 34, 35] use the same ResNet-50 backbone as TSM, which means they are expected to have similar latency and throughput. The results indicate that GCM Networks can better utilize GPU resources and achieve much higher throughput.

5. Conclusion

In this work, an efficient Glance and Combine Module for video action recognition is introduced. In GCM, we glance spatial and temporal features at a higher and broader level before combining with features at the original scale. We build a series of networks using the proposed GCM as building block. We achieve new state-of-the-art performance on several large scale video action classification datasets while consuming much less computational cost compared to related works. The results show that mixing spatial and temporal features at different scale can boost the spatio-temporal representation power.

We further demonstrate that by constructing 3D networks with compact and powerful design, they can be trained directly on the target video datasets to match or even surpass other models which rely on pre-training on ImageNet or Kinetics. Our work shows that pre-training on large scale datasets is not necessary for achieving good per-

formance on relatively small datasets. Therefore more 3D neural network structures can be explored and proposed instead of only tweaking pre-trained 2D backbones. We hope this work can motivate further research on designing efficient 3D network structures specifically for video action recognition.

It is also worthy to mention that most of the advances on 2D/3D networks are orthogonal to our work, and therefore can be combined to achieve better performance. For examples, the temporal attention module in TA3DNet [38], may be combined with GCM networks to further improve the performance by temporally assigning different importance to each frame. We will explore this direction in our future work.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

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

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861.

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

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

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

[8] J. Carreira, 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, 2017, pp. 6299–6308.

[9] H. Shao, S. Qian, Y. Liu, Temporal interlacing network, arXiv preprint arXiv:2001.06499.

[10] Z. Liu, D. Luo, Y. Wang, L. Wang, Y. Tai, C. Wang, J. Li, F. Huang, T. Lu, Teinet: Towards an efficient architecture for video recognition, arXiv preprint arXiv:1911.09435.

[11] P. Ma, Y. Zhou, Y. Lu, W. Zhang, Learning efficient video representation with video shuffle networks, arXiv preprint arXiv:1911.11319.

[12] Z. Huang, S. Zhang, L. Pan, Z. Qing, M. Tang, Z. Liu, M. H. Ang Jr, Tada! temporally-adaptive convolutions for video understanding, in: ICLR, 2022.

[13] 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.

[14] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al., The" something something" video database for learning and evaluating visual common sense., in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Vol. 1, 2017, p. 5.

[15] F. Mahdisoltani, G. Berger, W. Gharbieh, D. Fleet, R. Memisevic, On the effectiveness of task granularity for transfer learning, arXiv preprint arXiv:1804.09235.

[16] M. Monfort, K. Ramakrishnan, A. Andonian, B. A. McNamara, A. Lascelles, B. Pan, D. Gutfreund, R. Feris, A. Oliva, Multi-moments in time: Learning and interpreting models for multi-action video understanding, arXiv preprint arXiv:1911.00232.

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

[18] W. Hao, Z. Zhang, Spatiotemporal distilled dense-connectivity network for video action recognition, Pattern Recognition 92 (2019) 13–24.

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

[20] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, M. Paluri, A closer look at spatiotemporal convolutions for action recognition, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 6450–6459.

[21] S. Xie, C. Sun, J. Huang, Z. Tu, K. Murphy, Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 305–321.

[22] D. Tran, H. Wang, L. Torresani, M. Feiszli, Video classification with channel-separated convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 5552–5561.

[23] C. Feichtenhofer, H. Fan, J. Malik, K. He, Slowfast networks for video recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6202–6211.

[24] J. Li, X. Liu, M. Zhang, D. Wang, Spatio-temporal deformable 3d convnets with attention for action recognition, Pattern Recognition 98 (2020) 107037.

[25] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, Ghostnet: More features from cheap operations, arXiv preprint arXiv:1911.11907.

[26] D. Shao, Y. Zhao, B. Dai, D. Lin, Finegym: A hierarchical video dataset for fine-grained action understanding, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[27] B. Jiang, M. Wang, W. Gan, W. Wu, J. Yan, Stm: Spatiotemporal and motion encoding for action recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 2000–2009.

[28] Q. Fan, C.-F. R. Chen, H. Kuehne, M. Pistoia, D. Cox, More is less: Learning efficient video representations by big-little network and depthwise temporal aggregation, in: Advances in Neural Information Processing Systems, 2019, pp. 2261–2270.

[29] Z. Liu, L. Wang, W. Wu, C. Qian, T. Lu, Tam: Temporal adaptive module for video recognition, arXiv preprint arXiv:2005.06803.

[30] C. Luo, A. L. Yuille, Grouped spatial-temporal aggregation for efficient action recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 5512–5521.

[31] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, L. Wang, Tea: Temporal excitation and aggregation for action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 909–918.

[32] J.-M. Perez-Rua, B. Martinez, X. Zhu, A. Toisoul, V. Escorcia, T. Xiang, Knowing what, where and when to look: Efficient video action modeling with attention, arXiv preprint arXiv:2004.01278.

[33] Y. Zhou, X. Sun, C. Luo, Z.-J. Zha, W. Zeng, Spatiotemporal fusion in 3d cnns: A probabilistic view, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9829–9838.

[34] J. Weng, D. Luo, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, X. Jiang, J. Yuan, Temporal distinct representation learning for action recognition, arXiv preprint arXiv:2007.07626.

[35] H. Kwon, M. Kim, S. Kwak, M. Cho, Motionsqueeze: Neural motion feature learning for video understanding, in: European Conference on Computer Vision, Springer, 2020, pp. 345–362.

[36] B. Zhou, A. Andonian, A. Oliva, A. Torralba, Temporal relational reasoning in videos, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018.

[37] C. Feichtenhofer, X3d: Expanding architectures for efficient video recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 203–213.

[38] J. Kim, G. Li, I. Yun, C. Jung, J. Kim, Weakly-supervised temporal attention 3d network for human action recognition, Pattern Recognition 119 (2021) 108068.