Novel Design Ideas that Improve Video-Understanding Networks with Transformers

Yaxin Hu
Institute of Neuro- and Bioinformatics,
University of Luebeck[1]
Pattern Recognition Company GmbH[2]
Luebeck, Germany
yh@prcmail.de

Erhardt Barth
Institute of Neuro- and Bioinformatic,
University of Luebeck[1]
Luebeck, Germany

2024 International Joint Conference on Neural Networks (IJCNN) | 979-8-3503-5931-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/IJCNN60899.2024.10649969

***Abstract*—With the development of deep learning, video understanding has become a promising and challenging research field. In recent years, different transformer architectures have shown state-of-the-art performance on most benchmarks. Although transformers can process longer temporal sequences and therefor perform better than convolution networks, they require huge datasets and have high computational costs. The inputs to video transformers are usually clips sampled out of a video, and the length of the clips is limited by the available computing resources. In this paper, we introduce novel methods to sample and tokenize the input video, such as to better capture the dynamics of the input without a large increase in computational costs. Moreover, we introduce the MinBlocks as a novel architecture inspired by neural processing in biological vision. The combination of variable tubes and MinBlocks improves network performance by 10.67%.**

***Index Terms*—Video Transformer, Video Understanding, Action Recognition, Variable Tubes, MinBlocks, UniFormerV2**

I. Introduction

Action recognition is an important task in video understanding. The actions consist of movements, gestures, interactions, and activities usually conducted by humans. These actions may exhibit both strong intra- and inter-class variations. The same action can be performed by different people with different speeds in different scenarios. Some different actions may have some similar movement patterns or be performed in similar scenarios, these similarities make them difficult to be distinguished. Utilizing both spatial information and long-range temporal information is thus crucial for classification. Therefore, action-recognition networks must extract valid spatio-temporal features to make precise decisions. Since videos have an additional temporal dimension, the networks for videos processing require large computational resources. The balance between high accuracy and computational cost is another important topic for action recognition.

Convolutional neural networks (CNNs) have powerful spatial feature extraction capabilities and have achieved great results in image processing. However, CNNs have limitations in processing temporal sequences for video understanding. 3D-CNNs use convolution kernels extended to the time domain to extract temporal features. Thus, 3D-CNNs can aggregate local spatio-temporal context from a rather small 3D neighborhood and thus capture local dependencies. Because of the limited receptive field, however, 3D-CNNs are less effective in capturing global dependencies.

Transformers have led to large progress in both language and image processing. They can effectively capture long-time dependencies by self-attention mechanisms. In recent years, visual transformers have been the state of the art on video action recognition benchmarks. There are different transformer variants that achieve high performance on video datasets. Nevertheless, transformers have only limited capabilities to reduce local redundancies since they are "blindly" comparing the input tokens. Moreover, the requirements of huge datasets and computing resources are limiting their development, since only large labs with sufficient computation resources can afford training larger transformer networks.

The UniFormerV2 [1] is one of the most successful transformer variant. It combines 3D convolution and spatio-temporal self-attention to reduce the local redundancy and also capture long-time dependencies in videos. Moreover, the incorporation of visual transformers (ViTs) pre-trained on the large dataset CLIP-400M [2] significantly reduces the computation load and enhances the spatial modeling. We therefore choose the UniFormerV2 as our reference architecture that we further improve. However, although using pre-trained ViT weights, UniFormerV2 networks still need to be trained on huge video data sets for reaching state-of-the art performance. Therefore, our aim it to demonstrate the improvements obtainable with our novel designs on only smaller datasets.

Motivations: A first motivation is to find a way to utilize longer temporal sequences of videos without adding significant computational costs. This could avoid some of the limitations of transformer networks, which are currently not trainable for most users and applications. A second motivation is to introduce a bio-inspired nonlinear connection between neurons that has been successful in image recognition.

Contributions: Given the above motivations,

- We design a novel way to leverage more temporal information by sampling the red, green, and blue channels of the video at different times.

This project received funding from European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No.955590.

- Inspired by the Inception Network [3], we use different sizes of 3D kernels (tubes) to tokenize the videos such as to obtain temporal information based on different time intervals. Such, the network obtains richer temporal information without increasing the hidden dimensions.
- We introduce the MinBlock as a novel, bio-inspired, and nonlinear connection layer, which is added after the 3D embedding layer or inserted into Local UniBlocks.

II. Related Work

The literature on action recognition is vast, starting with the use of hand-crafted features together with standard classifiers [4]. The typical hand-crafted features are Space–Time Interest Points (STIP) [5], Histograms of Oriented Gradient (HOG) [6], Histogram of Optical Flow (HOF) [7], Histogram of Motion Boundary(HoMB) [7], and many more. Hand-crafted features are easier to understand and can obtain good results on small datasets. However, the features are not specific to a particular dataset and therefore less effective.

Convolutional Neural Networks (CNNs), have been adapted for videos processing since they achieved good results in image processing. Since videos have the additional temporal dimension, the key question is, how CNNs can learn to extract temporal information from videos. One approach is to use two-stream networks [8], a spatial stream to capture spatial information from raw frames of videos and a temporal stream to capture temporal information, e.g. by using optical flow [9]. Using Recurrent Neural Networks (RNNs) like Long-Short-Temporal Memory (LSTM) is also a popular way to model temporal information. Often CNNs and RNNs are combined to capture spatio-temporal information [10]. CNNs first extract spatial features from input frames, and then RNNs extract temporal features, and in a next step the features would be combined and fed into a Multilayer Perceptron (MLP) for prediction. However, RNNs have a rather short temporal memory and fail to capture long-range temporal information. A straightforward approach is to extend the 2D kernels of CNNs to 3D such that 3D-CNNs [11] can capture spatio-temporal information by using 3D kernels. One benefit of 3D-CNNs is that the spatial 2D weights of the 3D kernels can be pre-trained on image recognition tasks. Using this approach, various 3D-CNNs, such as I3D [12], ResNet3D [13], R(2+1)D [14] and X3D [15], were designed and achieved good performance on video datasets. However, 3D CNNs have the same limitations as RNNs, i.e., they can only model short-range temporal information.

Along with the great success of transformers in natural language processing tasks, transformers have also been applied for visual tasks and have achieved impressive results. Based on the breakthrough of visual transformers on image-based tasks, transformers have been also applied to video and have quickly become the state-of-the-art for almost all video benchmarks. A variant called VTN [16] adds a temporal attention-based encoder on top of a 2D spatial feature extraction model, and uses a MLP for classification. Timesformer [17] employed five space-time self-attention schemes and found an optimal accuracy-computation trade-off. ViviT [18] adopted a transformer based on a pretrained ViT [2], and tubelet embedding on video clips; and used four factorized designs of spatio-temporal attention to reduce complexity. MViT [19] uses a channel-resolution scale strategy to progressively expand the channel capacity while pooling the spatio-temporal resolution; and a Multi Head Pooling Attention (MHPA) to enable flexible resolution modeling. MTV [20] uses multi-view encoders to extract tokens from spatio-temporal tubelets of varying dimensions from the input video with a cross-view fusion, then a global encoder to produce the final token for prediction. Swin Transformers [21] are based on 3D Shifted Windows to introduce cross-window connections for utilizing the spatio-temporal locality of videos. UniFormer V2 [1] combined 3D CNNs with transformers to capture both local and global information and used pre-trained weights from a ViT [2]. TubeViT [22] uses sparse and differently sized video tubes that convert ViT encoders to efficient video models that work seamlessly with images and videos. VideoMAE [23] employs pre-training by randomly masking most patches and reconstructing them pixel by pixel; this self-supervised method improves generalization performance. The mainstream is currently defined by multi-modality transformers, such as InternVideo [24], OmniVec [25].

III. Methodology

In this section, we introduce the methods underlying our novel design ideas. A new method called RGBt Sampling samples temporal RGB frames at different times to extract local dynamics. Tubes of different dimensions are used to embed richer temporal information into the tokens. The MinBlock architecture implements a bio-inspired nonlinear connection between neurons; it is here extended to video processing and we investigate the best placement of MinBlocks within a UniFormerV2 network.

A. RGBt Sampling

For almost all video-transformer architectures, the inputs are obtained by sampling a number of clips from each video. Often, a video is uniformly divided into several segments along the temporal dimension, then one frame is randomly selected from each segment. All selected frames then form a clip to represent the video. This seems reasonable, since videos are proven to have high redundancy and temporal correlation [23].

Due to the limitation of computational resources, the number of used frames is limited, but usually increasing the number of frames will improve performance. Especially for datasets in which the actions are motion-related (such as Something-Something V2), more temporal information will improve the performance a lot. To maintain a good balance between performance and computational costs, we propose to sample the 3 color channels at different times. The idea is to sample N*3 frames rather than the original N frames from each video by using R_{i-1}, G_i, and B_{i+1} of the consecutive selected frames instead of using R_i, G_i, and B_i from the same frame -

see Fig. 1. Thus, we introduce additional temporal information without increasing the input size.

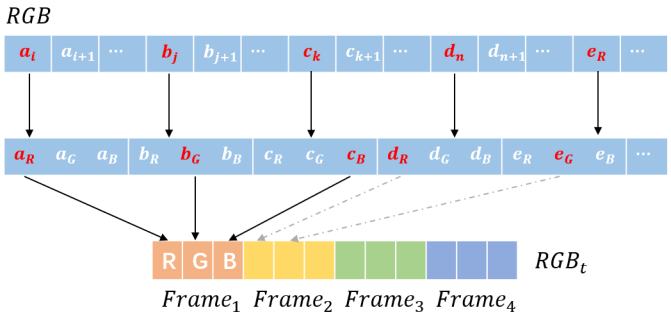


Fig. 1. RGBt Sampling

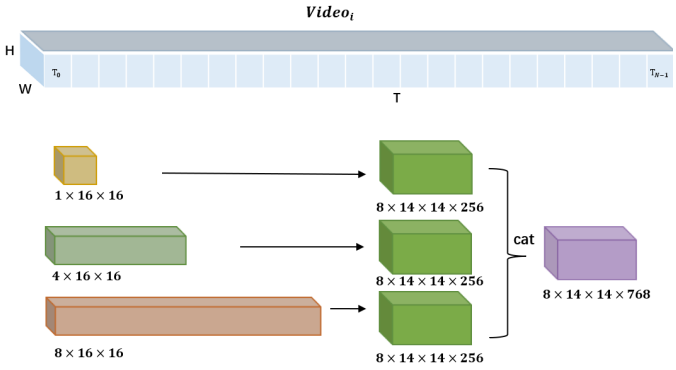
B. Tokens based on Tubes of Variable Size

The state-of-the-art performance of the UniFormerV2 network is largely due to using ViT weights pre-trained on the large CLIP400 dataset. This advantage, however, also introduces some limitations: the UniFormerV2 has to keep the structure of the ViT, and then insert its own blocks into the ViT structure. As a result, the first layer of 3D tokens is quite redundant since it has to keep the same channel dimension of 768 as the ViT in order to be able to use the pre-trained weights. Inspired by the Inception network, we can use different 3D kernels to tokenize the video clips and then concatenate the different feature maps in their channel dimension. In this case, temporal information from different frames is fused by using differently sized tubes. Therefore, the resulting tokens span different time frames and contain richer information about the dynamics of the actions.

As shown in Fig. 2, we first sample a video to form an input with dimension 32 × 224 × 224 × 3 (T × H × W × C). Then we use three different tubes to tokenize the video, and obtain three outputs with the same shape: 8 × 14 × 14 × 256 (T_1× H_1 × W_1 × C_1). Next, we concatenate the three outputs in the channel dimension to form the final tokens, which have the same channel dimension of 8 × 14 × 14 × 768 as the ViT architecture.

C. Bio-inspired MinBlock as additional layer

As a variant of FP-nets [26], Min-Nets are inspired by end-stopped cortical cells with units that output the minimum of two learned filters [27], [28]. In [26] and [28] it has been shown that the addition of Minblocks improves the object-recognition performance of state-of-the-art CNNs, and also makes the CNNs more robust. FP-nets and Min-Nets are particular variants of eNets [28], i.e., networks that employ the bio-inspired principle of end-stopping. We here generalize eNets to video by combining not two units but three pairs of two units. Such computations are related to optical flow computation [29] and also to the way biological neurons process motion information [30], [31]. The geometrical motivation is based on the fact that the curvature of a 3-dimensional manifold defines the structure of the manifold and is captured



Tubes	Kernel	Stride	Padding	Output
1	1 × 16 × 16	4 × 16 × 16	0 × 0 × 0	8 × 14 × 14 × 256
2	4 × 16 × 16	4 × 16 × 16	0 × 0 × 0	8 × 14 × 14 × 256
3	8 × 16 × 16	4 × 16 × 16	2 × 0 × 0	8 × 14 × 14 × 256

Fig. 2. Differently sized tubes for tokenization

by the invariants of the Riemann curvature tensor based on the sum of 3 pairwise combinations of the derivatives [29], [30].

As shown in Fig. 3, we add three 1×1×1 convolutional layers to process the former feature maps. A 1×1×1 convolutional layer is used as a channel-wise filter and creates a one-to-one projection of the feature maps to extract features across channels. Then minimum functions are used to element-wise combine the feature maps learned by depth-wise convolutions. The pairwise minimum operations make the neurons more selective and more robust than classical neurons [27].

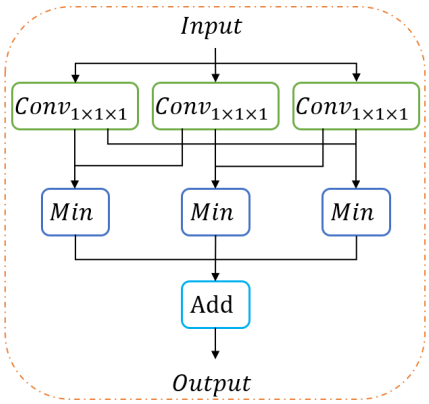


Fig. 3. The structure of a MinBlock.

As reported in [27], MinBlocks can improve the performance of convolutional neural networks such as ResNet and DenseNet on image datasets. Here, we insert MinBlocks at two different places in the backbone network where the convolutions are performed. One position is after the 3D convolutional tokenization layer (as shown in Fig. 4), the other is inside the Local UniBlock (as shown in Fig. 5).

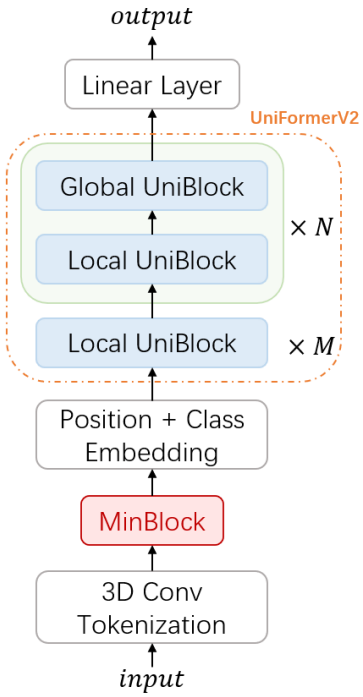


Fig. 4. Inserting MinBlock after the tokenization layer.

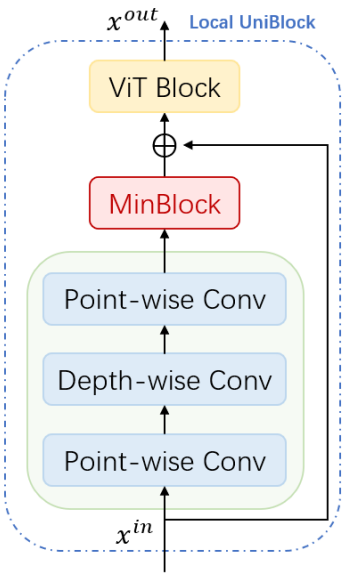


Fig. 5. Inserting MinBlock inside the local UniBlock.

IV. Experiments

A. Datasets

We choose the UCF101 and SthSth32 datasets to test our novel design ideas. UCF101 [32] is a small video dataset for action recognition. The actions are defined to a significant degree by spatial information. It has 101 different action classes, and it consists of around 9.5k training videos and 3.7k validation videos. We select around 1.6k videos from the training set for validation, and use the original validation videos as test set.

The actions in Something-Something V2 [33] are more related to scene dynamics and thus require more temporal information for correct predictions. SthSth32 is a subset of the Something-Something V2 dataset [33] containing only 32 classes selected to reduce computational costs. The resulting data set contains about 41k training videos, 6.1k validation videos and 6.2k test videos - see Appendix for more details.

Since pre-trained weights are not available for the RGBt frames, we train the network from scratch. Both UCF101 and SthSth32 are not large enough to train transformer networks to maximum performance, we here therefore focus on the relative differences in performance.

B. Implementation Details

We conduct all experiments on 4 NVIDIA A100 40G GPUs. All frames are resized in a jittering scale range [240, 320] and randomly cropped to 224x224 pixels for training. All frames are directly resized to 224x224 pixels for inference. Our training batch size is 256 and test batch size is 128. We use the AdamW [34] optimizer to learn model parameters and follow the training recipe in [1], a cosine learning rate schedule [35] with a linear warm-up strategy for the first 5 epochs. Our warm-up start learning rate and cosine end learning rate are both 1e-6, and the base learning rate is 1e-5. The momentum and the weight decay are set to 0.9 and 0.05 respectively.

C. Results and Discussion

As shown in Table I, for UCF101, using RGBt sampling leads to a 3.2% top-1 accuracy improvement compared to the original RGB baseline without increasing FLOPs and the number of parameters. Using differently sized tubes with RGB frames leads to a 6.85% higher top-1 accuracy with slightly more FLOPs and parameters. By inserting MinBlocks we obtain a 2.34% top-1 gain in accuracy.

TABLE I
Comparison of RGB, RGBt, RGB tubes and MinBlock on UCF101

Method	#Frames	Param.(M)	FLOPs(G)	Top1	Top5
RGB	8	123.82	157.41	43.67	69.84
RGB_t	3*8→8	123.82	157.41	46.87	74.23
RGB tubes	4*8→8	125.78	160.50	50.52	79.54
RGB Min*	8	145.08	190.71	46.01	72.83

[1] 3*8→8 means 3*8 frames are used to form 8 RGBt frames;
[2] 4*8→8 means 4*8 frames are used for tokenization;
[3] Min* reports MinBlock inserted inside Local UniBlock.

For the SthSth32 dataset, Table II shows that the top-1 accuracy obtained by using RGBt sampling is 5.75% higher than with RGB sampling. The top-1 accuracy is improved by 6.77% when using different tubes with RGB frames. The network achieves a 1.3% higher performance gain by adding MinBlocks.

The results shown in Table I and Table II indicate that our designs can improve the performance for both datasets (see

TABLE II
COMPARISON OF RGB, RGBT, RGB TUBES AND MINBLOCK ON STHSTH32

Method	#Frames	Param.(M)	FLOPs(G)	Top1	Top5
RGB	4	123.76	78.72	44.91	77.49
RGB_t	3*4→4	123.76	78.72	50.66	81.94
RGB tubes	4*4→4	125.73	80.26	51.68	83.14
RGB Min*	4	145.03	95.37	46.21	79.02

[1] 3*4→4 means 3*4 frames are used to form 4 RGBt frames;
[2] 4*4→4 means 4*4 frames are used for tokenization;
[3] Min* refers to MinBlocks inserted inside the Local UniBlock.

overview in Fig. 6). RGBt sampling helps to obtain a longer dynamic representation of videos without increasing FLOPs and the number of parameters. Tubes of variable size can embed richer temporal information. Inserting MinBlocks also improves performance, presumably by making the neurons more selective.

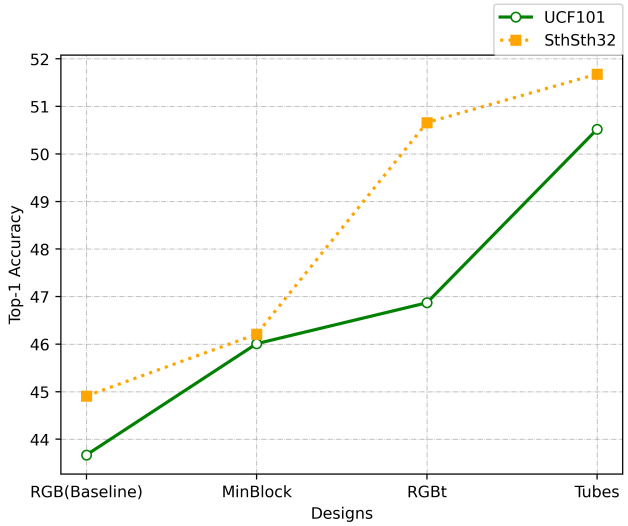


Fig. 6. Top-1 Accuracy improvements by our designs on UCF101 & SthSth32

We performed ablation experiments on UCF101 to show that adding the MinBlocks improves performance and tested two different positions of where to insert the MinBlocks. As discussed in Section III, one position is after the 3D convolution tockenization layer and the other is inside the Local UniBlock. As shown in Table III, RGB sampling with MinBlocks inserted after the tokenization layer improve performance by 0.26%, while MinBlocks inserted inside the local UniBlock lead to an improvement of 2.34% of top-1 accuracy. In Table I and Table II, we therefore report performances for MinBlocks inserted inside the local UniBlock.

Inspired by the Temporal Segments Networks (TSN) [36], we used both RGB and RGBt frames and fused them at a later stage. Table IV shows that this leads to a 4.63% performance gain compared to the using only RGB frames and a 1.43% top-1 improvement compared to using only RGBt frames.

TABLE III
OVERVIEW OF COMPARISON OF DIFFERENT MINBLOCKS WITH DIFFERENT POSITIONS ON UCF101

Method	Position	Param.(M)	FLOPs(G)	Top1	Top5
RGB Min	Token	125.59	160.19	43.93	70.08
RGB Min	Local	145.08	190.71	46.01	72.83

[1] Min indicates the use of MinBlocks;
[2] Token: MinBlock inserted after the tokenization layer as in Fig. 4;
[3] Local: MinBlock inserted inside the local UniBlock as in Fig. 5.

Since both RGBt sampling and differently-sized tubes improve the network's performance, we ran experiments using both simultaneously. Results are shown in Table IV: using RGBt sampling and differently-sized tubes leads to a 9.55% performance gain relative to the baseline. Moreover, we also combined variable tubes with MinBlocks; this leads to a 10.67% improvement. Finally, the combination of RGBt and MinBlocks also leads to higher accuracy (49.08%) than both only RGBt (46.87%) and only MinBlocks (46.01%). Hence, the results in Table IV show that combinations of our design elements can further improve the performance of the network.

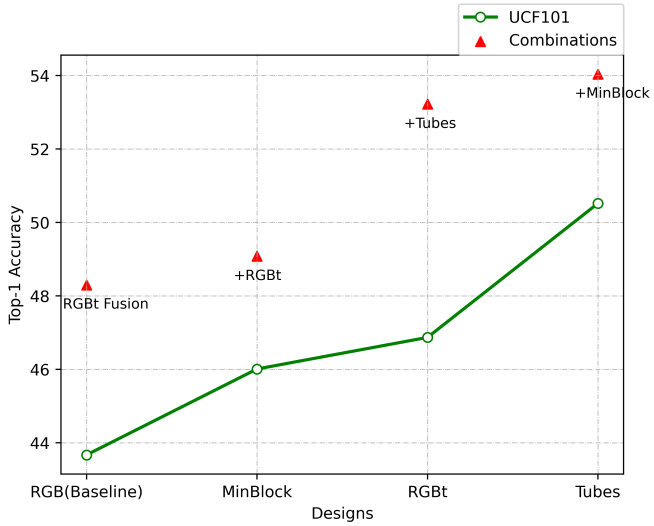


Fig. 7. Top-1 Accuracy improvements by different combinations on UCF101

TABLE IV
EXTRA EXPERIMENTS ON UCF101

Methods	Position	Top1	Top5
RGB&RGB_t	-	48.30	74.25
RGB_t tubes	Token	53.22	79.60
RGB tubes & Min	Local	54.04	80.63
RGB_t Min	Local	49.08	75.42

V. CONCLUSIONS

We have explored some novel design ideas for video understanding networks. The first idea, that we call RGBt frames,

is to sample the colors at different times such as to provide more temporal information without increasing the number of parameters and the computational load. The second idea is to use differently sized tubes with temporal dimensions of 1, 4, and 8 to obtain richer temporal information. The third idea is to introduce a novel type of neuron based on pairwise minimum operations on traditional neurons (the MinBlocks). Ablation experiments have been done to find an optimal placement of the MinBlocks.

We have shown that all three design elements lead to better classification performance on two action-recognition datasets. Moreover, we could show that combinations of the above design elements can further improve performance.

We have trained the networks from scratch and have focused on relative improvements. We can, however, expect that the ideas would lead to better than state-of-the art performance once the networks are pre-trained and trained on large datasets.

ACKNOWLEDGMENT

We thank Philipp Gruening for his advice, Marius Jahrens and Hans-Oliver Hansen for their help in setting up and allocating the GPUs. This project received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 955590.

REFERENCES

[1] K. Li, Y. Wang, Y. He, Y. Li, Y. Wang, L. Wang, and Y. Qiao, "Uniformerv2: Spatiotemporal learning by arming image vits with video uniformer," *arXiv preprint arXiv:2211.09552*, 2022.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[4] Y. Zhu, X. Li, C. Liu, M. Zolfaghari, Y. Xiong, C. Wu, Z. Zhang, J. Tighe, R. Manmatha, and M. Li, "A comprehensive study of deep video action recognition," *arXiv preprint arXiv:2012.06567*, 2020.

[5] I. Laptev, "On space-time interest points," *International journal of computer vision*, vol. 64, pp. 107–123, 2005.

[6] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3551–3558.

[7] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part II 9*. Springer, 2006, pp. 428–441.

[8] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *Advances in neural information processing systems*, vol. 27, 2014.

[9] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[10] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. Snoek, "Videolstm convolves, attends and flows for action recognition," *Computer Vision and Image Understanding*, vol. 166, pp. 41–50, 2018.

[11] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4507–4515.

[12] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[13] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.

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

[15] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 203–213.

[16] D. Neimark, O. Bar, M. Zohar, and D. Asselmann, "Video transformer network," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 3163–3172.

[17] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?" in *ICML*, vol. 2, no. 3, 2021, p. 4.

[18] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6836–6846.

[19] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6824–6835.

[20] S. Yan, X. Xiong, A. Arnab, Z. Lu, M. Zhang, C. Sun, and C. Schmid, "Multiview transformers for video recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 3333–3343.

[21] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu, "Video swin transformer," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 3202–3211.

[22] A. Piergiovanni, W. Kuo, and A. Angelova, "Rethinking video vits: Sparse video tubes for joint image and video learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2214–2224.

[23] Z. Tong, Y. Song, J. Wang, and L. Wang, "Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training," *Advances in neural information processing systems*, vol. 35, pp. 10 078–10 093, 2022.

[24] Y. Wang, K. Li, Y. Li, Y. He, B. Huang, Z. Zhao, H. Zhang, J. Xu, Y. Liu, Z. Wang *et al.*, "Internvideo: General video foundation models via generative and discriminative learning," *arXiv preprint arXiv:2212.03191*, 2022.

[25] S. Srivastava and G. Sharma, "Omnivec: Learning robust representations with cross modal sharing," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1236–1248.

[26] P. Grüning, T. Martinetz, and E. Barth, "Fp-nets as novel deep networks inspired by vision," *Journal of Vision*, vol. 22, no. 1, pp. 8–8, 2022.

[27] P. Grüning and E. Barth, "Bio-inspired min-nets improve the performance and robustness of deep networks," *arXiv preprint arXiv:2201.02149*, 2022.

[28] P. Grüning and E. Barth, "Efficient coding in human vision as a useful bias in computer vision and machine learning," *Journal of Perceptual Imaging*, vol. 6, pp. 1–10, 2023.

[29] E. Barth, "The minors of the structure tensor," in *Mustererkennung 2000*, G. Sommer, Ed. Berlin: Springer, 2000, pp. 221–228.

[30] E. Barth and A. B. Watson, "A geometric framework for nonlinear visual coding," *Optics Express*, vol. 7, no. 4, pp. 155–165, 2000.

[31] E. Barth, "A geometric view on early and middle level visual coding," *Spatial Vision*, vol. 13, no. 2–3, pp. 193–199, 2000.

[32] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[33] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The" something something" video database for learning and evaluating visual common sense," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5842–5850.

[34] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," 2018.

[35] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[36] 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 *European conference on computer vision*. Springer, 2016, pp. 20–36.

APPENDIX

TABLE V
THE 32 SELECTED CLASSES OF STHSTH32

Class names	Index
Approaching [something] with your camera	0
Closing [something]	1
Folding [something]	2
Holding [something]	3
Holding [something] next to [something]	4
Moving [something] away from [something]	5
Moving [something] away from the camera	6
Moving [something] closer to [something]	7
Moving [something] down	8
Moving [something] towards the camera	9
Moving away from [something] with your camera	10
Opening [something]	11
Picking [something] up	12
Plugging [something] into [something]	13
Pretending to pick [something] up	14
Pretending to put [something] next to [something]	15
Pretending to put [something] on a surface	16
Pretending to take [something] from [somewhere]	17
Pushing [something] so that it slightly moves	18
Pushing [something] with [something]	19
Putting [something] into [something]	20
Showing a photo of [something] to the camera	21
Showing that [something] is empty	22
Stacking [number of] [something]	23
Throwing [something] against [something]	24
Turning [something] upside down	25
Turning the camera downwards while filming [something]	26
Turning the camera left while filming [something]	27
Turning the camera right while filming [something]	28
Turning the camera upwards while filming [something]	29
Uncovering [something]	30
Unfolding [something]	31

Selected from the 40-selected classes reported in [33], from Sth-Sth V2