Gate-Shift Networks for Video Action Recognition

Swathikiran Sudhakaran[1], Sergio Escalera[2,3], Oswald Lanz[1]
[1]Fondazione Bruno Kessler, Trento, Italy
[2]Computer Vision Center, Barcelona, Spain
[3]Universitat de Barcelona, Barcelona, Spain
{sudhakaran,lanz}@fbk.eu, sergio@maia.ub.es

Abstract

Deep 3D CNNs for video action recognition are designed to learn powerful representations in the joint spatio-temporal feature space. In practice however, because of the large number of parameters and computations involved, they may under-perform in the lack of sufficiently large datasets for training them at scale. In this paper we introduce spatial gating in spatial-temporal decomposition of 3D kernels. We implement this concept with Gate-Shift Module (GSM). GSM is lightweight and turns a 2D-CNN into a highly efficient spatio-temporal feature extractor. With GSM plugged in, a 2D-CNN learns to adaptively route features through time and combine them, at almost no additional parameters and computational overhead. We perform an extensive evaluation of the proposed module to study its effectiveness in video action recognition, achieving state-of-the-art results on Something Something-V1 and Diving48 datasets, and obtaining competitive results on EPIC-Kitchens with far less model complexity.

1. Introduction

Video action recognition is receiving increasing attention due to its potential applications in video surveillance, media analysis, and robotics, just to mention a few. Although great advances have been achieved during last years, action recognition models have not yet achieved the success of image recognition models, and the 'AlexNet momentum for video' has still to come.

A key challenge lies in the space-time nature of the video medium that requires temporal reasoning for fine-grained recognition. Methods based on temporal pooling of frame-level features (TSN [41], ActionVLAD [12]) process the video as a (order-less) set of still images and work well enough when the action can be discerned from objects and scene context (UCF-101, Sports-1M, THUMOS). More akin to the time dimension in video, late temporal aggregation of frame-level features can be formulated as sequence learning (LRCN [5], VideoLSTM [23]) and with attention (Attentional Pooling [11], LSTA [33]). At the other hand, early temporal processing is used to fuse short term motion features from stack of flow fields (Two-Stream [32]) or predicted directly from the encoded video (DMC-Net [31]).

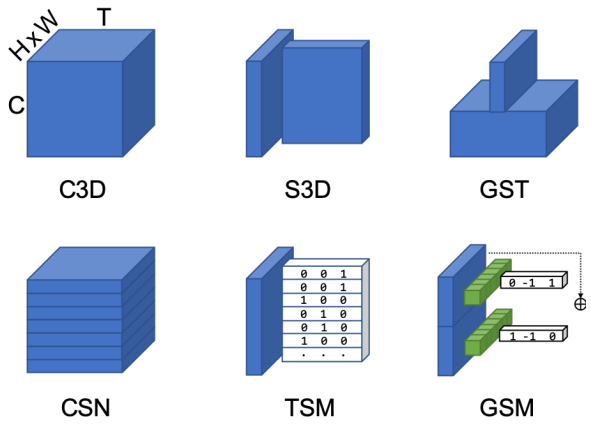


Figure 1: 3D kernel factorization for spatio-temporal learning in video. Existing approaches decompose into channel-wise (CSN), spatial followed by temporal (S3D, TSM), or grouped spatial and spatio-temporal (GST). In all these, spatial, temporal, and channel-wise interaction is hard-wired. Our **Gate-Shift Module (GSM)** leverages group spatial gating (blocks in green) to control interactions in spatial-temporal decomposition. GSM is lightweight and a building block of high performing video feature extractors.

Fine-grained recognition can benefit from deeper temporal modeling. Full-3D CNNs (C3D [37, 15]) process the video in space-time by expanding the kernels of a 2D ConvNet along the temporal dimension. Deep C3Ds are designed to learn powerful representations in the joint spatio-temporal feature space with more parameters (3D kernels) and computations (kernels slide over 3 densely sampled dimensions). In practice however, they may under-perform due to the lack of sufficiently large datasets for training

them at scale. To cope with these issues arising from the curse of dimension one can narrow down network capacity by design. Fig. 1 shows several C3D kernel decomposition approaches proposed for spatio-temporal feature learning in video. A most intuitive approach is to factorize 3D spatio-temporal kernels into 2D spatial plus 1D temporal, resulting in a structural decomposition that disentangles spatial from temporal interactions (P3D [30], R(2+1)D [39], S3D [45]). An alternative design is separating channel interactions and spatio-temporal interactions via group convolution (CSN [38]), or modeling both spatial and spatio-temporal interactions in parallel with 2D and 3D convolution on separated channel groups (GST [26]). Temporal convolution can be constrained to hard-coded time-shifts that move some of the channels forward in time or backward (TSM [24]). All these existing approaches learn structured kernels with a hard-wired connectivity and propagation pattern across the network. There is no data dependent decision taken at any point in the network to route features selectively through different branches, for example, group-and-shuffle patterns are fixed by design and learning how to shuffle is combinatorial complexity.

In this paper we introduce spatial gating in spatial-temporal decomposition of 3D kernels. We implement this concept with Gate-Shift Module (GSM) as shown in Fig. 1. GSM is lightweight and turns a 2D-CNN into a highly efficient spatio-temporal feature extractor. The GSM first applies 2D convolution, then decomposes the output tensor using a learnable spatial gating into two tensors: a gated version of it, and its residual. The gated tensor goes through a 1D temporal convolution while its residual is skip-connected to its output. We implement spatial gating as group spatio-temporal convolution with single output plane per group. We use hard-coded time-shift of channel groups instead of learnable temporal convolution. With GSM plugged in, a 2D-CNN learns to adaptively route features through time and combine them, at almost no additional parameters and computational overhead. For example, when GSM is plugged into TSN [41], an absolute gain of $+32$ percentage points in accuracy is obtained on Something Something-V1 dataset with just 0.48% additional parameters and 0.55% additional Floating Point Operations (FLOPs).

The contributions of this paper can be summarized as follows: (i) We propose a novel spatio-temporal feature extraction module that can be plugged into existing 2D Convolutional Neural Network (CNN) architectures with negligible overhead in terms of computations and memory; (ii) We perform an extensive ablation analysis of the proposed module to study its effectiveness in video action recognition; (iii) We achieve state-of-the-art or competititve results on public benchmarks with less parameters and FLOPs compared to existing approaches.

2. Related Work

Inspired by the performance improvements obtained with deep convolutional architectures in image recognition [16, 36], much effort has gone into extending these for video action recognition.

Fusing appearance and flow. A popular extension of 2D CNNs to handle video is the Two-Stream architecture by Simonyan and Zisserman [32]. Their method consists of two separated CNNs (streams) that are trained to extract features from a sampled RGB video frame paired with the surrounding stack of optical flow images, followed by a late fusion of the prediction scores of both streams. The image stream encodes the appearance information while the optical flow stream encodes the motion information, that are often found to complement each other for action recognition. Several works followed this approach to find a suitable fusion of the streams at various depths [8] and to explore the use of residual connections between them [7]. These approaches rely on optical flow images for motion information, and a single RGB frame for appearance information, which is limiting when reasoning about the temporal context is required for video understanding.

Video as a set or sequence of frames. Later, other approaches were developed using multiple RGB frames for video classification. These approaches sparsely sample multiple frames from the video, which are applied to a 2D CNN followed by a late integration of frame-level features using average pooling [41], multilayer perceptrons [48], recurrent aggregation [5, 23], or attention [11, 33]. To boost performance, most of these approaches also combine video frame sequence with externally computed optical flow. This shows to be helpful, but computationally intensive.

Modeling short-term temporal dependencies. Other research has investigated the middle ground between late aggregation (of frame features) and early temporal processing (to get optical flow), by modeling short-term dependencies. This includes differencing of intermediate features [28] and combining Sobel filtering with feature differencing [35]. Other works [6, 29] develop a differentiable network that performs TV-L1 [46], a popular optical flow extraction technique. The work of [20] instead uses a set of fixed filters for extracting motion features, thereby greatly reducing the number of parameters. DMC-Nets [31] leverage motion vectors in the compressed video to synthesize discriminative motion cues for two-stream action recognition at low computational cost compared to raw flow extraction.

Video as a space-time volume. Unconstrained modeling and learning of action features is possible when considering video in space-time. Since video can be seen as a temporally dense sampled sequence of images, expanding 2D convolution operation in 2D-CNNs to 3D convolution is

a most intuitive approach to spatio-temporal feature learning [37, 15, 2]. The major drawback of 3D CNNs is the huge number of parameters involved. This results in increased computations and the requirement of large scale datasets for pre-training. Carreira and Zisserman [2] addressed this limitation by inflating video 3D kernels with the 2D weights of a CNN trained for image recognition. Several other approaches focused on reducing the number of parameters by disentangling the spatial and temporal feature extraction operations. P3D [30] proposes three different choices for separating the spatial and temporal convolutions and develops a 3D-ResNet architecture whose residual units are a sequence of such three modules. R(2+1)D [39] and S3D-G [45] also show that a 2D convolution followed by 1D convolution is enough to learn discriminative features for action recognition. CoST [21] performs 2D convolutions, with shared parameters, along the three orthogonal dimensions of a video sequence. MultiFiber [3] uses multiple lightweight networks, the fibers, and multiplexer modules that facilitate information flow using point-wise convolutions across the fibers.

Spatial-temporal modeling. Recently, the focus of research is moving to the development of efficient (from a computational point of view) and effective (from a performance point of view) architectures. CNNs provide different levels of feature abstractions at different layers of the hierarchy. It has been found that the bottom layer features are less useful for extracting discriminative motion cues [34, 51, 45]. In [34] it is proposed to apply 1D convolution layers on top of a 2D CNN for video action recognition. The works of [51] and [45] show that it is more effective to apply full 3D and separable 3D convolutions at the top layers of a 2D CNN for extracting spatio-temporal features. These approaches resulted in performance improvement over full 3D architectures with less parameters and computations. Static features from individual frames represent scenes and objects and can also provide important cues in identifying the action. This is validated by the improved performance obtained with two-path structures that apply a parallel 2D convolution in addition to the 3D convolution [49, 26]. MiCT [49] is designed by adding 3D convolution branches in parallel to the 2D convolution branches of a BN-Inception-like CNN. GST [26] makes use of the idea of grouped convolutions for developing an efficient architecture for action recognition. They separate the features at a hierarchy across the channel dimension and separately perform 2D and 3D convolutions followed by a concatenation operation. In this way, the performance is increased while reducing the number of parameters. STM [17] proposes two parallel blocks for extracting motion features and spatio-temporal features. Their network rely only on 2D and 1D convolutions and feature differencing for encoding motion and spatio-temporal features. TSM [24] proposes to shift the features across the channel dimension as a way to perform temporal interaction between the features from adjacent frames of a video. This parameter-less approach has resulted in similar performance to 3D CNNs. However, in all previous approaches, spatial, temporal, and channel-wise interaction is hard-wired. Here, we propose the Gate-Shift Module (GSM), which control interactions in spatial-temporal decomposition and learns to adaptively route features thought time and combine them, at almost no additional parameters and computational overhead.

3. Gate-Shift Networks

In this section we present Gate-Shift Networks for fine-grained action recognition. We first describe their building block, Gate-Shift Module (GSM), that turns a 2D CNN into a high performing spatio-temporal feature extractor, with minimal overhead. We then discuss and motivate the design choices leading to our final GSM architecture used in the experiments.

3.1. Gate-Shift Module

Fig. 2 illustrates the network schematics of 3D kernel factorization approaches (cf. Fig. 1) that have been successfully applied to video action recognition. S3D, or R(2+1)D, P3D, decompose 3D convolutions into 2D spatial plus 1D temporal convolutions. TSM replaces 1D temporal convolution with parameter-free channel-wise temporal shift operations. GST uses group convolution where one group applies 2D spatial and the other 3D spatio-temporal convolution. GST furthermore applies point-wise convolution before and after the block to allow for interactions between spatial and spatio-temporal groups, and for channel reduction and up-sampling. In these modules, the feature flow is hard-wired by design, meaning that features are forwarded from one block to the next without data-dependent pooling, gating or routing decision.

GSM design, in Fig. 2, is inspired by GST and TSM but replaces the hard-wired channel split with a learnable spatial gating block. The function of gate block, paired with fuse block, is selectively routing gated features through time-shifts and merging them with the spatially convolved residual to inject spatio-temporal interactions adaptively. GSM is lightweight as it uses 2D kernels, parameter-free time-shifts and few additional parameters to compute the spatial gating planes.

Based on the conceptual design in Fig. 2, we instantiate GSM as in Fig. 3. GSM first applies spatial convolution on the layer input; this is the operation inherited from the 2D CNN base model where GSM is build in. Then, grouped spatial gating is applied, that is, gating planes are obtained for each of two channel groups, and applied on them. This separates the 2D convolution output into group-gated features and residual. The gated features are group-shifted for-

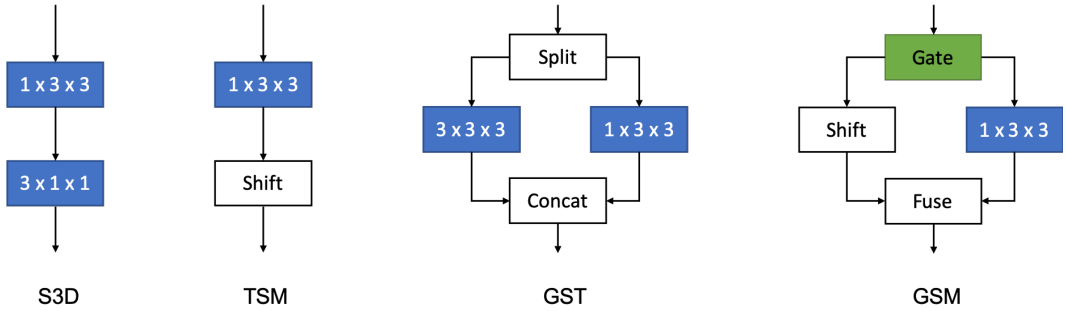


Figure 2: C3D decomposition approaches in comparison to GSM schematics. GSM is inspired by GST and TSM but replaces the hard-wired channel split with a learnable spatial gating block.

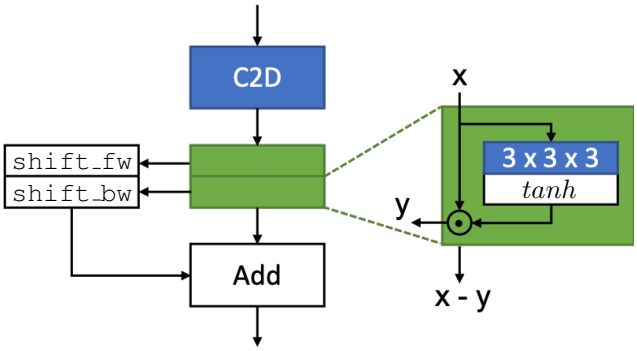


Figure 3: GSM implementation with group gating and forward-backward temporal shift. A gate is a single 3D convolution kernel with tanh calibration, thus very few parameters are added when GSM is used to turn a C2D base model into a spatio-temporal feature extractor.

ward and backward in time, and zero-padded. These are finally fused (added) with the residual and propagated to the next layer. This way, GSM selectively mixes spatial and temporal information through a learnable spatial gating.

Gating is implemented with a single spatio-temporal 3D kernel and tanh activation. With a 3D kernel we utilize short-range spatio-temporal information in the gating. tanh provides spatial gating planes with values in the range $(-1, +1)$ and is motivated as follows. When the gating value at a feature location is +1 and that of the time-shifted feature was +1, then a temporal feature averaging is performed at that location. If the gating value of the time-shifted feature was -1 instead, then a temporal feature differencing is performed. Using tanh, the gating can thus learn to apply either of the two modes, location-wise. It is also found in our ablation study that tanh provides better results than *e.g.* sigmoid that would be the standard choice with gating, see Sec. 4.3 last paragraph.

GSM layer implementation. Let tensor X be the GSM input after 2D convolution (output of blue block in Fig. 3), of shape $C \times T \times W \times H$ where C is the number of channels and WH, T are the spatial and temporal dimensions. Let $X = [X_1, X_2]$ be the group=2 split of X along the channel dimension, and $W = [W_1, W_2]$ be the two $C/2 \times 3 \times 3 \times 3$ shaped gating kernels. Then, the GSM output $Z = [Z_1, Z_2]$ is computed as

$$
\begin{aligned}
Y_1 &= tanh(W_1 * X_1) \odot X_1 && (1)\\
Y_2 &= tanh(W_2 * X_2) \odot X_2 && (2)\\
R_1 &= X_1 - Y_1 && (3)\\
R_2 &= X_2 - Y_2 && (4)\\
Z_1 &= \texttt{shift\_fw}(Y_1) + R_1 && (5)\\
Z_2 &= \texttt{shift\_bw}(Y_2) + R_2 && (6)
\end{aligned}
$$

where '$*$' represents convolution, '$\odot$' is Hadamard product, and `shift_fw`, `shift_bw` is forward, backward temporal shift. Note that the parameter count in this GSM implementation is $2 \times (27 \cdot C/2) = 27 \cdot C$; this is far less than that of a typical C2D block. E.g., the $1 \times 3 \times 3$ block in Fig. 3 has C kernels of size $(9 \cdot C_{in})$ where typically $C \geq C_{in} \gg 3$.

Relation to Residual Architectures. It should be noted that Eqns. 3 and 5 can be reformulated as $R_1 = \texttt{shift\_fw}(Y_1) - Y_1$ and $Z_1 = X_1 + R_1$. This is in analogy to the residual learning of ResNet. In GSM, the residual is the learned spatio-temporal features that are added to the input X_1 to generate discriminative spatio-temporal features for identifying an action class.

Relation to existing approaches. GSM is a generalization of several existing approaches. With gating=0, GSM collapses to TSN [41]; with gating=1, converges to TSM [24] style; with gating=1 and replacing temporal shift with expensive 3D convolution, converges to GST [26] style.

3.2. Gate-Shift Architecture

We adopt TSN as reference architecture for action recognition. TSN performs temporal pooling of frame-level features using C2D backbone. We choose BN-Inception and InceptionV3 as backbone options with TSN, and describe here how we GSM them.

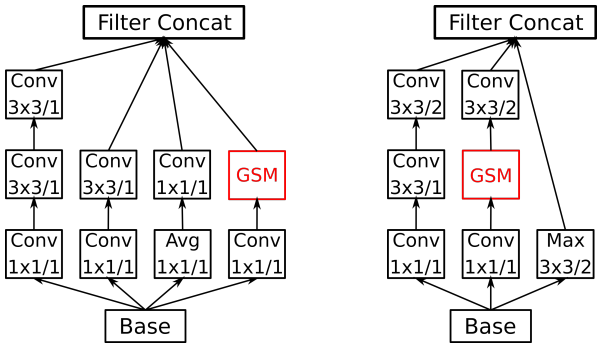


Figure 4: BN-Inception blocks with GSM. The kernel size and stride of convolutional and pooling layers are annotated inside each block.

As shown in Fig. 4, we insert GSM inside one of the branches of Inception blocks. We analyze the branch to which GSM is to be applied, empirically. From the experiments we conclude that adding GSM to the branch with the least number of convolution layers performs the best. A hypothesis is that the other branches consist of spatial convolutions with larger kernel sizes and applying GSM on those branches will affect the spatial learning capacity of the network. This hypothesis is strengthened further by observing a reduced performance when GSM was added inside all the branches. Because of the presence of additional branches in Inception blocks, that encode spatial information, there is no need for a separate spatial convolution operation in GSM. That is, the GSM blocks in Fig. 4 are as in Fig. 3 without the $1 \times 3 \times 3$ spatial convolution block.

For clip level action classification we follow the approach of TSN, that is, we predict the action by average pooling the frame level (now spatio-temporal) scores.

4. Experiments and Results

This section presents an extensive set of experiments to evaluate GSM.

4.1. Datasets

We evaluate Gate-Shift Module (GSM) on three standard action recognition benchmarks, Something Something-V1 [14] (Something-V1), Diving48 [22] and EPIC-Kitchens [4]. Something-V1 consists of 100K videos with 174 fine-grained object manipulation actions. Performance is reported on the validation set. Diving48 dataset contains around 18K videos with 48 fine-grained dive classes. EPIC-Kitchens dataset comprises 34K egocentric videos with fine-grained activity labels. We report the performance obtained on the two standard test splits. Since the test labels are withheld, the recognition scores are obtained from the submission server after we submitted the prediction scores.

Branch	Accuracy (%)
Branch 1	45.11
Branch 2	44.98
Branch 3	45.05
Branch 4	**47.24**
All branches	43.5

Table 1: Ablation analysis done to determine the Inception branch that is most suitable for plugging in GSM.

All the three considered datasets are diverse in nature and require strong spatio-temporal reasoning for predicting the action classes. For instance, Something-V1 dataset does not distinguish among the objects being handled. On the other hand, EPIC-Kitchens dataset require strong spatio-temporal reasoning as well as information about the objects being handled. The videos in Diving48 dataset generally contain a uniform background with fine-grained diving actions and require strong understanding of temporal dynamics of the human body in the video.

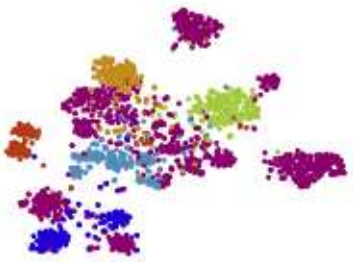
4.2. Implementation Details

As explained in Sec. 3.2, we choose BN-Inception and InceptionV3 as the CNN backbones. GSM is added inside each Inception block of the respective CNNs. Thus a total of 10 GSMs are added. We initialize the 3D convolution in the gating layer with zeros. Thus the model starts as a standard TSN architecture and the gating is learned during training. All models are initialized with ImageNet pre-trained weights. The entire network is trained end-to-end using Stochastic Gradient Descent (SGD) with an initial learning rate of 0.01 and momentum 0.9. We use a cosine learning rate schedule [25]. The network is trained for 60 epochs on Something-V1 and EPIC-Kitchens while Diving48 is trained for 20 epochs. The first 10 epochs are used for gradual warm-up [13]. The batch size is 32 for Something-V1 and EPIC-Kitchens, and 8 for Diving48. Dropout is applied at the classification layer at a rate of 0.5 for Something-V1 and EPIC-Kitchens and 0.7 for Diving48 dataset. Random scaling, cropping and flipping are applied as data augmentation during training. The dimension of the input is 224×224 and 229×229 for BN-Inception and InceptionV3, respectively. The reduced input dimension to InceptionV3 reduces the computational complexity without degradation in performance. If not specified, we use just the center crop during inference.

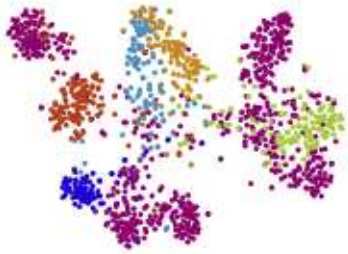
4.3. Ablation Analysis

In this section, we report the ablation analysis performed on the validation set of Something-V1 dataset. In all the experiments, we apply 8 frames as input to the network. We first conducted an analysis to determine the Inception branch that is most suitable for adding GSM. The results of this experiment are reported in Tab. 1. We number each

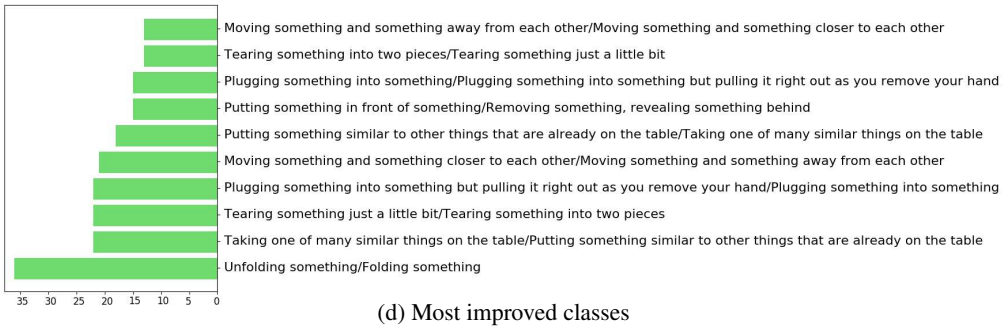
(a) t-SNE plot of TSN features



(b) t-SNE plot of GSM features



(c) t-SNE of most improved classes



(d) Most improved classes

Figure 5: t-SNE plots of the output layer features preceding the final fully connected layers for (a) TSN with BN-Inception, and (b) same TSN but with GSM built-in as in Fig. 4. In these two plots the 10 action categories described in [14] are visualized. In (d) we list the action classes with the highest improvement over Temporal Segment Network (TSN) baseline. X-axis shows the number of corrected samples for each class. Y-axis labels are in the format true label (GSM)/predicted label (TSN). In (c) we visualize the corresponding t-SNE plot.

Model	Accuracy (%)	Params.	FLOPs
BN-Inception (baseline)	17.25	10.45M	16.37G
BN-Inception + 1 GSM	22.7	10.46M	16.37G
BN-Inception + 5 GSM	43.13	10.48M	16.39G
BN-Inception + 10 GSM	**47.24**	**10.5M**	**16.46G**

Table 2: Recognition Accuracy by varying the number of Gate-Shift Modules (GSMs) added to the backbone.

branch from left to right. It can be seen that the best performing model is obtained when GSM is added in branch 4. When GSM is added inside all the branches, we observed the lowest performance as this adversely affects the spatial modeling capacity of the network. From the above experiments, we conclude that GSM is most suited to be added inside the branch which contains the least number of convolutions. We follow the same design choice for InceptionV3 as well. More details regarding the architecture of InceptionV3 is provided in the supplementary material.

We then compared the performance improvement by adding GSM on BN-Inception. Tab. 2 shows the ablation results. Baseline is the standard TSN architecture, with an accuracy of 17.25%. We then applied GSM at the last Inception block of the CNN. This improved the recognition performance by 5%. Increasing the number of GSMs added to the backbone consistently improved the recognition performance of the network. The final model, in which GSM is applied in all Inception blocks results in a recognition accuracy of 47.24%, *i.e.*, $+30\%$ absolute improvement over TSN baseline, with only 0.48% and 0.55% overhead in parameters and complexity, respectively.

Since sigmoid is the general choice used in gating mechanims, we also analyzed the performance of GSM when sigmoid non-linearity is used inside the gating function. Compared to tanh non-linearity, sigmoid unperforms by absolute 3% (47.24% vs 44.75%) proving the suitability of tanh for gate calibration.

4.4. State-of-the-Art Comparison

Something-V1. The recognition performance obtained by GSM is compared with state-of-the-art approaches that just use RGB frames in Tab. 3. We also report the number of frames used by each approach during inference and the corresponding computational complexity in terms of FLOPs. The first block in the table lists the approaches that use

Method	Backbone	Pre-training	#Frames	GFLOPs	Accuracy (%)
TSN [41] (ECCV'16)	BN-Inception	ImageNet	16	32.73	17.52
MultiScale TRN [48] (ECCV'18)	BN-Inception	ImageNet	8	16.37	34.44
R(2+1)D [39] (CVPR'18)	ResNet-34	Sports-1M	32	152	45.7
R(2+1)D [39] from [10] (CVPR'19)	ResNet-34	External	32	152	51.6
S3D-G [45] (ECCV'18)	InceptionV1	ImageNet	64	71.38	48.2
MFNet [20] (ECCV'18)	ResNet-101	-	10	NA	43.9
TrajectoryNet [47] (NeurIPS'18)	ResNet-18	Kinetics	7×10	NA	47.8
TSM [24] (ICCV'19)	ResNet-50	Kinetics	16	65	47.2
STM [17] (ICCV'19)	ResNet-50	ImageNet	16×30	66.5×30	50.7
GST [26] (ICCV'19)	ResNet-50	ImageNet	16	59	48.6
ABM [50] (ICCV'19)	ResNet-50	ImageNet	16×3	35.33×3	46.08
CorrNet [40]	ResNet-101	-	32×30	224×30	51.1
I3D [2] (CVPR'17)	ResNet-50	Kinetics	32×2	108×2	41.6
Non-local [42] (CVPR'18)	ResNet-50	Kinetics	32×2	168×2	44.4
GCN+Non-local [43] (ECCV'18)	ResNet-50	Kinetics	32×2	303×2	46.1
ECO [51] (ECCV'18)	BNInc + ResNet-18	Kinetics	16	64	41.4
Martinez *et al.* [27] (ICCV'19)	ResNet-50	ImageNet	NA	52.17* ×NA	50.1
Martinez *et al.* [27] (ICCV'19)	ResNet-152	ImageNet	NA	113.4* ×NA	53.4
GSM	BN-Inception	ImageNet	8	16.46	47.24
	InceptionV3	ImageNet	8	26.85	49.01
	BN-Inception	ImageNet	16	32.92	49.56
	InceptionV3	ImageNet	16	53.7	50.63
	InceptionV3	ImageNet	16×2	53.7×2	51.68
	InceptionV3	ImageNet	8×2+12×2+16+24	268.47	55.16

Table 3: Comparison to state-of-the-art on Something-V1. *: Computed assuming a single clip of 16 frames as input.

2D CNN and efficient 3D CNN implementations. The second block shows the approaches that use Full-3D CNNs. From the table, it can be seen that GSM results in an absolute gain of +32% (17.52% vs 49.56%) over the TSN baseline. GSM performs better than 3D CNNs or heavier backbones and also those approaches that use external data for pre-training, with considerably less number of FLOPs. GSM performs comparatively to the top performing method [27] with less number of FLOPs. It should be noted that the FLOPs of the architecture described in [27] is computed assuming a single clip of 16 frames. It can also be seen that by using InceptionV3, which is a larger backbone than BN-Inception, the performance of the proposed approach improves. We also evaluated the performance of ensemble of models trained with different number of input frames and achieved a state-of-the-art recognition accuracy of 55.16% [1].

Diving48. Tab. 4 compares performance of GSM on Diving48 dataset with state-of-the-art approaches. We train the network using 16 frames and sample two clips during inference. We use InceptionV3 as the CNN backbone. In this dataset, the actions cannot be recognized from the scene context alone and require strong spatio-temporal reasoning. GSM achieves a recognition accuracy of 40.27%, an improvement of +1.3% over previous state-of-the-art [26].

EPIC-Kitchens. In EPIC-Kitchens, the labels are provided as `verb-noun` pairs and the performance is evaluated on `verb`, `noun` and `action` recognition accuracies. For this dataset, we train GSM as a multi-task problem for `verb`, `noun` and `action` prediction. In the classification layers, we apply `action` scores as bias to the `verb` and `noun` classifiers, as done in LSTA [33]. We use BN-Inception as the backbone CNN. The network is trained with 16 frames. Two clips consisting of 16 frames are sampled from each video during inference. We report the recognition accuracy obtained on the two standard test splits, S1 (seen) and S2 (unseen), in Tab. 5. The first block in the table shows the methods that use both RGB frames and optical flow as inputs while the second block lists the approaches that only use RGB images. From the table, it can be seen that GSM performs better than other approaches that use optical flow images for explicit motion encoding. The only two methods that beat GSM, R(2+1)D [10] and LFB [44], train two separate networks, one trained for `verb` and the other for `noun` classification, and leverage additional data for pre-training. GSM uses a single network for predicting all three labels from a video, thereby making it faster and more memory efficient. In fact, GSM performs better than R(2+1)D model pre-trained on Sports-1M dataset, suggesting that GSM can also improve its performance by pre-training on external data.

4.5. Discussion

In Fig. 5d, we show the top 10 action classes that improved the most by adding GSM to the CNN backbone of TSN. From the figure, it can be seen that the network has enhanced its ability to distinguish between action classes that are similar in appearance, such as `Unfolding something` and `Folding something`, `Putting something infront of something` and `Removing something, revealing something behind`, etc. Sample frames from some

[1] See supplementary document for more details on model ensembles.

Figure 6: Sample frames from Something-V1 videos that belong to the most improved classes when GSM is added.

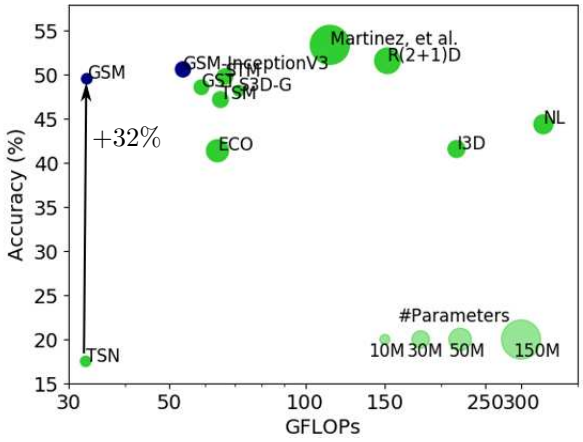


Figure 7: Accuracy-vs-complexity of state-of-the-art on Something-V1, from Tab. 3. Size indicates number of parameters (M, in millions). GSM outperforms or competes in recognition performance with far less model complexity.

of these most improved classes are shown in Fig. 6. From these frames, we can see that reversing the order of the frames changes the action and thus the orderless pooling present in TSN fails to identify the action. On the other hand, GSM is able to improve the recognition score on these classes, providing a strong spatio-temporal reasoning. In order to validate the temporal encoding capability of GSM, we evaluated its performance by applying the video frames in the reverse order. This resulted in a drastic degradation in the recognition performance from 47.24 to 15.38%. On the other hand, there was no change in the recognition performance of TSN when frames reversed in time were applied.

The t-SNE plots of the features from the final layer of the CNN corresponding to the 10 action groups described in [14] are shown in Fig. 5a and 5b. Fig. 5c shows the t-SNE visualization of the most improved classes compared to TSN. We sample 1800 videos from the validation split for the t-SNE visualization. It can be seen that the features from GSM show a lower intra-class and higher inter-class variability compared to those from TSN.

Method	Pre-training	Accuracy (%)
TSN [41] (from [22])	ImageNet	16.77
TRN [48] (from [18])	ImageNet	22.8
R(2+1)D [39] (from [1])	Kinetics	28.9
DiMoFs [1]	Kinetics	31.4
P3D [30] (from [26])	ImageNet	32.4
C3D [37] (from [26])	ImageNet	34.5
Kanojia *et al.* [18]	ImageNet	35.64
CorrNet [40]	-	37.7
GST	ImageNet	38.8
GSM	ImageNet	40.27

Table 4: Comparison to state-of-the-art on Diving48.

Method	Pre-train	S1			S2		
		Verb	Noun	Action	Verb	Noun	Action
TSN [41]	ImageNet	45.68	36.8	19.86	34.89	21.82	10.11
TBN [19]	ImageNet	60.87	42.93	30.31	49.61	25.68	16.80
LSTA [33]	ImageNet	59.55	38.35	30.33	47.32	22.16	16.63
RU-LSTM [9]	ImageNet	56.93	43.05	33.06	43.67	26.77	19.49
LFB [44]	Kinetics	60.0	45	32.7	50.9	31.5	21.2
R(2+1)D [10]	Sports-1M	59.6	43.7	31.0	47.2	28.7	18.3
R(2+1)D [10]	External	65.2	45.1	34.5	58.4	36.9	26.1
GSM	ImageNet	59.41	41.83	33.45	48.28	26.15	20.18

Table 5: Comparison to state-of-the-art on EPIC-Kitchens.

We also analyzed the memory requirement and computational complexity of GSM and various state-of-the-art approaches. Fig. 7 shows the accuracy, parameter and complexity trade-off computed on the validation set of Somthing-V1 dataset. The graph plots accuracy vs GFLOPs and the area of the bubbles indicate the number of parameters present in each method. From the plot, it can be seen that GSM performs competitively to the state-of-the-art [27] with less than one tenth the number of parameters and half the number of FLOPs.

5. Conclusion

We proposed Gate-Shift Module (GSM), a novel temporal interaction block that turns a 2D-CNN into a highly efficient spatio-temporal feature extractor. GSM introduces spatial gating to decide on exchanging information with neighboring frames. We performed an extensive evaluation to study its effectiveness in video action recognition, achieving state-of-the-art results on Something Something-V1 and Diving48 datasets, and obtaining competitive results on EPIC-Kitchens with far less model complexity. For example, when GSM is plugged into TSN, an absolute gain of +32% in recognition accuracy is obtained on Something Something-V1 dataset with just 0.48% additional parameters and 0.55% additional FLOPs.

Acknowledgement

This work is partially supported by ICREA under the ICREA Academia programme.

References

[1] G. Bertasius, C. Feichtenhofer, D. Tran, J. Shi, and L. Torresani. Learning Discriminative Motion Features Through Detection. *arXiv preprint arXiv:1812.04172*, 2018. 8

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proc. CVPR*, 2017. 3, 7

[3] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng. Multi-fiber networks for video recognition. In *Proc. ECCV*, 2018. 3

[4] D. Damen, H. Doughty, G. Maria Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *Proc. ECCV*, 2018. 5

[5] 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 *Proc. CVPR*, 2015. 1, 2

[6] L. Fan, W. Huang, C. Gan, S. Ermon, B. Gong, and J. Huang. End-to-end learning of motion representation for video understanding. In *Proc. CVPR*, 2018. 2

[7] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *Proc. NIPS*, 2016. 2

[8] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proc. CVPR*, 2016. 2

[9] A. Furnari and G. M. Farinella. What Would You Expect? Anticipating Egocentric Actions With Rolling-Unrolling LSTMs and Modality Attention. In *Proc. ICCV*, 2019. 8

[10] D. Ghadiyaram, D. Tran, and D. Mahajan. Large-scale weakly-supervised pre-training for video action recognition. In *Proc. CVPR*, 2019. 7, 8

[11] R. Girdhar and D. Ramanan. Attentional pooling for action recognition. In *Proc. NIPS*, 2017. 1, 2

[12] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. ActionVLAD: Learning spatio-temporal aggregation for action classification. In *Proc. CVPR*, 2017. 1

[13] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *arXiv preprint arXiv:1706.02677*, 2017. 5

[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 *Proc. ICCV*, 2017. 5, 6, 8

[15] K. Hara, H. Kataoka, and Y. Satoh. Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? In *Proc. CVPR*, 2018. 1, 3

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 2

[17] B. Jiang, M. Wang, W. Gan, W. Wu, and J. Yan. STM: SpatioTemporal and Motion Encoding for Action Recognition. *Proc. ICCV*, 2019. 3, 7

[18] G. Kanojia, S. Kumawat, and S. Raman. Attentive Spatio-Temporal Representation Learning for Diving Classification. In *Proc. CVPRW*, 2019. 8

[19] E. Kazakos, A. Nagrani, A. Zisserman, and D. Damen. EPIC-Fusion: Audio-Visual Temporal Binding for Egocentric Action Recognition. In *Proc. ICCV*, 2019. 8

[20] M. Lee, S. Lee, S. Son, G. Park, and N. Kwak. Motion feature network: Fixed motion filter for action recognition. In *Proc. ECCV*, 2018. 2, 7

[21] C. Li, Q. Zhong, D. Xie, and S. Pu. Collaborative Spatiotemporal Feature Learning for Video Action Recognition. In *Proc. CVPR*, 2019. 3

[22] Y. Li, Y. Li, and N. Vasconcelos. RESOUND: Towards action recognition without representation bias. In *Proc. ECCV*, 2018. 5, 8

[23] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. GM Snoek. VideoLSTM convolves, attends and flows for action recognition. *Computer Vision and Image Understanding*, 166:41–50, 2018. 1, 2

[24] J. Lin, C. Gan, and S. Han. Temporal Shift Module for Efficient Video Understanding. In *Proc. ICCV*, 2019. 2, 3, 4, 7

[25] I. Loshchilov and F. Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proc. ICLR*, 2017. 5

[26] C. Luo and A. Yuille. Grouped Spatial-Temporal Aggregation for Efficient Action Recognition. In *Proc. ICCV*, 2019. 2, 3, 4, 7, 8

[27] B. Martinez, D. Modolo, Y. Xiong, and J. Tighe. Action recognition with spatial-temporal discriminative filter banks. In *Proc. ICCV*, 2019. 7, 8

[28] J.Y. Ng and L.S. Davis. Temporal difference networks for video action recognition. In *Proc. IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018. 2

[29] A. J. Piergiovanni and M. S. Ryoo. Representation flow for action recognition. In *Proc. CVPR*, 2019. 2

[30] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *Proc. ICCV*, 2017. 2, 3, 8

[31] Z. Shou, X. Lin, Y. Kalantidis, L. Sevilla-Lara, M. Rohrbach, S. Chang, and Z. Yan. DMC-Net: Generating discriminative motion cues for fast compressed video action recognition. In *Proc. CVPR*, 2019. 1, 2

[32] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proc. NIPS*, 2014. 1, 2

[33] S. Sudhakaran, S. Escalera, and O. Lanz. LSTA: Long Short-Term Attention for Egocentric Action Recognition. In *Proc. CVPR*, 2019. 1, 2, 7, 8

[34] L. Sun, K. Jia, D. Yeung, and B. E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proc. ICCV*, 2015. 3

[35] S. Sun, Z. Kuang, L. Sheng, W. Ouyang, and W. Zhang. Optical flow guided feature: a fast and robust motion representation for video action recognition. In *Proc. CVPR*, 2018. 2

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proc. CVPR*, 2016. 2

[37] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proc. ICCV*, 2015. 1, 3, 8

[38] D. Tran, H. Wang, L. Torresani, and M. Feiszli. Video Classification With Channel-Separated Convolutional Networks. In *Proc. ICCV*, 2019. 2

[39] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proc. CVPR*, 2018. 2, 3, 7, 8

[40] H. Wang, D. Tran, L. Torresani, and M. Feiszli. Video Modeling with Correlation Networks. *arXiv preprint arXiv:1906.03349*, 2019. 7, 8

[41] 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 *Proc. ECCV*, 2016. 1, 2, 4, 7, 8

[42] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proc. CVPR*, 2018. 7

[43] X. Wang and A. Gupta. Videos as space-time region graphs. In *Proc. ECCV*, 2018. 7

[44] C. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick. Long-term feature banks for detailed video understanding. In *Proc. CVPR*, 2019. 7, 8

[45] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proc. ECCV*, 2018. 2, 3, 7

[46] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223, 2007. 2

[47] Y. Zhao, Y. Xiong, and D. Lin. Trajectory Convolution for Action Recognition. In *Proc. NIPS*, 2018. 7

[48] B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. In *Proc. ECCV*, 2018. 2, 7, 8

[49] Y. Zhou, X. Sun, Z. Zha, and W. Zeng. MiCT: Mixed 3d/2d convolutional tube for human action recognition. In *Proc. CVPR*, 2018. 3

[50] X. Zhu, C. Xu, L. Hui, C. Lu, and D. Tao. Approximated Bilinear Modules for Temporal Modeling. In *Proc. ICCV*, 2019. 7

[51] M. Zolfaghari, K. Singh, and T. Brox. ECO: Efficient Convolutional Network for Online Video Understanding. In *Proc. ECCV*, 2018. 3, 7