IEMS5910G

Advanced Research and Development Project I

Event-based Action Recognition

BY

ZHANG, Haoyu

1155180042

MScIE

DEPARTMENT OF INFORMATION ENGINEERING

THE CHINESE UNIVERSITY OF HONG KONG

December, 2022

Event-based action recognition

Haoyu Zhang 1155180042

Abstract

Event-based camera provides a new paradigm for us to understand the visual information. Unlike traditional cameras which capture absolute brightness of each pixel simultaneously at a fixed rate (frame per second), the event-based camera only captures the change of each pixel independently and asynchronously, providing spatially sparse but temporally dense results. With microsecond level of resolution, high speed actions can be recognized in a better way.

This paper investigated action recognition of event data captured by DAVIS (Dynamic and Active-pixel Vision Sensor) which also produces grayscale image frames sharing same photosensor array with event data. We compared the size of event data and frame data in the perspective of pixels and converted event data into frames so that we can apply frame-based Temporal Shift Module (TSM) model to it. We found that with data augmentation, the accuracy rate of action recognition using pure event data is on par with grayscale frame images.

However, there are still many challenges for efficient light-weight event-based action recognition and other applications. We discussed these issues and pointed out future study directions for researchers.

Index Terms – action recognition, event-based camera, data augmentation, temporal shift model.

The implementation is available on: https://github.com/Tonystark64/Event-based-action-recognition

1. Introduction

Inspired by the neuron architecture of silicon retina design, a new paradigm to represent visual data using event-based cameras has gained more and more attentions recently. By asynchronously measuring changes in light intensity of each pixel, event-based sensors show a lot of advantages, releasing the potentials of many new applications. Featuring high dynamic range, high pixel bandwidth and low latency, the event data can better reduce motion blur which is beneficial to motion estimation also. The ability to restore high-speed actions is very useful in Extended Reality (XR), mobile robots and self-driving cars.

The principle of event camera is to convert changes in light intensity into “on” and “off” events based on a threshold. [1] A level is to measure whether the change reaches a certain level to trigger an event. When an event is indeed triggered, the level would be reset to prepare for next event. This level is our threshold, normally it is set to be 10%--50% [2] of illumination change. We use 1-bit (0 for decrease, 1 for increase) to denote the changing state (polarity) of brightness and record it in a spatial-temporal coordinate as $e_i = (X_i, Y_i, T_i, P_i)$. We measure log intensity and compare it with threshold to decide an event with formula: $\Delta Log(X_i, Y_i, T_i) = |Log(X_i, Y_i, T_i) - Log(X_i, Y_i, T_i - \Delta t)| \geq threshold$. As shown in Figure 1, some

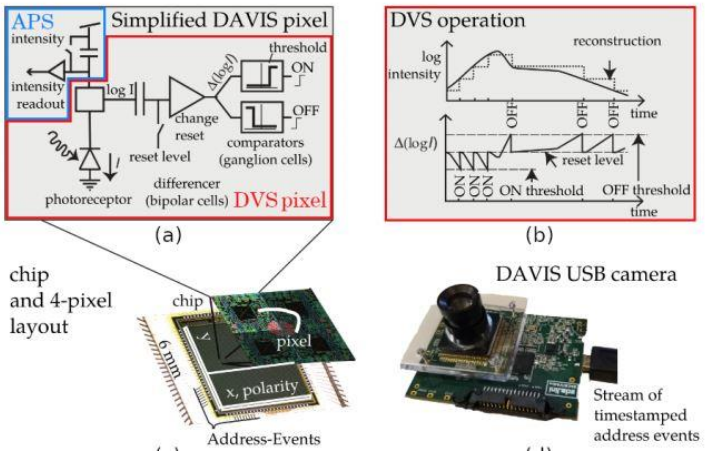

Figure 1 Principles of DAVIS camera [1]

manufacturers would integrate event sensor with frame-based sensor, sharing the same photosensor array.

As for the advantages, event camera is inherently insensitive to changes in the scene illumination but is adept at the change in reflectance [3]. This feature makes segmenting moving objects from background very easy. On the other hand, the low latency is satisfied because event camera does not need a global exposure for the whole frame.

However, the event sensor is far from ideal due to below drawbacks. First the low fill factor (photon collection rate) is a main obstacle for massive deployment. Some researchers suggest a back-site illumination method to tackle this problem, however, it would lead to spurious leakage of photon and the extra cost cannot be ignored. Another issue is that the shot noise is more severe for event camera. Thirdly, from some reconstruction demo videos, we noticed when both sensor and object are static, the event camera would stop generating images.

To fill the gap in practical applications, we need novel methods specified for event data. For example, spiking neuron network can act as a specified processor.

2. Related work

Much work has been done to represent event data, preprocess the event data to better fit in the off-the-shelf DNN models. We would briefly go through some related research work.

With so much event data, it is a natural way to group them and process together because single or little event data contain trivial information. The number of events in each group is an important hyperparameter. Also, as mentioned previously, some researchers combined events and frames for object tracking. They chose to detect edges on frames and strong edges indicated more events. With event data, they track the edge patterns asynchronously. It is worthy to mention even reconstructed frames by events can achieve great performances in object detection and tracking.

There are so many novel methods in representing the event data. Firstly, we may use event frame or 2D histogram to accumulate the polarity pixel-wise [4]. We can convert event data to several frames with identical time intervals. As the formula in [4] implies: $\sum_{t\in I} P(x,y,t) = f[n](x,y)$. Secondly, we may represent with time surface method. In a 2D map,

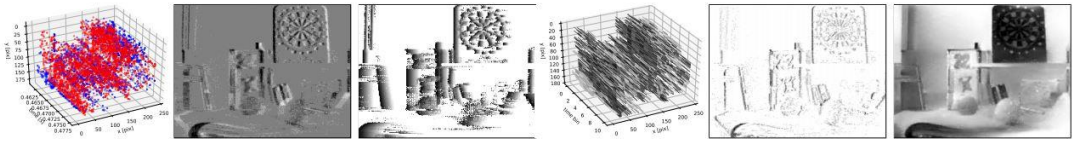

Figure 3. Several event representations (Section 3.1) of the *slider_depth* sequence [98]. (**a**) Events in space time, colored according to polarity (positive in blue, negative in red). (**b**) Event frame (brightness increment image $\Delta L(\mathbf{x})$). (**c**) Time surface with last timestamp per pixel (darker pixels indicate recent time), only for negative events. (**d**) Interpolated voxel-grid ($240 \times 180 \times 10$ voxels), colored according to polarity, from dark (negative) to bright (positive). (**e**) Motion-compensated event image [99] (sharp edges obtained by event accumulation are darker than pixels with no events, in white). (**f**) Reconstructed intensity image by [8]. Grid-like representations are compatible with conventional computer vision methods [100].

Figure 2 Representations of event-based data [1]

each pixel contains the timestamp of latest event in the 2D location. More recent events have a larger weight, and we may capture spatial-temporal neighbors and use these correlations to reduce the noise. The third method is to use voxel grid to best restore the event data in a 3D histogram. As shown in figure 2, different representations are displayed.

The spiking neuron network can process in an event-by-event style and yield a high accuracy in image classification and action recognition but with a high computation overhead.

One important issue in this field is a lack of training data for event-based computer vision tasks. It would take too much time to label event data also. Therefore, Zanardi et al. [5] proposed a ingenious worm-hole learning method to solve this problem. It is a semi-supervision learning method incorporating teacher student training model applied in both RGB and event detector. As a result, the illumination invariance is successfully transferred to RGB detector.

3. Datasets

Our dataset records 31 actions of 31 people with a few direction-sensitive actions. Each action lasts for about 30 seconds, and we want to predict the action in specific video segments.

Each volunteer would perform all the actions in a specified sequence with intervals in between, and the whole process would be recorded with both a frame based RGB camera and a DAVIS camera capturing both event and grayscale frame images in the same photosensor array. To train for a classifier in action recognition, we need to manually label and segment for each action of each volunteer. We would discuss our datasets with more details in the following.

(a) Analysis on data types

There are in total three sensors: RGB, grayscale and event-based camera. Because grayscale and event camera share the same photosensor array the RGB camera used another, there exists spatial and temporal bias between event data and RGB frames.

Figure 3 RGB and event-based sampled data

As shown in Figure 3, the perspectives of event camera and RGB camera are different, and we can see that the actions are not synchronous. We displayed the event points by

accumulating polarities according to RGB frame rate. On the other hand, if we have the same settings for event data and grayscale frame images, they perfectly match in both in spatial and temporal domain, as shown in Figure 4.

Figure 4 Grayscale and event-based sampled data

(b) Size comparison

When performing action recognition tasks, edges are more important than low frequency plain area. Since event data already captured edges and segmented moving objects from the background, we expect a smaller size for event data.

The grayscale images we used are of size 346x260 at 25 frames per second. As for the event data, it is observed that in an interval of 0.4—0.6 second, the 346x260 frame can be filled fully with event polarities. As we mentioned earlier, each polarity is a binary value, but event data also includes 3D spatial-temporal coordinates. Because all the data is collected indoors and the illumination can be considered as an approximate constant, we can use 0.4 second as the time for event to fill up 346x260 space for size calculation.

Apart from the 1-bit polarity, we need 9-bit (0—511) for X, 9-bit (0—511) for Y and 19-bit for microsecond level of time (0—524287 microseconds). Hence in each location, the size of event data would be 38 bits. For the grayscale image, each pixel takes up 8-bit and the frame per second (FPS) is 25. So, in an interval of 0.4 second, there are 10 frames, resulting in a size of 80 bits for grayscale image in each location.

Above we compared the size of event data and grayscale frames in a coarse and simple way which did not consider any compression schemes. In fact, there are many mature compression algorithms for frame-based images like H.264. Actually, the event data occupies much larger space than grayscale frames. For example, a labelled and segmented grayscale video is of size 3.83MB. (encoded in PIM4 as .avi file with duration of 30 seconds) The corresponding event data stored as pickle format occupies 80.3MB which is about 20X of grayscale frame images.

Therefore, novel compression algorithms designed for event-based data is required. Although event data is compact and does not have much low frequency components, there is indeed much room for compression and noise reduction. For example, the stripes on the clothes can generate dense points in an area but these patterns are not useful for action recognition tasks. We can also use temporal neighbors to reduce the noise and use quantization to decrease the temporal resolution.

(c) Action Types

Our action datasets include many different types of actions. We studied the IBM event-based gesture datasets [6] and made some improvements on it.

In IBM event dataset, the distance between sensor and person is relatively fixed, while some actions may involve moving from close-up to long shot, showing different sizes of a same person. In our dataset, we design some actions (e.g., walking) that allow a person to get

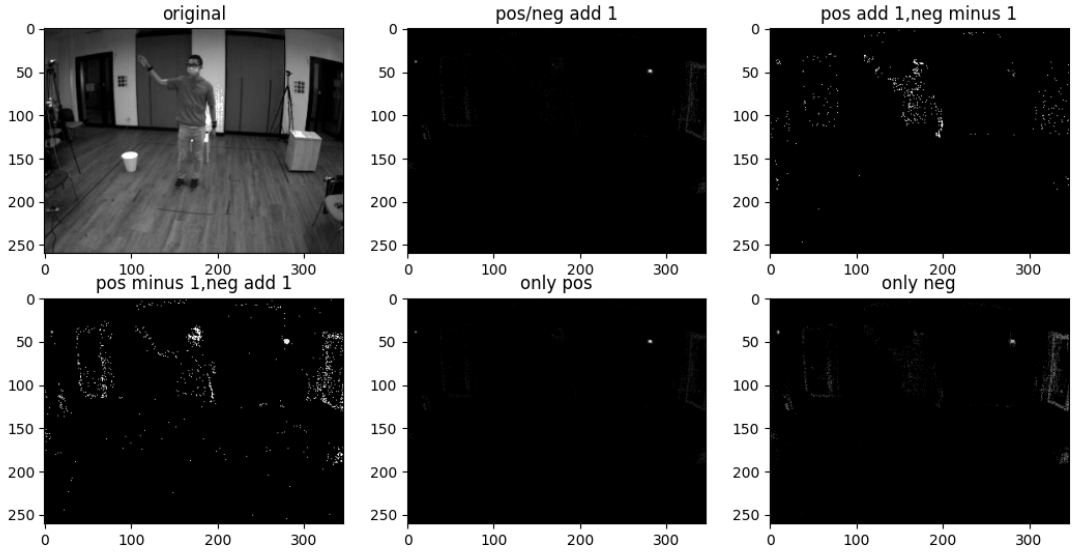

Figure 5 Event accumulation to form a frame (waving)

closer to or further from the camera in just an action.

Additionally, our dataset also includes some actions with very little movements like bemusing which cannot be found in IBM event dataset. We have large movement actions (e.g., jumping, pushing, …) and actions with unobvious movements. We have horizontal and vertical actions, direction-sensitive and direction-insensitive actions.

However, due to the limited time for manual labelling and limited GPU resources. In the experiment, the training set only uses 31 actions of 10 people (subjects) and test set uses 3 people. This can explain the accuracy rate of the experiment, but we can still find insights when comparing the results of fully event-based recognition, fully grayscale recognition and combined recognition.

4. Methods

Nowadays there are many off-the shelf video understanding deep neural network models. To make use of these models, we need to preprocess our event data and generate “event frames” so that we can use DNN models to train action classifiers with pure event data. To have a better comparison with frame-based data, we can generate event frames with the same frame per second (FPS).

(4a) Event preprocessing

The FPS of grayscale frame is 25 so we group event data in every 0.04 second. The problem is how to generate a frame with event data within 0.04 second. In Figure 5 we tried several ways in processing both positive and negative polarities. The action in Figure 5 is “waving” so naturally, the region of interest should be in the arm of the person. From an empty frame, if we add 1 for positive polarity and subtract 1 for negative polarity based on its (X, Y) location, we can get an approximate binary image in Figure 5. In this way, the noises are greatly suppressed but the ROI important for action recognition is preserved. We have tried different increments but still found that a single increment is enough to render binary frames.

(4b) Combine event and grayscale

Since the event data and grayscale share the same photosensor array, we can combine these two before feeding into

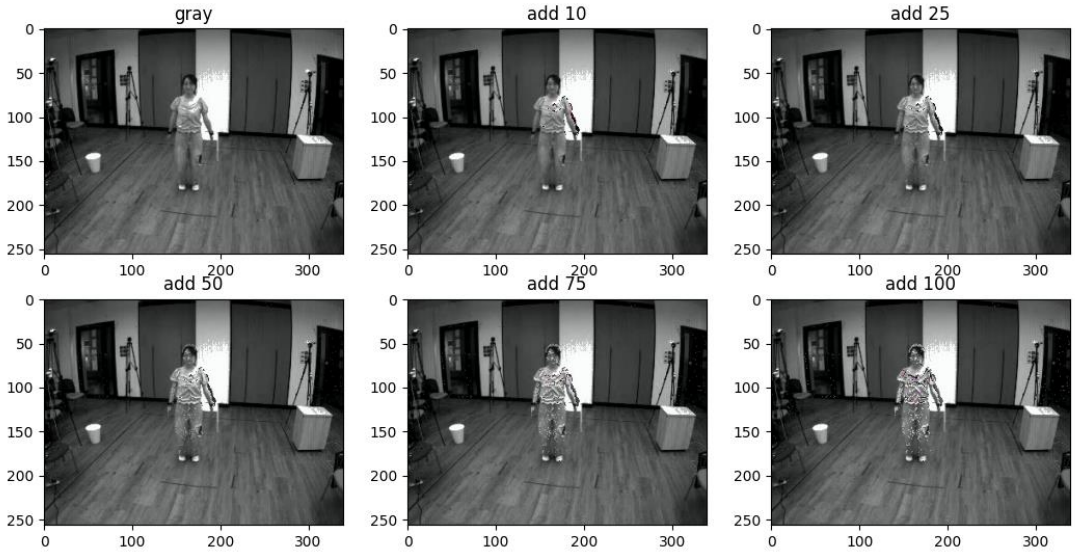

Figure 6 Combine event and grayscale with large movement action (Jump)

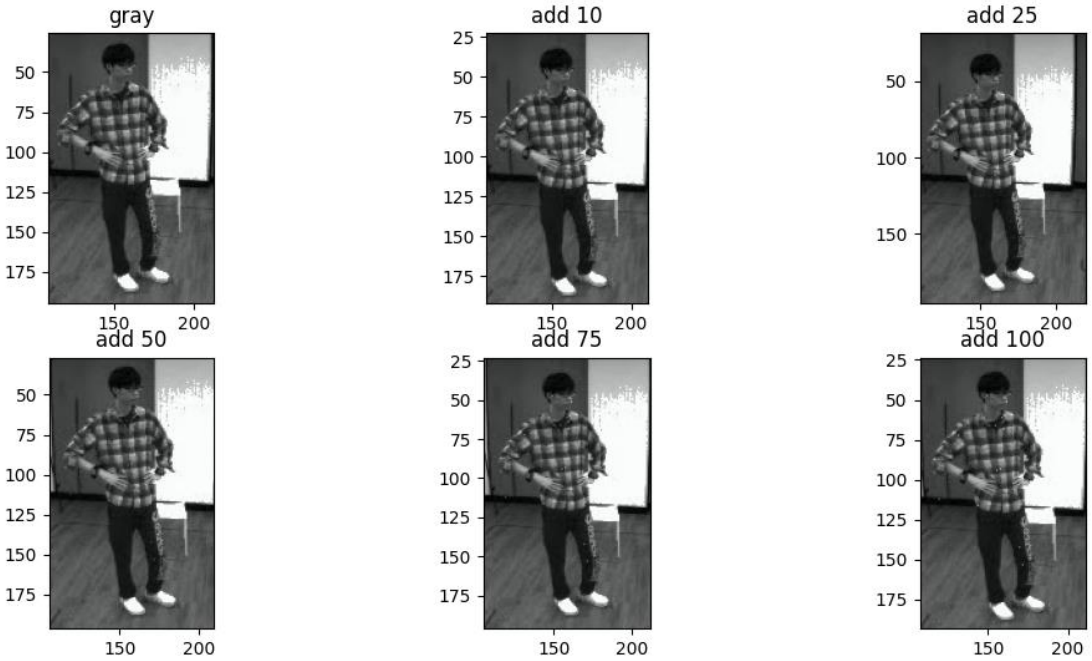

Figure7 Combine event and grayscale with unobvious movement action (Bemuse & Stand)

the DNN models. We add the binary event-frame onto the grayscale frame and tried different increments. The effect is different for actions with large movements and actions with unobvious movements, as shown in Figure 7. For instance, with an increment of 25, the edges have been outlined in grayscale obviously for large movement actions. While for unobvious actions, we can see several points when the increment reaches 50. Therefore, we would try different combinations with different increments into the DNN models.

(4c) Data Augmentation

Due to the limit of labeling time and GPU resources, we only have actions of ten people for training and three people for testing. To improve accuracy without extra labelling work, we adopted data augmentation to increase training data. The general method includes flipping, random cropping, random rotation, etc. There are researchers apply flipping for

Figure 8 Flipping data augmentation

convolution neural network training [7]. In our dataset, all horizontal actions are not Direction-sensitive, and the afterwards experiments do show an improvement in recognizing horizontal movements. The four direction-sensitive actions are all vertical. (sit-down/stand-up/fall/crawl)

(4d) Temporal Shift Module

Under resource-constraint scenario, we need to find some efficient DNN models to perform action recognition with low GPU memory requirements. Temporal Shift Module (TSM) shifts part of the channels along temporal dimensions [8], providing a general and efficient action recognition model. If we use ResNet50 as the backbone, the GPU memory requirement is just around 7GB. So, we would use TSM to train with our preprocessed event frames with different combinations. We also tried another similar model but with even lower GPU memory requirement: Temporal Interlacing Network (TIN) [9] which can train the model even faster but with a little cost in accuracy. TIN reduces computation by embedding temporal information into spatial domain. It can discover spatial-temporal correlation at once but with little cost in accuracy.

5. Experiments

After preprocessing event data, we can turn to MMACTION2 [10] framework to train TSM model with our own datasets. MMACTION2 supports mainstream models and tasks, so that we can compare the performance in our dataset using different models and pretrained dataset. Although our dataset contains few direction-sensitive actions, the proportion is rather low. Therefore, we would prefer the more general pretrained dataset Kinetics 400 [11].

The TSM/TIN models on MMACTION2 [8] [12] support two type of model input: raw frames and videos. For videos, extra decoding time is required but the size is smaller since the encoding is compression. On the other hand, raw frames occupy larger space, but the processing speed is faster for the model. Firstly, we need to configure a model with pretrained checkpoints and configure python file and modify the configuration accordingly. For example, the GPU resource available, learning rate (LR is also related to the number of GPU), training epochs, etc.

After the training is finished, we may check for the error cases and study the reason, so that we can further improve the accuracy rate.

Below the accuracy rate is not optimal mainly due to a lack of labelled training data. But still, we can infer that the flipping augmentation can indeed increase the accuracy rate and training with pure event-based data can have performance close to grayscale frames. When we further check for the accuracy rates of horizontal actions, they indeed increase after flipping augmentation with better capabilities to distinguish among horizontal actions.

However, the improvement margin of data augmentation is not very ideal, because the bottleneck is still at the shortage of labelled training data. Using ten subjects to train on 31 classes is

Accuracy Rate (%)	Dataset						
Test Set	Pure event	Flipping-augmented event	grayscale	Grayscale+ event (10)	Grayscale+ event (20)	Grayscale+ event (50)	Grayscale+ event (100)
Top1	49.46	52.15	52.69	44.09	54.84	49.46	57.00
Top5	79.57	83.87	88.17	89.25	90.32	88.17	90.32
Pushing	33.33	66.67	/	/	/	/	/
Walking	33.33	50	/	/	/	/	/

Table 1 Accuracy Rate on Test Set

quite difficult, let alone the variations in actions. For example, there are at least three variations in action “cough” (no hand movement/hands in front of chest/hands in front of the mouth). With more labelled training data, the accuracy rate can much likely increase. Earlier with even fewer training data, the accuracy rate is even lower, even for RGB training models. But the accuracy increases as we have labelled more data, but it took too much time in manual labelling.

When the movement of an action is not so obvious, the generated event polarities may not be enough for the model to distinguish among similar actions. Besides “bemuse”, there are several other actions (e.g., “use mobile”) with very few movements so that our model can be confused among them.

Grayscale images can compensate for above situation by combining data from two sensors. However, the hyperparameter of the superposition strength of event points on grayscale images is very important. If not carefully set, the superposition events can become noises, impairing the accuracy rate. From Table 1, we know that when we use a superposition of event binary frame on grayscale with strength 20 or 100, we can improve the Top1/Top5 accuracy. More needs to be done to investigate the reasons behind the settings of this hyperparameter. It is also worth noting that with flipping augmentation, pure event data can achieve Top1 accuracy very close to that of pure grayscale images.

6. Conclusions

In this paper, we introduce event-based camera and discuss its principles, advantages and disadvantages. From its data structure, we compare the size of event-based data and traditional frames. We studied multiple methods to represent event-based data and various new applications. The event datasets we used contain heterogenous actions including close-up and longshot actions, actions with few and large movements and direction-sensitive actions. We preprocess event data and train an action recognizer using TSM model. With data augmentation, the model trained fully by event data can be on par with that trained fully by grayscale data. Furthermore, by combining grayscale and event data we can improve the accuracy using a small training set.

References

[1] G. Gallego et al., "Event-Based Vision: A Survey," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 1, pp. 154-180, 1 Jan 2022, doi: 10.1109/TPAMI.2020.3008413.

[2] B. Son et al., “A 640x480 dynamic vision sensor with a 9µm pixel and 300Meps address-event representation,” in IEEE Intl. Solid-State Circuits Conf. (ISSCC), 2017

[3] P. Lichtsteiner, C. Posch, and T. Delbruck, “A 128×128 120 dB 15 µs latency asynchronous temporal contrast vision sensor,” IEEE J. Solid-State Circuits, vol. 43, no. 2, pp. 566–576, 2008

[4] A. Chadha et al., "Neuromorphic Vision Sensing for CNN-based Action Recognition," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 7968-7972, doi: 10.1109/ICASSP.2019.8683606.

[5] A. Zanardi et al., “Cross-modal learning filters for rgb-neuromorphic worm-hole learning,” in Proceedings of Robotics: Science and Systems, FreiburgimBreisgau, Germany, June 2019.

[6] A. Amir et al., "A Low Power, Fully Event-Based Gesture Recognition System," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7388-7397, doi: 10.1109/CVPR.2017.781.

[7] I. Ito, "Flipping Data Augmentation of Convolutional Neural Networks Using Discrete Cosine Transforms," 2021 29th European Signal Processing Conference (EUSIPCO), 2021, pp. 1501-1505, doi: 10.23919/EUSIPCO54536.2021.9616212.

[8] J. Lin, C. Gan and S. Han, "TSM: Temporal Shift Module for Efficient Video Understanding," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 7082-7092, doi: 10.1109/ICCV.2019.00718.

[9] Shao, H., Qian, S., & Liu, Y. (2020). Temporal Interlacing Network. Proceedings of the AAAI Conference on Artificial Intelligence, 34(07), 11966-11973. https://doi.org/10.1609/aaai.v34i07.6872

[10] MMAction2 Contributors. (2020). OpenMMLab's Next Generation Video Understanding Toolbox and Benchmark [Computer software]. https://github.com/open-mmlab/mmaction2

[11] W. Kay et al., "The Kinetics Human Action Video Dataset", 19 May 2017, arXiv:1705.06950, doi: https://doi.org/10.48550/arXiv.1705.06950

[12] Xiaolong Wang and Ross Girshick and Abhinav Gupta and Kaiming He, " Non-local Neural Networks", CVPR, 2018, NonLocal2018