electronics

MDPI

Article

Action Recognition Using Deep 3D CNNs with Sequential Feature Aggregation and Attention

Fazliddin Anvarov, Dae Ha Kim and Byung Cheol Song *

Department of Electronic Engineering, Inha University, Incheon 22212, Korea; fazliddinanvarov19950611@gmail.com (F.A.); kdhht5022@gmail.com (D.H.K.)

* Correspondence: bcsong@inha.ac.kr; Tel.: +82-32-860-7413

Received: 27 November 2019; Accepted: 9 January 2020; Published: 12 January 2020

Abstract: Action recognition is an active research field that aims to recognize human actions and intentions from a series of observations of human behavior and the environment. Unlike image-based action recognition mainly using a two-dimensional (2D) convolutional neural network (CNN), one of the difficulties in video-based action recognition is that video action behavior should be able to characterize both short-term small movements and long-term temporal appearance information. Previous methods aim at analyzing video action behavior only using a basic framework of 3D CNN. However, these approaches have a limitation on analyzing fast action movements or abruptly appearing objects because of the limited coverage of convolutional filter. In this paper, we propose the aggregation of squeeze-and-excitation (SE) and self-attention (SA) modules with 3D CNN to analyze both short and long-term temporal action behavior efficiently. We successfully implemented SE and SA modules to present a novel approach to video action recognition that builds upon the current state-of-the-art methods and demonstrates better performance with UCF-101 and HMDB51 datasets. For example, we get accuracies of 92.5% (16f-clip) and 95.6% (64f-clip) with the UCF-101 dataset, and 68.1% (16f-clip) and 74.1% (64f-clip) with HMDB51 for the ResNext-101 architecture in a 3D CNN.

Keywords: action recognition; 3D CNN; deep feature attention

1. Introduction

One of the main objectives of artificial intelligence is to build a model that can accurately learn human actions and intentions [1]. Human action recognition is important because it has been applied to various applications, such as surveillance systems, health care systems, and social robots. Recently, a three-dimensional (3D) convolutional neural network (CNN) for action recognition with spatiotemporal convolutional kernels achieved better performance than 2D CNNs that can only cover the spatial kernel. The representative research in video-based action recognition is based on two-stream architectures [2], recurrent neural networks (RNN) [3], or spatiotemporal convolutions [4,5]. Two-stream approaches use two separate CNNs, one using red–green–blue (RGB) data, and the other using optical flow images to deal with movement.

Recently, most of the research has relied on the modeling of motion and temporal structures. Temporal segment network (TSN) [6] uses a sparse segment to model long-range temporal structure. Other 3D CNN methods [4,7,8] tried to solve temporal modeling issues by using an additional dimension of convolution on the temporal axis with the ambition that the models learn hierarchical motion patterns as in the image space. In [9,10], the temporal modeling is further enhanced through tracking feature points and body joints over time. The above-mentioned methods demonstrated the advantages of the 3D CNN over the 2D CNN because of the ability to learn spatiotemporal characteristics. However, all previous methods have the limitation of analyzing action changes in consecutive frames because of limited coverage of convolutional filter. Since action recognition is a

fine-grained recognition problem, identifying and differentiating small changes in consecutive frames and connecting all frames logically is important. Another milestone of action recognition is that applying 3D CNN on action recognition causes overfitting problems because of a huge number of parameters. To alleviate the overfitting problem, Carreira and Zisserman launched the Kinetics dataset [11], which is large enough to train a 3D CNN successfully while coping with the challenge of overfitting. Moreover, Hara et al. [12] conducted experiments to train residual architectures on four different action datasets and achieved an outstanding result. However, the main idea of [12] was to check whether the dataset is efficient to tackle a huge number of parameters of 3D CNN in action recognition or not. So, we believe that using more complex sequential-based architectures pretrained with large datasets could achieve better results.

Drawing inspiration from [12], we propose a sequential version of squeeze-and-excitation (SE) and self-attention (SA) modules, prove the aggregation of both modules, and propose a new method for recognizing video action behavior. We assume that the connection of spatial information with a temporal stream logically provides a better understanding of action behavior. Figure 1a demonstrates ResNext-101 and ResNet-18. Due to the simplicity of applying additional modules, we choose the residual architecture mentioned above. It consists of four blocks and each block consists of convolution filter, batch normalization, Rectified Linear Unit (ReLU) activation, and max pooling. Figure 1b is residual architecture with our proposed module, i.e., SE and SA. We located SE and SA modules after the last layer of each block. The SE module for channels is first; after that, the SE module is used for the sequence (number of frames), and then lastly, the SA module is applied. Implementing exactly in that order provides us the best performance. The reason is that first the SE module, explicitly capturing the correlation between channels of convolutional features, learns to selectively differentiate information features and removes less useful features. After that, the SA module computes the response as a weighted sum of characteristics at positions that contain more useful features (since the SE removes less useful features). In this way, using the SE module improves the channel and sequence interdependencies, and the SA module learns each part of the image, which provides useful information to identify and differentiate small changes in consecutive frames. Note that the detailed architecture of the proposed network is given in Appendix A like [13]. To our best knowledge, it is the first time that sequential-based action recognition has used an adaptive feature aggregation scheme with large pretrained weights. In summary, our contributions are as follows:

- We propose a sequential version of SE and SA modules and apply them to create a new approach for efficiently analyzing action behavior on 3D CNN.
- We validate our proposed modules in both quantitative and qualitative ways and attain s tate-of-the-art results with a marginal computation.

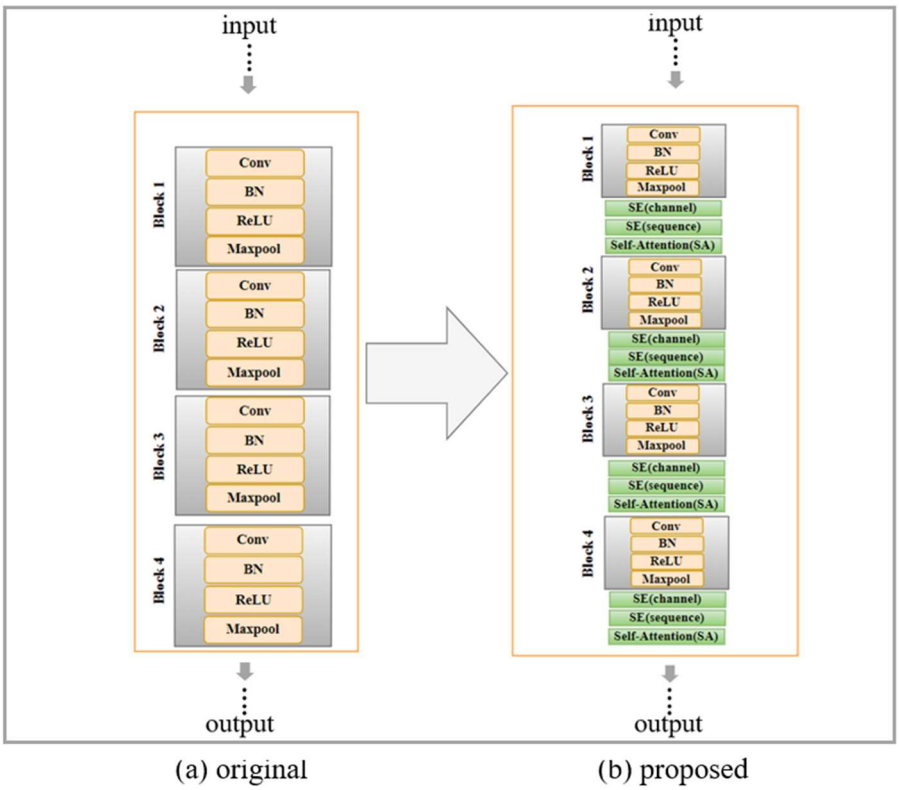


Figure 1. The overall framework of our network. Squeeze-and-excitation and self-attention follow after each layer. The number of blocks is four.

2. Related Works

2.1. Squeeze-and-Excitation

SE is a module that is designed to improve the network's representation ability by allowing dynamic channel-specific recalibration. VGG-Nets from the Visual Geometry Group at the University of Oxford [14] and Inception [15] showed that increasing the depth of a network can lead to a considerable increase in the quality of representation. The same idea can be applied to the SE module to enhance the representation ability of feature analysis. Hu et al. [16] came up with a mechanism for explicitly modeling dynamic, non-linear dependencies between channels using global information that can facilitate the learning process and enhance the representation ability of networks. Based on the previous success history of SE experiences, we show how we can make a sequential version of the SE module and how the SE module can be useful for video action recognition.

2.2. Self-Attention

SA is a commonly used module in computer vision and other fields of artificial intelligence. For example, the SA module has been widely used in many tasks, such as machine translation, skeleton hand-gesture recognition, and task-independent sentence representation [17–19]. Vaswani et al. [17] implemented a self-attention module in machine translation to analyze semantic and temporal relationships among words within a sentence. Chen et al. [18] demonstrated the self-attention mechanism represented by graphs to learn the spatiotemporal information contained in hand skeletons. Wang et al. [8] formalized self-attention as a non-local process to define the spatiotemporal dependencies in a video sequence. Despite such huge progress, the self-attention module has not been implemented for video-based action recognition yet. Therefore, in the next section, we propose basic SE and SA modules and how to modify them for sequential information.

3. Proposed Method

In this section, we will discuss the basic knowledge about SE and SA modules and the method of applying these modules to a 3D CNN. Moreover, we will analyze the importance of active movement in one way of applying both short-term and long-term temporal information.

3.1. Proposed SE Module

SE is a module for enhancing channels' interdependencies with a low computation cost. Using an SE module can improve feature representation by explicitly developing channel interconnections so that the network can increase its sensitivity to information. More specifically, we give access to global data, and reset filter responses in two steps, i.e., *squeeze* and *excitation*, before transforming them into the next transformation.

We first consider the signal to each channel of output features to address the problem of the exploitation of channel dependencies. Since each convolutional filter operates only within a local field and is not able to provide information outside of this region, a squeeze global spatial operation is implemented to solve this problem. We use global average pooling to squeeze each channel into a single numeric value. The formula of the squeeze operation is:

$$z_c = F_{sq}(u_c) = \frac{1}{HWS}\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{k=1}^{C} u_c(i,j,k) \quad (1)$$

where the transformation output, u_c, is a collection of local descriptors that are expressive for the entire video. *H*, *W*, and *S* represent height, weight, and sequence, respectively. Bias terms are omitted to simplify the notation. To use the information in the squeeze operation, we implement a second excitation operation. The idea of the excitation method completely captures channel-wise dependencies provided by the squeeze operation:

$$s = F_{ex}(z, W) = \sigma\big(g(z, W)\big) = \sigma\big(W_2\delta(W_1 z)\big) \quad (2)$$

where δ represents the ReLU activation function, F_{ex} is the excitation operation, $W_1 \in R^{\frac{C}{r}\times C}$, and $W_2 \in R^{C\times\frac{C}{r}}$.

Equations (1) and (2) present methods operating on channels. For a sequence:

$$z_c = F_{sq}(u_c) = \frac{1}{HWC}\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{k=1}^{C} u_c(i,j,k) \quad (3)$$

$$s = F_{ex}(z, W') = \sigma\big(g(z, W')\big) = \sigma\big(W'_2\delta(W'_1 z)\big) \quad (4)$$

where $W'_1 \in R^{\frac{S}{d}\times S}$ and $W'_2 \in R^{S\times\frac{S}{d}}$.

In contrast to the original SE for a 2D CNN [16], we consider not only the channel but also the number of frames (Figure 2). We consider the concatenation of channels and sequences after each layer in our network, which makes our module more effective and allows us to achieve better efficiency. Figure 3 (left) demonstrates the SE module for image-based action recognition, and Figure 3 (middle) represents the SE (channel) and SE (sequence) parts of the proposed method for video-based action recognition, which consider both channel and sequence of frames. Figure 3 (right) is an abstract block diagram of Figure 3 (middle) which consists of two parts. Each part consists of global pooling, two fully connected (FC) layers, ReLU, and sigmoid. In the first part, we applied the SE block for channels. When we implement the SE operation, we set all the values (weight, height, and sequence) to 1 except for the channel value. So, SE operates on the channels (in other words, we need only channels). The output of each SE block for channels ($\widetilde{\mathrm{X}}_c$) is obtained using rescaling with the activation *s*:

$$\widetilde{\mathrm{X}}_c = F_{scale}(u_c, s_c) = s_c u_c \quad (5)$$

where $F_{scale}(u_c, s_c)$ is a channel-wise multiplication of scalar s_c and the feature map $u_c \in \mathbb{R}^{H\times W}$.

In the second part, to conduct an experiment on the sequence, we swap channel and sequence places using the transpose function. Then the same process is applied as in the first part, but this time for the sequence. The output of the SE block for the sequence is obtained by:

$$\widetilde{X}_s = F_{scale}(u_s, s_s) = s_s u_s \quad (6)$$

where $F_{scale}(u_s, s_s)$ is sequence-wise multiplication of scalar s_s and the feature map $u_s \in \mathbb{R}^{H \times W}$.

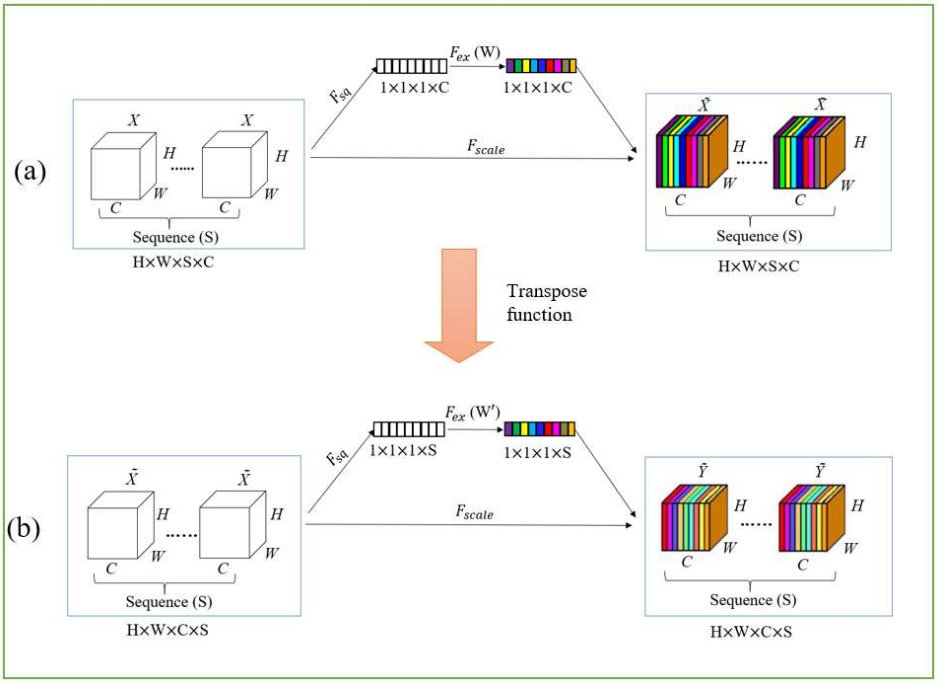


Figure 2. Squeeze-and-excitation block for a 3D convolutional neural network (CNN). Sequential (S) means the number of frames. In our case, 16 frames and 64 frames were used: (**a**) squeeze-and-excitation for a channel, and (**b**) squeeze-and-excitation for a sequence.

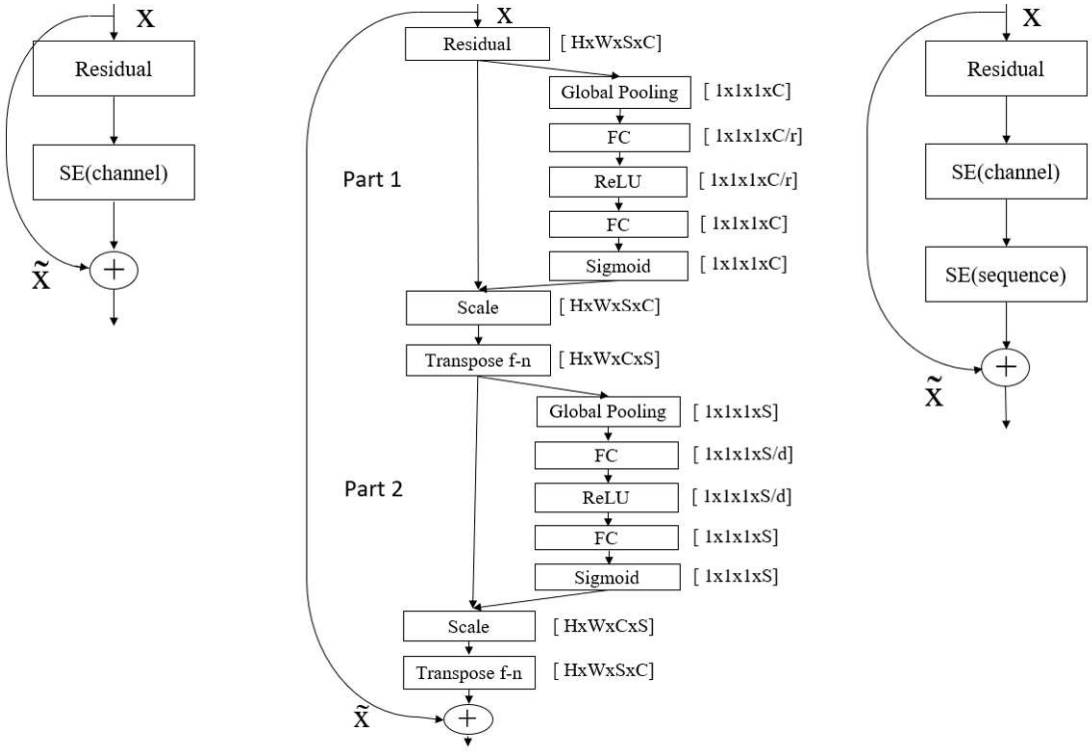


Figure 3. Squeeze-and-excitation (SE) module. We divide the SE module into two parts: channels and sequences. S represents sequence, and C represents channel. Reduction values: $r = 16,\ d = 2$.

In the end, we again apply the transpose function to make the order of weight, height, sequence, and channels the same as at the beginning. The reason for using the transpose function is to continue further operation; the output and input of the SE block should be the same. Then, we add the output of the transpose function with initial X and provide X, which is the final output. Reduction numbers (d and r) are hyper-parameters that allow us to vary the capacity and computational cost of the SE blocks in the network. To provide a good balance between performance and computational cost, we conduct experiments with the SE block for different r values. Setting $r = 16$ (for channel) and $d = 2$ (for sequence) gave us the best trade-off between performance and complexity. Note that by applying squeeze-and-excitation for both channel and sequence, we can consider both long and short-term action correlations adaptively.

3.2. *Self-Attention (SA) Module*

Self-attention is a module that calculates the response as a weighted sum of the features at all positions. The main idea of self-attention is to help convolutions throughout the image domain to capture long-range, full-level interconnections. The network implemented with a self-attention module can help to determine images with small details that are connected with fine details in different areas of the image at each position [20–22].

Our task in this experiment is to extend the SA idea to a 3D CNN; more concretely, we implemented the self-attention idea for multiple frames (a sequence). Unlike [23], where the SA module is implemented on a single image, we examine a multi-frame module for video frames. Since we apply the SA block to every single image in a sequence (16 or 64 frames), the benefit we can get from the SA module is more valuable compared to the single-frame case, and the overall performance is higher.

Most videos in UCF-101 and HMDB51 involve human and item interactions. While the previous methods only focused on a single action [23], in not all cases can a sequence exactly present the interaction. We implemented self-attention for sequential actions to better understand the interactions of humans and items, since the environment around humans is also an important part of defining actions. Since we consider a 3D CNN with multiple frames, the SA module helps to capture all images to make a better prediction based on the action aspects in images, and helps us use this knowledge to connect all frames logically. Figure 4 demonstrates our self-attention module for a 3D CNN. Compared to the basic self-attention module, we concatenate these operations sequentially and embed to our SA module for the 3D CNN, as shown in Figure 1.

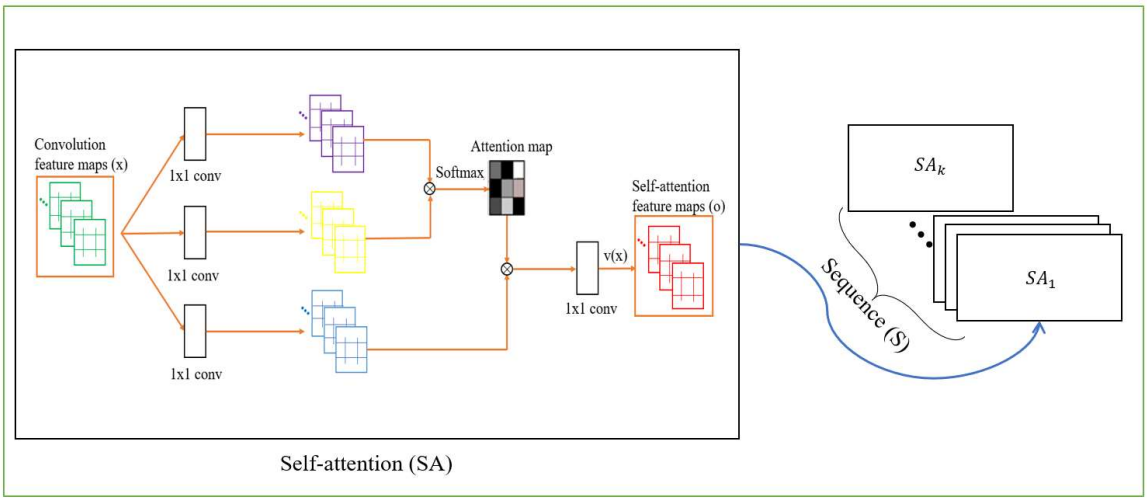


Figure 4. Self-attention module. The ⊗ denotes matrix multiplication. The SoftMax operation is applied to each row. The self-attention module is executed individually for each frame; k means the number of frames, which is 16 or 64.

4. Experiments

4.1. Dataset and Training Configuration

We adopted UCF-101 [24] and HMDB51 [25] datasets for evaluating the performance of our model. The UCF-101 dataset contains 13,320 images of action from 101 classes of human actions. Both datasets include three training/testing splits (70% and 30%, respectively). UCF-101 consists of unconstrained videos downloaded from YouTube with challenges such as poor lighting, cluttered backgrounds, and severe camera movement. To remove non-action frames, the videos were temporarily cut. The average duration of each video is about seven seconds. The HMDB-51 dataset contains 6766 videos from 51 human action classes. Similar to UCF-101, the videos were cut to an average length of three seconds. The training/testing split was the same as UCF-101. The main difference between the two datasets is the number of classes and instances of actions, dynamic backgrounds, and camera movements.

Details of data preprocessing are as follows. We conducted experiments with the same data augmentation for 16 frames (16f-clip) and 64 frames (64f-clip). In training, we randomly sampled a 112×112 crop from a random clip of the given length and applied random horizontal flipping, which includes reversing the horizontal axis in the case of flow input. Since ResNext-101 achieved state-of-the-art performance on both UCF-101 and HMDB51 datasets, we chose it as our main architecture for implementation. We used stochastic gradient descent with a momentum of 0.9 to train a mini-batch dataset. The initial learning rate for the 16f-clip was 0.1 and for the 64f-clip, it was 0.01. Finally, we used the top-1 mean accuracy for evaluating the action recognition dataset.

To assess the suggested techniques, we conducted experiments on the two distinct architectures: ResNext-101 and ResNet-18. Both architectures have four layers. Compared to ResNet-18, ResNext-101 (using the setting of $32 \times 4d$) adds an additional (cardinality) block to the base network, which makes it more powerful at extracting convolution feature maps [12]. We expanded the SE module of image-based models into video-based models to consider the number of frames. After the first, second, third, and fourth blocks, the number of frames in a sequence was 8, 4, 2, and 1, respectively (see Figure 1). Similar to the SE approach, we expanded the SA module of the 2D CNN to a 3D CNN to consider the number of frames in Figure 1.

4.2. Experimental Results

In this section, we provide experimental results and a comparison of our results with earlier state-of-the-art methods of the 3D CNN. Note that before training, we fine-tuned the networks using the Kinetic400 dataset. In Tables 1 and 2, we see higher performance than in previous work in which SE and SA modules were not applied. The only difference between Tables 1 and 2 is the number of video frames used for training. As SE and SA modules were added, the performance improvements were 0.5% and 0.4%, respectively, with UCF-101 in Table 1. In addition, we can see the same performance increment with HMDB51 in Table 1. However, most importantly, the aggregation of SE and SA achieved an additional 0.8% performance improvement. We can also say that the performance tendency is consistent in 64-frame clips.

Table 1. Top-1 accuracy on UCF-101 and HMDB51 datasets for 16f-clips (%).

Method	UCF-101	HMDB51
ResNext-101	91.7	66.7
ResNext-101 + SA	92.1	67.3
ResNext-101 + SE	92.2	67.8
ResNext-101 + SE + SA	92.5	68.1

Table 2. Top-1 accuracy on UCF-101 and HMDB51 dataset for 64f-clips (%).

Method	UCF-101	HMDB51
ResNext-101	95.2	73.5
ResNext-101 + SA	95.5	74.0
ResNext-101 + SE	95.4	73.8
ResNeXt-101 + SE + SA	95.6	74.1

We also performed experiments on ResNet-18 to prove model generalization. Results with the ResNet-18 architecture are in Table 3. Note that the synergy effect of using both SE and SA is much higher than that in previous experiments (2.8% on UCF-101 and 2.6% on HMDB51). We can conclude that for the shallower model, our approach shows better improvement compared to the more complex ones. All the obtained results indicate that both squeeze-and-excitation and self-attention were successfully implemented and worked well enough in the 3D CNN. These observations are compatible with our assumption that the layers of self-attention are useful in capturing structural data and long-distance dependence.

Table 3. Top-1 accuracies using ResNet-18 architecture on UCF101 and HMDB51.

Method	UCF-101 (%)	HMDB51 (%)
ResNet-18	84.4	56.4
ResNet-18 + SA	85.1	57.1
ResNet-18 + SE	86.3	58.2
ResNet-18 + SE + SA	87.2	59.0

Figures 5 and 6 depict the confusion matrixes of ResNext-101 in UCF101 and HMDB51 respectively, together with precision and recall values to make readers understand evaluation metrics better and easier. Due to the large numbers of classes in UCF101 (101 classes) and HMDB51 (51 classes), it is difficult to show all the classes in one figure together with precision and recall. So, we present the confusion matrix for 20 classes, which were randomly chosen in the dataset. The black values in recall represent how many examples were given to the network during the testing period. The black values in precision represent how many times each label was predicted during the testing period. In other words, precision demonstrates qualitative results, while recall demonstrates quantitative result. The overall accuracies are 95.06% and 74.06% for the UCF101 and HMDB51 datasets, respectively. The corresponding balanced accuracies are 95.03% and 74.05%. According to the results of Figures 5 and 6, the model has almost the discriminative ability to all classes (low variance), which means that our methods are effective at identifying action changes for all actions. In addition, we provide the normalized confusion matrices for all classes on the UCF101 and HMDB51 datasets. From Figure 7 on UCF101, we can see that our model performs very well on most categories. Categories that contain only objects without interaction with items such as 'Body Weight Squats', 'Handstand Pushups', and 'Jumping Jacks' give the best accuracies. However, our model misclassifies some samples from 'Walking', 'Diving', 'Golf Swing', and 'Soccer Penalty' because of similar action characteristics between actions. As a result, we found that the background context information is very important when analyzing action behaviors.

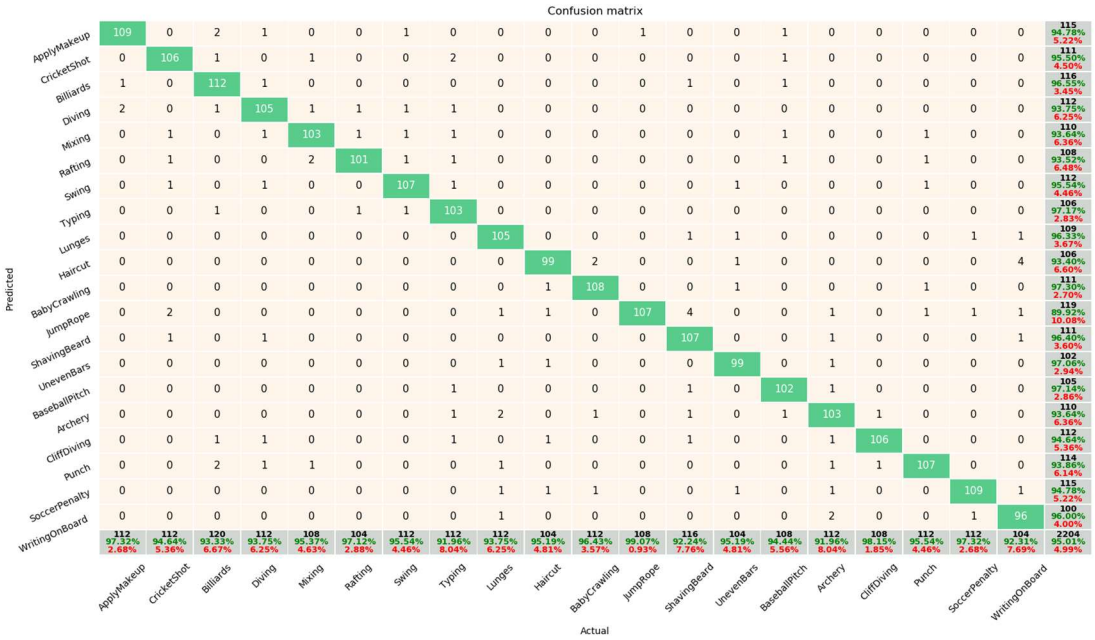


Figure 5. Confusion matrix for ResNext-101 (64f-clips) on UCF101. The last row and last column present recall and precision percentages, respectively.

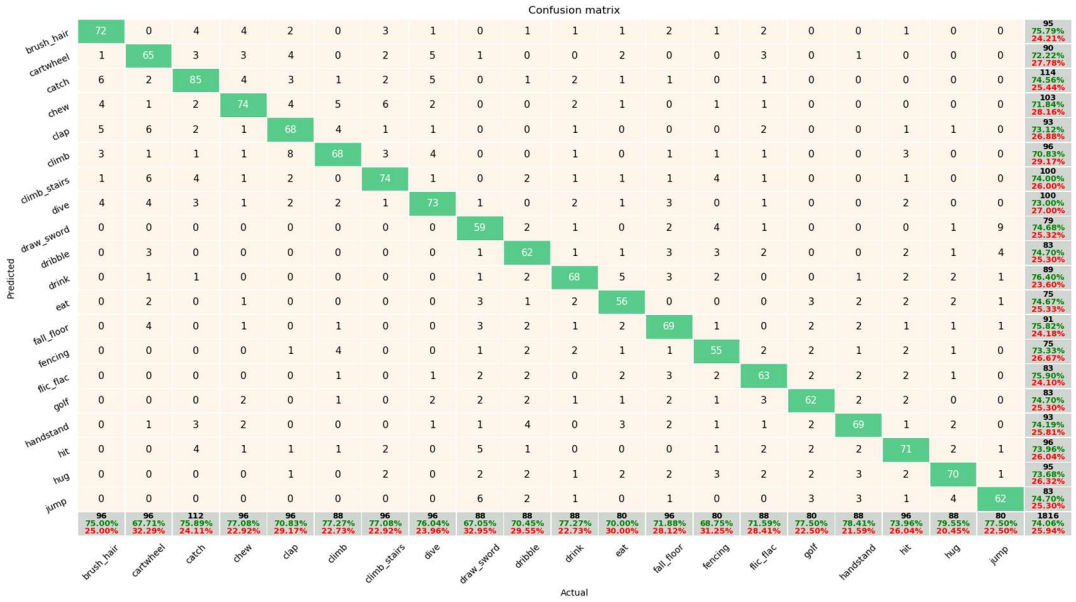


Figure 6. Confusion matrix for ResNext-101 (64f-clips) on HMDB51. The last row and last column present recall and precision percentages, respectively.

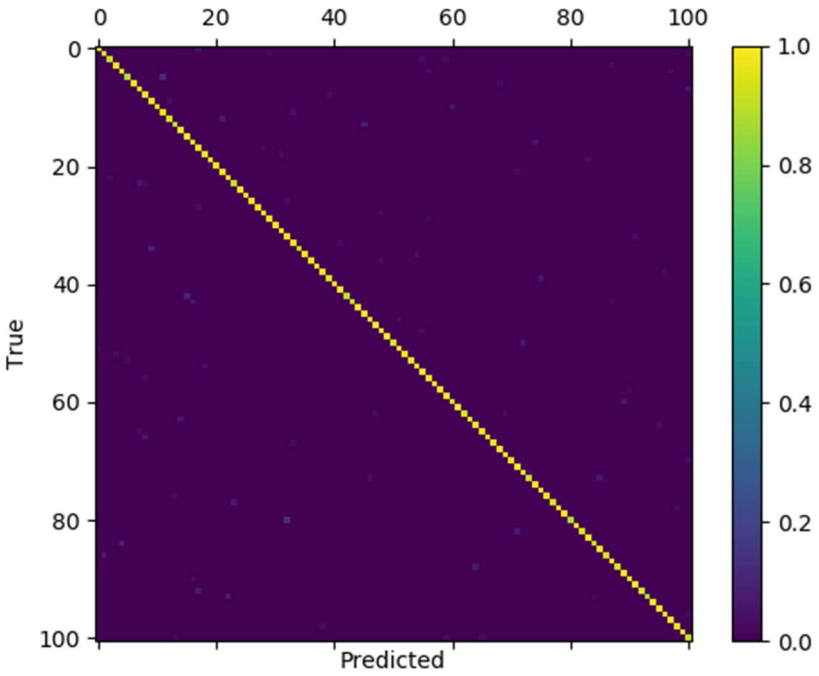


Figure 7. Normalized confusion matrix for ResNext-101 (64f-clips) on UCF101 for all classes.

Figure 8 also depicts the normalized confusion matrix for the HMDB51 dataset. The trend is similar to Figure 7. Still, we can see the misclassifications for several pairs of labels. This is due to the difficulty of identifying complex action behaviors such as for instance basketball, catching, jumping, and throwing. It is usually difficult to understand such actions because they are composed of many different sub-actions.

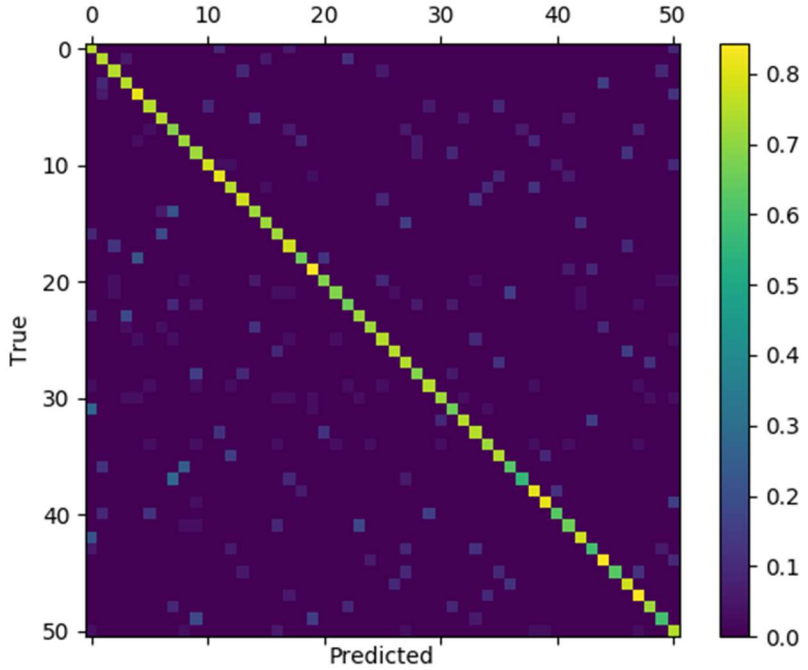


Figure 8. Normalized confusion matrix for ResNext-101 (64f-clips) on HMDB51 for all classes.

The comparison of our method with other state-of-the-art methods is given in Table 4. The accuracies for ResNext-101 (fourth and fifth rows) are our own training results obtained by using this paper's configuration with their default parameter settings. Except for [4,6], all methods were fine-tuned 3D CNN models on the Kinetics dataset. The results of this experiment are essential in determining whether the squeeze-and-excitation and self-attention methods are useful for 3D CNN action recognition or not. Our method achieved state-of-the-art performance confined to using only RGB sequence frames. For example, in a 64-frame clip, the performance of prior ResNext on UCF-101 was 95.2%, compared to the performance of ResNext with SE and SA at 95.6%. For ResNet-101, we can see a performance improvement of 2.6%.

Table 4. Top-1 accuracies on UCF101 and HMDB51 compared with the state-of-the-art methods.

Method	UCF-101 (%)	HMDB51 (%)
P3D ResNet [6]	88.6	-
C3D [4]	82.3	-
ResNet-18 [12]	84.4	56.4
ResNext-101 (16f) [26]	91.7	66.7
ResNext-101 (64f) [26]	95.2	73.5
ResNet-18 + SE + SA	87.2	59.0
ResNext-101 + SE + SA (16f)	92.5	68.0
ResNext-101 + SE + SA (64f)	95.6	74.1

4.3. Discussion of SE and SA Modules

From the results we acquired, we can conclude that both SE and SA work for 3D CNN action recognition well enough; however, in all cases, the SE module performs better than the SA module except for the 64f-clip in the ResNext-101 architecture. The reason that the overall SE module works better than the SA module is that we consider both channel and sequence for SE, but only the channel for SA, and because the sequence and channel concatenation did not yield the meaningful results we expected.

5. Ablation Study

5.1. Integration Strategy of the SE Module

The objective of Table 5 is to conduct an ablation study of analyzing the influence of the SE module one stage at a time. More specifically, we added the SE module after each specific block and checked which layer has more impact in terms of analyzing action behavior, i.e., block1, block2, block3, and after each layer. As expected, the performance was higher with each subsequent layer. The reason is that as the number of layers increases, the amount of information (for both channels and sequences) increases as well. We noted that SE blocks produce performance advantages when they are implemented in each of these architecture phases. The gains produced by SE blocks at various stages are complementary in the sense that they can be effectively combined to further improve the network performance. The results are given in Table 5.

Table 5. Effect of integration of SE blocks with ResNext-101 at different stages on HMDB51.

Method	Top-1 Accuracy (%)
ResNext-101	66.7
ResNext-101 + SE (stage 1)	66.95(+0.25)
ResNext-101 + SE (stage 2)	67.23(+0.53)
ResNext-101 + SE (stage 3)	67.43(+0.73)
ResNext-101 + SE (all stages)	68.03(+1.33)

5.2. Computation Inference Time of SE and SA Modules

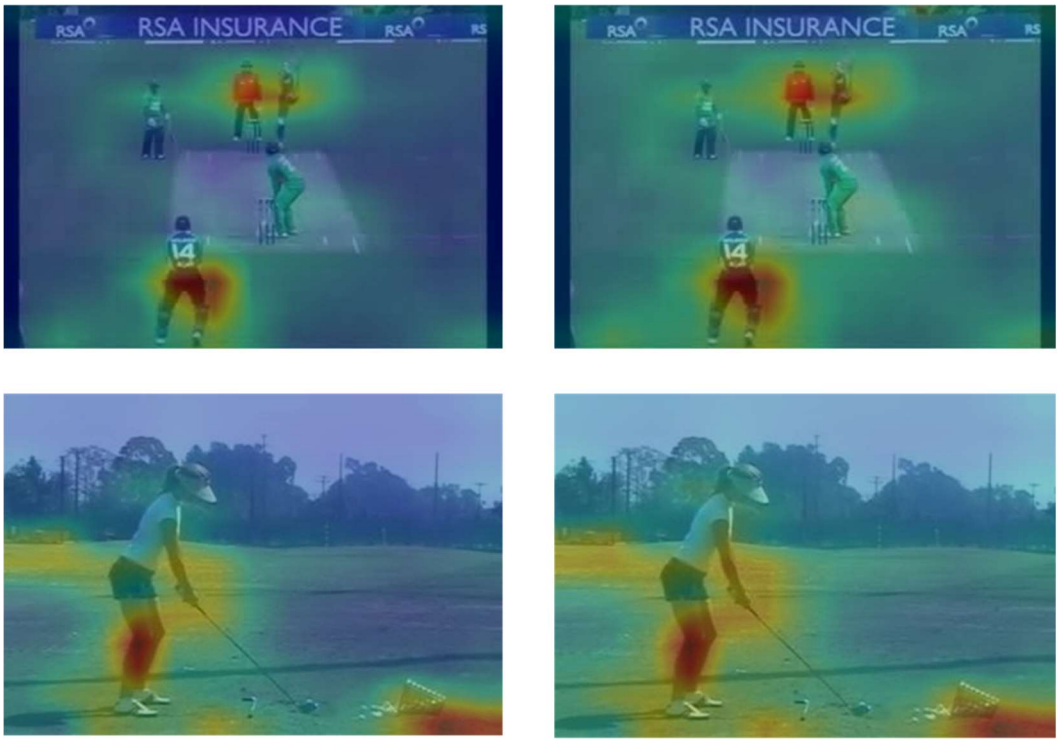
The reason for showing Table 6 is to check which module (SE or SA) was heavier to train and provides a good balance between inference time and network complexity. As you can see from Table 6, SA makes our network inference time longer (heavier) compared to the SE module. That is why we removed the SA module after the first and second blocks and only trained after the third and fourth block, since the SA module provides more efficiency after the third and fourth layers compared to the first and second. The result shows (ResNext-101 + SE + SA **) that our inference time decreased pretty much; however, the performance did not decrease too much. So, it means that the SA after third and fourth blocks can provide almost the same performance with much less inference time. The purpose of this experiment is that other approaches in future work can apply the SA module after only the third and fourth layers with about the same performance to save time.

Table 6. Top-1 accuracy and complexity time for 16f-clips on HMDB51. All the streams are fine-tuned from Kinetics400 pretrained models. * means SA implemented after each layer, ** means SA implemented only after the third and fourth layers.

Method	UCF-101 (%)	Inference Time per Image (ms)
ResNext-101	66.7	2.5
ResNext-101 + SE	67.82	15.9
ResNext-101 + SA *	67.28	22
ResNext-101 + SA **	67.19	16.5
ResNext-101 + SE + SA *	68.12	26.1
ResNext-101 + SE + SA **	67.97	20.6

5.3. Qualitative Results.

The attention map is a method to analyze the implicit attention of a CNN. We implemented an activation map [27] to make our findings more comprehensive. The class activation map indicates the discriminative image regions used by the CNN to identify an action. For example, we demonstrate some results for public datasets (see Figure 9) and the other results for randomly chosen images from web crawling (see Figure 10). From the results, we can see that the main focus is on action, and there is comparably less attention to other aspects of the image. Since the idea with SA is to learn the entire picture, the network starts learning every single detail around the action, and the final decision on action recognition is made based on the aggregation of action and the environment. From qualitative results, we can conclude that the network's ability to identify the action region increases. All of these prove that our trained model can focus on important and meaningful actions.



(**a**) (**b**)

Figure 9. Class activation maps on red–green–blue (RGB) frames. After implementing SE and self-attention (SA), we can see that our activation map's active region increased a bit and also covered non-active areas: (**a**) results before and (**b**) after the implementation of our approach.

Figure 10. Qualitative analysis for real images. These samples do not belong to our dataset, and they were randomly chosen by web crawling. It can be seen that areas such as the human arm have a significant influence on behavior judgment.

6. Conclusions

In this paper, we proposed a novel approach to video action recognition, which uses the aggregation of squeeze-and-excitation and self-attention modules. Using these two modules together, we showed that dynamic changes across frames can be captured more accurately and efficiently with almost no additional computation cost. We also presented qualitative results from experiments to make a balanced decision based on both types of data. Extensive experiments demonstrated the effectiveness of our approach, which achieved state-of-the-art performance across different datasets and architectures.

Author Contributions: The work described in this article is the collaborative development of all authors. All authors contributed to the idea of data processing and designed the algorithm. F.A. and D.H.K. made contributions to data measurement and analysis and B.C.S. led us to research direction. All authors participated in the writing of the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by an Inha University research grant.

Conflicts of Interest: The authors declare they have no conflicts of interest.

Appendix A

Table A1 provides the layer-by-layer description of our network, which corresponds to Figure 1b. This table includes every detail of the proposed network.

Table A1. Layer-by-layer description of our network.

Layer	Name	Description
1	Input	Input layer
2	Conv_1	64 3 × 3 × 3 Convolution
3	BN_1	Batch normalization
4	ReLU_1	Rectified-linear unit
5	Maxpool_1	3 × 3 × 3 Max pooling
6	SE (channel)_1	SE module for channel
7	SE (sequence)_1	SE module for sequence
8	SA_1	Self-attention module
9	Conv_2	512 3 × 3 × 256 Convolution
10	BN_2	Batch normalization
11	ReLU_2	Rectified-linear unit
12	Maxpool_2	3 × 3 × 3 Max pooling
13	SE (channel)_2	SE module for channel
14	SE (sequence)_2	SE module for sequence
15	SA_2	Self-attention module
16	Conv_3	1024 3 × 3 × 512 Convolution
17	BN_3	Batch normalization
18	ReLU_3	Rectified-linear unit
19	Maxpool_3	3 × 3 × 3 Max pooling
20	SE (channel)_3	SE module for channel
21	SE (sequence)_3	SE module for sequence
22	SA_3	Self-attention module
23	Conv_4	2048 3 × 3 × 1024 Convolution
24	BN_4	Batch normalization
25	ReLU_4	Rectified-linear unit
26	Maxpool_4	3 × 3 × 3 Max pooling
27	SE (channel)_4	SE module for channel
28	SE (sequence)_4	SE module for sequence
29	SA_4	Self-attention module

30	Avgpool	Global average pooling
31	FC	2048 dim. Fully Connected layer
32	Output	Output vectors for 51/101 classes

References

1. Seok. W; Park. C. Recognition of human motion with deep reinforcement learning. IEIE Transactions on Smart Processing and Computing. **2018**, 7, 245-250.
2. Simonyan, K.; Zisserman, A. Two-stream convolutional networks for action recognition in videos. In Proceedings of the NIPS, Montreal, Canada, 8–13 December 2014.
3. Donahue, J.; Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the CVPR, Boston, MA, USA, 7–12 June 2015.
4. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning spatiotemporal features with 3D convolutional networks. In Proceedings of the ICCV, Santiago, Chile, 7–13 December 2015; pp. 4489–4497.
5. Tran, D.; Ray, J.; Shou, Z.; Chang, S.; Paluri, M. Convnet architecture search for spatiotemporal feature learning. *arXiv preprint* **2017**, arXiv:1708.05038.
6. Qiu, Z.; Yao, T.; Mei, T. Learning spatiotemporal representation with pseudo-3d residual networks. In Proceedings of the ICCV, Venice, Italy, 22–29 October 2017.
7. Carreira, J.; Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the CVPR Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
8. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-local neural networks. In Proceedings of the CVPR, Salt Lake City, UT, USA, 18–22 June 2018.
9. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal segment networks: Towards good practices for deep action recognition. In Proceedings of the 14th European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016.
10. Zhao, Y.; Xiong, Y.; Lin, D. Trajectory convolution for action recognition. In Proceedings of the Advances on Neural Information Processing Systems, Montréal, Canada, 3–8 December 2018.
11. Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, CH.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; et al. The kinetics human action video dataset. *arXiv preprint* **2017,** arXiv:1705.06950**.**
12. Hara, K.; Kataoka, H.; Satoh, Y. Can spatiotemporal 3d cnns retrace the history of 2d CNNS and imagenet. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 18–22.
13. Kahaki, S.M.M.; Nordin, M.J.; Ahmad, N.S.; Arzoky, M.; Ismail, W. Deep convolutional neural network designed for age assessment based on orthopantomography data. *Neural Comput. Appl.* **2019**, *9*, 1–12.
14. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the ICLR, San Diego, CA, USA, 7–9 May 2015.
15. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the CVPR, Boston, MA, USA, 7–12 June 2015.
16. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Beijing, China, 20–24 August 2018; pp. 7132–7141.
17. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the NIPS, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
18. Chen, Y.; Zhao, L.; Peng, X.; Yuan, J.; Dimitris, N. Construct Dynamic Graphs for Hand Gesture Recognition via Spatial-Temporal Attention. In Proceedings of the BMCV, Cardiff, UK, 12 September 2019.
19. Lin, Z.; Feng, M.; Nogueira dos Santos, C.; Yu, M.; Xiang, B.; Zhou, B.; Bengio, Y. A structured self-attentive sentence embedding. *arXiv preprint* **2017**, arXiv:1703.03130.
20. Zhixing, T.; Mingxuan, W.; Xie, J.; Chen, Y.; Shi, X. Deep semantic role labeling with self-attention. In Proceedings of the AAAI, New Orleans, LO, USA, 2–7 February 2018.
21. Lee, J.; Lee, I.; Kang, J. Self-Attention Graph Pooling. *arXiv preprint* **2019**, arXiv:1904.08082.
22. Ye, L.; Rochan, M.; Liu, Z.; Wang, Y. Cross-modal self-attention network for referring image segmentation. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16-20 June 2019; pp. 10502-10511.
23. Zhang, H.; Goodfellow, I.; Metaxas, D.; Odena, A. Self-attention generative adversarial networks. *arXiv preprint* **2018**, arXiv.1805.08318.

24. Soomro, K.; Roshan Zamir, A.; Shah, M. UCF101: A dataset of 101 human action classes from videos in the wild. *arXiv preprint* **2012**, arXiv.1212.0402.
25. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. HMDB: A large video database for human motion recognition. In Proceedings of the ICCV International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2556–2563.
26. Crasto, N.; Weinzaepfel, P.; Alahari, K.; Schmid, C. MARS: Motion-augmented RGB stream for action recognition. In Proceedings of the Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
27. Zhou, B.; Khosla, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the CVPR, Las Vegas, NV, USA, 27–30 June 2016.