http://dx.doi.org/10.21622/ACE.2022.02.1.001

Modern Architectures Convolutional Neural Networks in Human Activity Recognition

H. Mahmoud

Building Physics and Environmental Research Institute
Housing and Building National Research Center,
Cairo, Egypt

Email: {hanan252000@yahoo.com}

Received on, 28 December 2021 - Accepted on, 20 February 2022 - Published on, 10 May 2022

ABSTRACT

In recent years, many researchers have focused on using convolutional neural networks to perform human activity recognition as evidenced by the emergence of a number of convolutional neural network architectures such as LeNet-5, AlexNet and VGG16 and modern architectures such as ResNet, Inception V3, Inception-ResNet, MobileNetV2, NASNet and PNASNet. The main characteristic of a convolutional neural network (CNN) is its ability to extract features automatically from input images, which facilitates the processes of activity recognition and classification. Convolutional networks indeed derive more relevant and complex features with every additional layer. In addition, CNNs have achieved perfect classification on highly similar activities that were previously extremely difficult to classify. In this paper, the researcher evaluated modern convolutional neural networks in terms of their human activity recognition accuracy, and she compared the results with the state-of-the-art methods. In this research, the researcher used two public data sets, HMDB (Shooting gun, kicking, falling to the floor, and punching) and the Weizman dataset (walking, running, jumping, bending, one hand waving, two-hand waving, jumping in place, jumping jack, and skipping). The experimental results indicated that the CNN with NASNet architecture achieves the best performance of the six CNN architectures on both human activity data sets (HMDB and Weizman).

I. INTRODUCTION

The use of deep learning methods in human activity recognition has become the focus of many researchers. The strength of deep learning lies in its ability to extract features automatically in a task-dependent manner. It avoids reliance on heuristic handcrafted features and scales better, making it suitable for more complex behavior recognition tasks. Furthermore, the convolution neural network is sufficiently fast to online human activity recognition. A large number of deep learning techniques have been developed and successfully applied to recognition tasks. Szegedy from Google, Inc. proposed the Inception architecture for image classification [1]. He and Zhang from Microsoft introduced residual units in residual networks [2], and Szegedy and Ioffe combined residual connections with the Inception architecture to create the Inception V4 or Inception- ResNet architecture [3]. Subsequently, Zoph and Vasudevan from Google introduced the idea of building the architecture on a small data set and then transferring the result block to another, larger data set. They applied this idea to introduce NASNet [4]. Liu and Zoph used the idea behind NASNet and modified it to create the progressive PNASNet [5]. Finally, Sandler and Howard used the idea of deep convolution layers to build MobileNet, which can be used to make smaller models more efficient [6].

Volume 2, Issue 1, June 2022 - ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2022.02.1.001

Many researchers have applied deep learning in human activity recognition based on its ability to extract features automatically in a task-dependent manner. Plotz, Hammerla, and Olivier discussed the utilization of a few feature learning methods, including deep learning in activity recognition systems [7]. Zeng et al. demonstrated an algorithm for human activity recognition using mobile sensors [8]. Yosins and Jeffclune demonstrated a method to quantify feature transferability from each layer of a neural network. They showed that transferability is negatively affected by two issues: optimization difficulties, which is caused by splitting networks in the middle of fragilely co-adapted layers and the specialization of higher layer features to the original task at the expense of performance on the target task [9]. Yang and Nguyen proposed a method to build a new deep architecture for CNNs to investigate multichannel time series data. The advantages of this method are that it performs feature extraction in a task-dependent manner and that the extracted features have discriminating power with respect to the classes of human activities [10]. Zeng and Menshoel proposed a CNN-based feature extraction approach that extracts scale-invariant and locally dependent characteristics from an acceleration time series. The CNN-based approach outperformed the previous state-of-the-art approaches [11].

Most human activity recognition techniques using CNNs are sensor-based techniques that require wearable sensors to be attached to the limbs and torso of a person. However, this solution is impractical for human activity recognition. Instead, the only useful way of recognizing human activity in public places is through surveillance videos. Sensor-based methods can be used for older people in homes to discover cases of faintness or falling but cannot be used in public places. Thus, the only available approach is to analyze surveillance videos.

There is a problem when comparing different approaches of human activity recognition because of differences in the data pre-processing operations, data sets, segmentation techniques and classification models. The contribution of this paper is that it studied the effectiveness of using CNNs for human activity recognition and found these techniques to be dependent on the quality of the images used in terms of individual or group activity. This study enables the researcher to evaluate and compare the different CNN architectures used for human activity recognition. She applies her framework to conduct comparative studies on two public data sets: HMDB and Weizman. The remainder of this paper is organized as follows: Section 1 provides an introduction. Section 2 describes the various methods. Section 3 presents the experimental results, and Section 4 concludes the paper.

II. CONVOLUTIONAL NEURAL NETWORKS

In 2012, the convolutional neural network architecture called AlexNet proved successful at a large variety of computer vision tasks such as object detection [12], segmentation [13], video classification [14], human pose estimation [15], object tracking [16] and super resolution [17]. These successes encouraged researchers to find even better-performing convolutional neural networks. In 2014, researchers utilized deeper and wider networks to improve the quality by building VGG [18] and GoogLeNet [19], which achieved high performance in the ILSVRC-2014 classification challenge. However, they found that executing classical convolutional neural networks such as AlexNet, VGG and GoogleNet requires numerous calculations; for example, AlexNet used 60 million parameters, VGG used 20 million parameters and GoogleNet employed 5 million parameters. Consequently, in recent years, researchers have focused on finding new CNN architectures to reduce the computational cost. Many new architectures have appeared, such as ResNet, Inception, Inception-ResNet, MobileNet, NASNet and PNASNet.

Journal of Advances in Computing and Engineering (ACE) Volume 2, Issue 1, June 2022 - ISSN 2735-5985
http://dx.doi.org/10.21622/ACE.2022.02.1.001

A) Inception

The main objectives when building optimal convolutional neural networks is to obtain high performance with low computational overhead. The inception architecture [1] has proven to be highly tunable because the researcher was able to change the number of filters in the various layers to optimize the training speed without affecting the quality of the trained network. However, the layer sizes must be tuned carefully to balance the computational burden of the various models and their sub-networks. One of the main advantages of the inception architecture is that it allows the number of units to increase at each stage without introducing an uncontrolled computational complexity. Additionally, the researcher can avoid computational difficulties by determining the width of each stage and choosing the appropriate number of stages to limit the use of computational resources. Another benefit of the inception architecture is that it intuitively aligns with the idea that visual information should be processed at various scales and then aggregated so that the next stage can abstract features from the different scales simultaneously.

The network includes different modules such as a naive module and a module to perform dimension reduction; these modules are stacked upon each other with a stride of 2 max pooling layers that halve the resolution of the grid. During training, to maintain efficient use of memory, inception modules are used only for the higher layers, while the lower layers are treated in a traditional convolutional fashion. It is preferable to use an incorporation module with dimension reduction over a naive module because the naive version suffers from one large problem: its 5 **x** 5 convolutions can be expensive on top of a convolutional layer with a large number of filters even when the number is fairly modest.

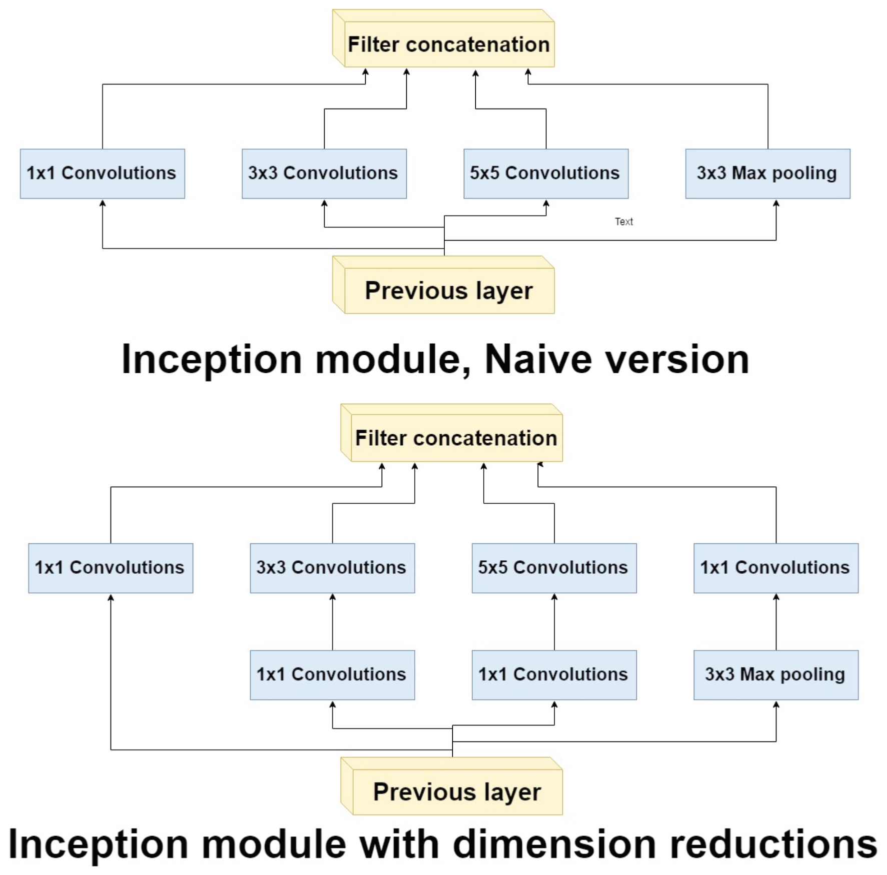


Fig. 1. Inception module Naive version and dimension reductions version

http://dx.doi.org/10.21622/ACE.2022.02.1.001

B) Inception V3

Inception V3 was introduced in 2015. It uses the inception block introduced in GoogleNet. ImageNet reduced the top-5 error rate (The Top-5 error is the portion of test images for which the correct label is not among the five labels the model considers most likely) to 5.6% (for a single model) and to 3.6% (for an ensemble model). It uses Normalization, Image distortions as open issues and RMS prop for gradient descent. It has 25 million parameters and is trained on 8 GPUs for 2 weeks. Its deep architecture consists of inception blocks. All the operations inside the inception blocks use a stride of 1 and sufficient padding to output the same spatial dimensions (W **x** H) as the feature map. Four different feature maps are concatenated on depth at the end. Inside the inception blocks, the researcher used a number of filters such as 5**x**5, 3**x**3, 1**x**1 for the convolution and pooling layers, and she added the input to the output using a single 1**x**1 convolution. Using different filter dimensions allows her to capitalize on all the features at the same time. Figure 1 shows the Inception Module Naive version.

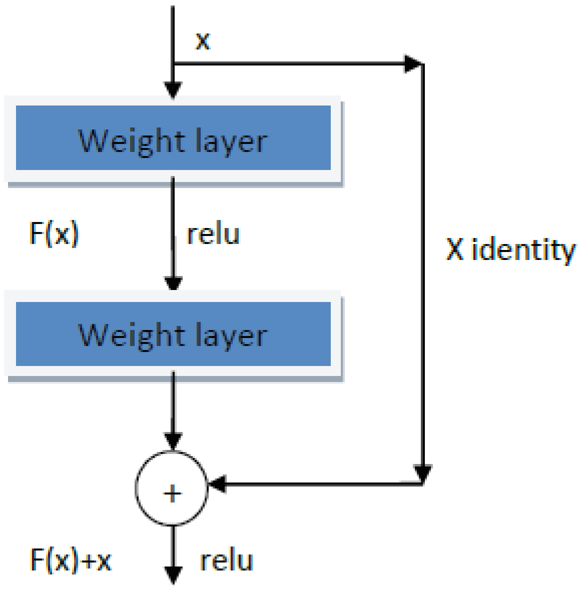


Fig. 2: Residual block

C) ResNet

Researchers designed ResNet [2] to solve the problem that as network depth increases, the accuracy first becomes saturated and then degrades rapidly. A ResNet block is either 2 layers deep (which is used in small networks such as ResNet 18 and 34) or 3 layers deep (used in larger networks such as ResNet 50, 101 and 152). Figure 2 shows a residual block. The ResNet network converges faster than its plain counterpart. ResNet 34 achieved a top-5 validation error rate of 5.71% which was better than BN-Inception and VGG. Res Net-152 achieved a top-5 validation error rate of 4.49%. An ensemble of 6 models with different depths achieved a top-5 validation score of 3.57%. ResNet 152 includes 152 layers. Figure 3 shows ResNet152 architecture, a few 7x7 convolutional layers and the rest are 3x3, batch normalization, max and average pooling layers. In total, this model has 60 million parameters and requires training on 8 GPUs for 2-3 weeks. The output channels are created by adding a small delta F(x) to the original input channels x, and F(x) is represented as a weight layer followed by ReLU activation and one weight layer. In this way, thousands of layers can be stacked, and the gradients do not vanish. Table I shows the Top-5 error rate for different ResNet architectures.

TABLE I: Top5 error for different ResNet architectures

ResNet architectures	Top5 Error
ResNet 34	5.71%
ResNet 50	5.25%
ResNet 101	4.6%
ResNet 152	4.49%

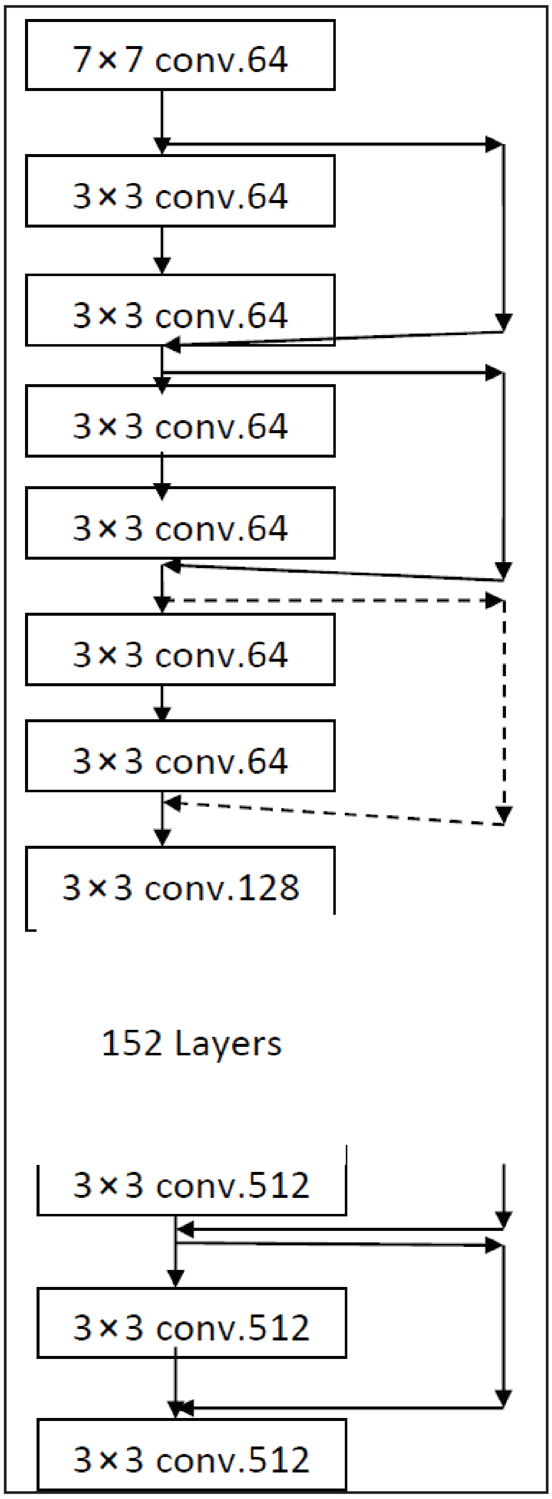


Fig. 3. ResNet152

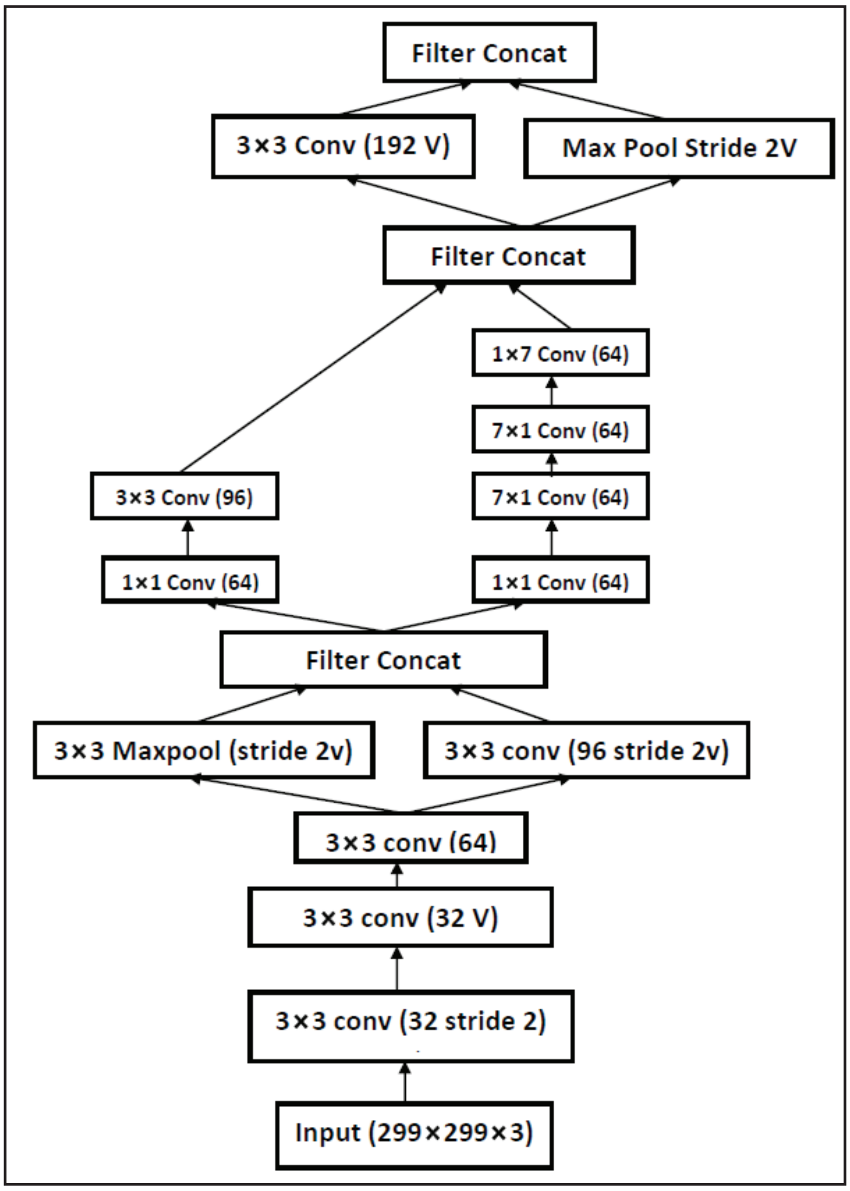


Fig. 4. Inception-ResNet architecture

D) Inception-ResNet

This architecture is the result of combining the inception architecture with residual connections [3]. By empirical evidence, the researchers found that training with residual connections accelerates the training of an inception network. Several versions of residual inceptions exist, but when the number of filters exceeded 1,000, instabilities appeared in residual variants, the network died early in the training, and after a few tens of thousands of iterations, the last layer before the average pooling started producing only zeros. Reducing the learning rate or adding extra batch normalization did not solve these problems. Scaling the residuals by factors between 0.1 and 0.3 before adding them to the accumulated layer activations seemed to stabilize the training. Figure 4 shows the Inception-ResNet architecture.

E) MobileNetV2

The architecture of MobileNetV2 [6] contains an initial fully convoluted layer with 32 filters followed by 19 residual bottleneck layers, as described in Table II. ReLU6 is adopted as the activation function, and a 3 x 3 kernel size is utilized as is standard in modern networks. Dropout and batch normalization are also used during training. The networks have a constant expansion rate except for the first layer. When the expansion rates are between 5 and 10, the performance curves are nearly identical. Larger networks achieve slightly better performance with a larger expansion rate but smaller networks perform better with slightly smaller expansion rates [6]. In MobileNetV2, the convolution is split into two layers. The first is called a depthwise convolution, and the second is called a pointwise convolution. The depthwise convolution applies a single convolutional filter per input channel to perform lightweight filtering. The pointwise convolution computes a linear combination of the input channels to buildnew features. Figure 5 shows MobileNetV2 architectures.

TABLE II:
MobileNetV2: Each row describes a sequence of 1 or more identical layers, repeated n times. In the same sequence, all layers have the same number of output channels c. All layers use a stride of 1 except the first layer, which has a stride of s. The expansion factor t is always applied to the input size. All the spatial convolutions use 3 x 3 kernels. (this cannot be a title for the table; just use heading and keep the explanation to the body of the research)

Input	Operator	t	c	n	s
$224^2 \times 3$	Conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	Conv2d 1×1	-	1280	1	1
$7^2 \times 1280$	Avgpool 7×7	-	-	1	
$1 \times 1 \times 1280$	Conv2d 1×1	-	k		

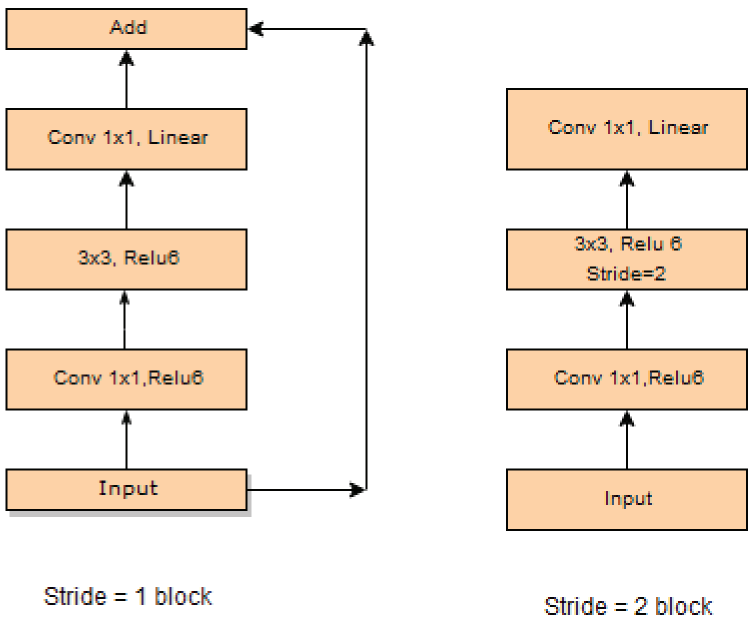


Figure 5. MobileNetV2 architectures

F) NASNet

NASNet [4] development began when Google AI introduced the AutoML project to automate the design of machine learning models. They found that AutoML can design small neural networks that perform at levels equal to those of neural networks designed by human experts. These small neural networks have achieved strong performances on small academic data sets such as CIFAR-10 (Data set consists of 60000 (32 x 32) colour images in 10 categories with 6000 images per category) and Penn Treebank. The designers also tried to apply the AutoML method to larger data sets such as ImageNet image classification and COCO object detection, which are two of the largest data sets for computer vision. However, the designers found that if naively applying AutoML directly to ImageNet required many months of training; therefore, they redesigned the search space so that AutoML can find the best layer and then repeat it many times to create a final network.

The designers performed architectural searches on CIFAR10 and transferred the best architecture to ImageNet image classification and COCO object detection. Using this approach, AutoML can find the layers that not only work best on small data sets such as CIFAR10, but also work well on large data sets such as ImageNet image classification and COCO object detection.

These two layers were combined to form the NASNet architecture. In NASNet although the general structure is predefined as a series of normal cells (convolutional cells that return a feature map of the same dimension) and reduction cells (convolutional cells that return a feature map where the height and width of the feature map are reduced by a factor of two), blocks are not defined Or cells previously by the authors instead, they are searched by the reinforcement learning search method.

NASNet achieved a prediction accuracy of 82.7% [4] on the validation set, and it performed 1.2% better than all previous models, including Inception V2, V3, Xception, Inception V4, Inception-ResNet V2 and PollyNet. NASNet can be resized to produce a group of models that achieve good accuracy at a low computational cost. Figures 6 and 7 show a schematic of the best performing reduction cell in NASNet with 4,5 blocks identified using CiFAR-10.

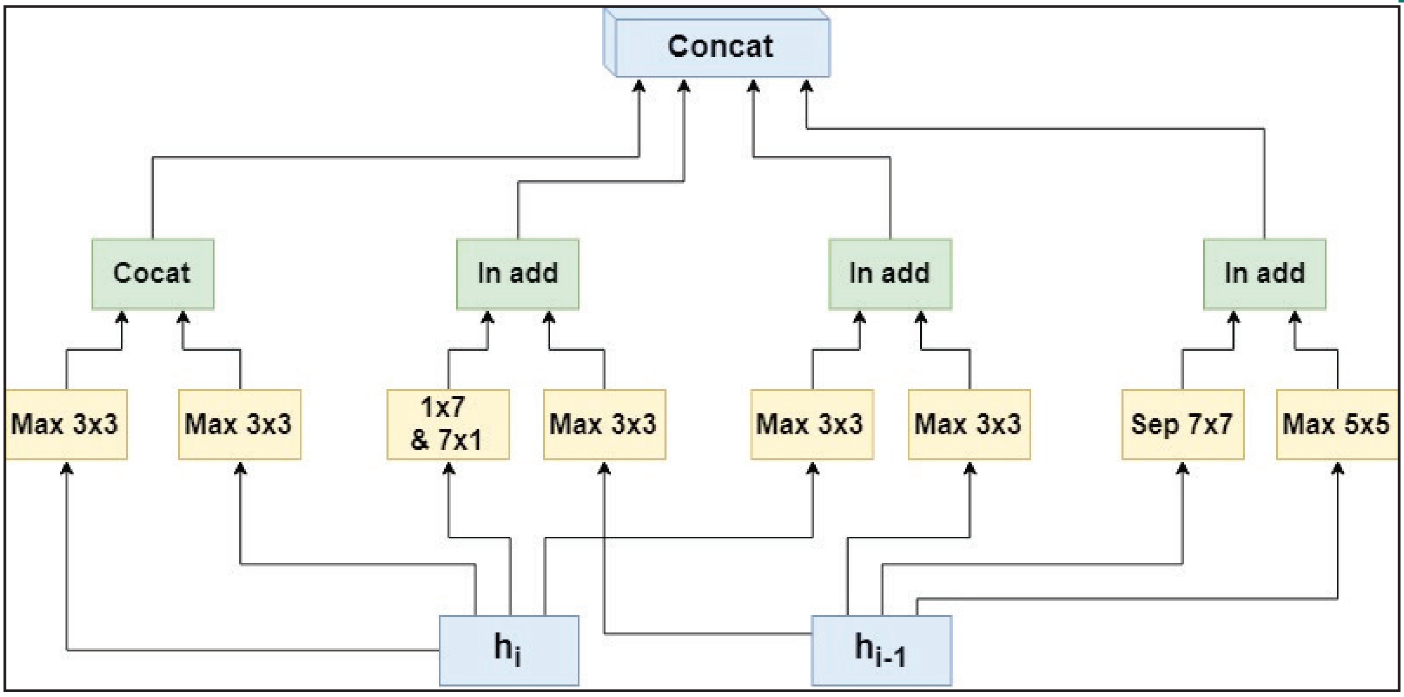
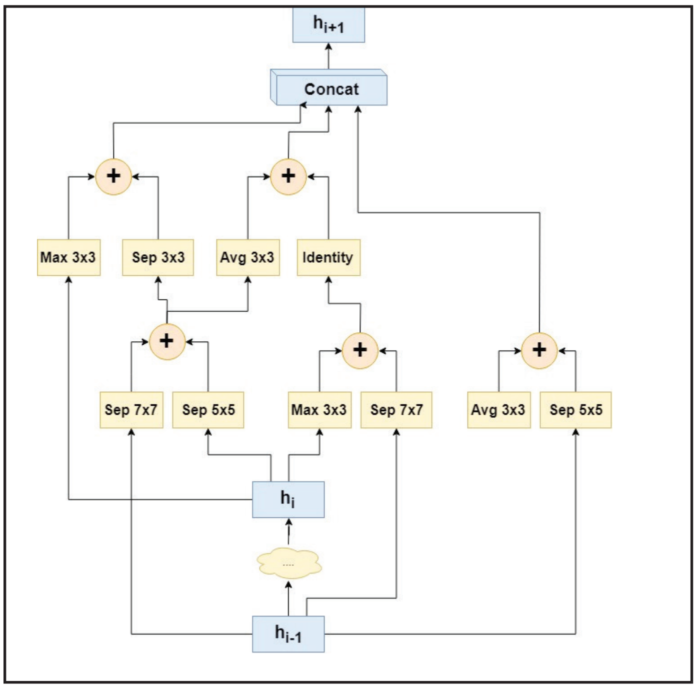


Fig. 6. Reduction cell of the NASNet architecture, 4 blocks, with CIFAR-10 identification



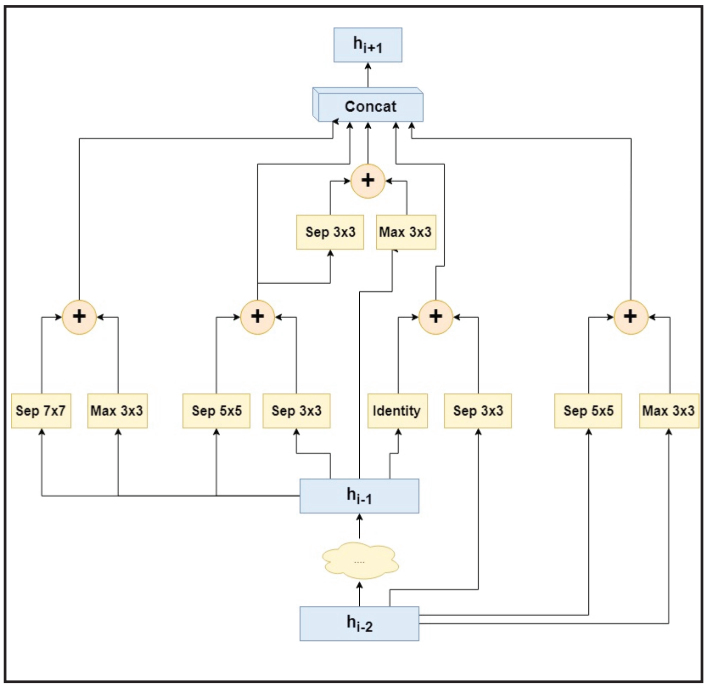


Fig. 7. Reduction cell of the NASNet architecture, 5 blocks, with CIFAR-10 identification

Fig. 8. Cell structure of the progressive neural architecture, with 5 blocks

G) Progressive Neural Architecture Search (PNAS)

PNAS [5] has achieved state-of-the-art classification accuracies on ImageNet and CIFAR-10. PNAS has several advantages over other techniques; for example, it can be trained faster due to its simple structures, allowing the researcher to factorize the search space into a collection of smaller search spaces and to potentially create models with many more blocks. PNAS is 5 times more efficient than is the reinforcement learning (RL) method [20] in terms of the number of models evaluated, and it is 8 times faster in terms of total computation. PNAS is a surrogate-based search method. Surrogate-based optimization is a method that depends on learning a surrogate function that expresses a relationship between sampled models and validation errors. PNAS achieved high performance on the CIFAR-10 data set. It performs a progressive scan of the neural architecture search space. The validation errors are collected by training the selected architectures for several epochs; then, the top performing architectures are chosen at each step of the algorithm. These errors are used to train the surrogate function that predicts the validation error of subsequent architectures. The surrogate function reduces the number of architectures that actually need to be trained, thus allowing an efficient exploration of the search space. To achieve the best results requires 100 GPUS working for 2 days. PNAS is more efficient than either NAS or previous methods, which depend on up to 800 GPUs working for a month. Figure 8 shows the cell structure of PNAS.

III. EXPERIMENTAL RESULTS

In this section, the researcher presents and evaluates 286 videos from two data sets: HMDB [21] and Weizman [22]. The eighty-six videos from the Weizman data set include nine activities (bending, jumping jacks, jumping, jumping in place, running, skipping, walking, waving one hand and waving two hands), and the 200 videos from the HMDB data set include four activities (falling to the floor, punching, kicking and shooting a gun). The researcher divided the data set into two parts: 50% for training and 50% for testing where it is optimized. She evaluated the performance of six CNN network architectures, namely, ResNet, Inception V3, Inception-ResNet, MobileNetV2, NASNet and PNASNet for recognizing human activities.

For the CNNs, the ResNet block is either 2 layers deep (used in small networks such as ResNet 18 and 34) or 3 layers deep (used in larger networks such as ResNet 50, 101 and 152). She used a 3-layer deep architecture in ResNet152 and noticed that ResNet converges faster than its plain counterpart. For the Inception architectures, the researcher used Inception V3 and We also tested Inception-ResNet V2, which achieved slightly better results, and MobileNetV2. The main difference between the MobileNet architecture and a traditional CNN is that instead of a single 3×3 convolutional layer, followed by batch normalization and ReLU, MobileNets splits the convolution into two 3x3 depth-wise and point-wise convolutional layers.
In addition, the researcher also tried large NASNet and PNASNet architectures to choose the best combination of layers and connections that reach higher accuracy on human activity recognition tasks and where it is not guaranteed that a network that works well on one problem can work well on another. Because significant changes may be required to achieve accurate results, the researcher needed to select parameters such as the learning rate and initialization values carefully to avoid overfitting problems. Figure 9 shows the CNN model using a Tensor Flow diagram.

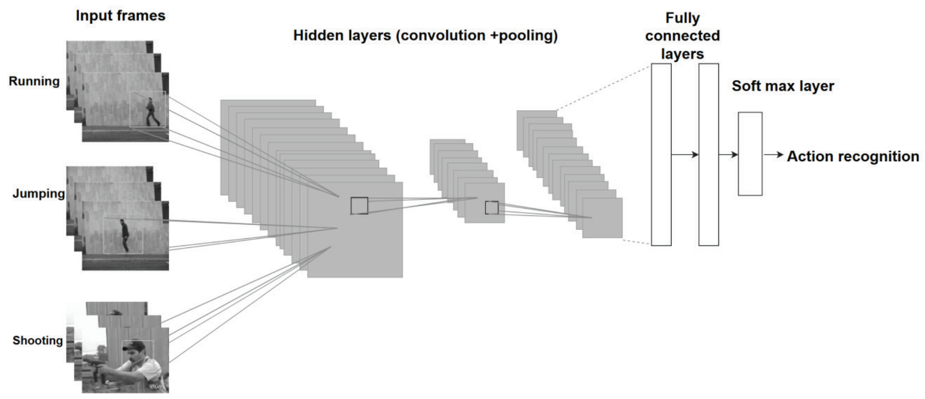


Fig. 9. Overview of the CNN model with Tensor Flow

A) Data sets Description

This section includes a complete description of each data set used. The researcher conducted her experiments on two data sets: HMDB [21] and Weizman [22]. The HMDB data set was collected from various sources, mostly movies, but a small proportion comes from public databases such as the Prelinger Archive, YouTube and Google videos. The HMDB data set contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips. These data set include suspicious behaviours such as falling to the floor, punching, kicking and shooting a gun because the goal of this work was to recognize suspicious behaviors to prevent crimes before they happen. The Weizman data set includes 10 types of activities (walking, running, jumping, galloping sideways, bending, one hand waving, two-hand waving, jumping in place, jumping jacks and skipping). Each action is performed by nine actors. In total, the researchers selected 86 video sequences, each with a spatial resolution of 180 × 144 pixels and a frame rate of 50 frames per second.

B) Hardware and Software Setup

To perform deep learning training on training data with more than 10,000 larger images, GPU training techniques must be applied; when the equivalent training is performed on a CPU with the same data, it would take weeks to complete the training. Therefore, the researcher tried to perform training using a suitable GPU to achieve good, fast training. The NVIDIA GeForce GTX 1060 GPU (6 GB version) includes 1,280 Cuda cores, 6 GB of RAM and a memory bandwidth of 192 GB/sec, allowing training to be performed using 100 images per batch without introducing memory failures. An Intel i5-8400 CPU was used to control the entire process. The researcher used the newest Ubuntu operating system (18.04) with Python because the latter has good support for matrix libraries. She used the newest CUDA toolkit (9.2) to control the GPU-accelerated processing on our GPU. Finally, to implement the deep learning models, she used TensorFlow-GPU 1.8. TensorFlow is the brainchild of Google and intended to assist all deep learning researchers and engineers in implementing deep learning models. Additionally, TensorFlow comes with a toolkit called Tensor Board. The researcher used this tool during training to create real-time charts showing the number of steps, the training error, and the validation error, making it easier to know where overfitting may happen during the training process and to make comparisons between the various DL models. Using TensorFlow code for all the deep learning models, the training and validation data sets, and the configuration file for the training process makes it easier for researchers to make any required modifications in the models during the entire deep learning process. Figure 10 shows the hardware and software setup used for these experiments.

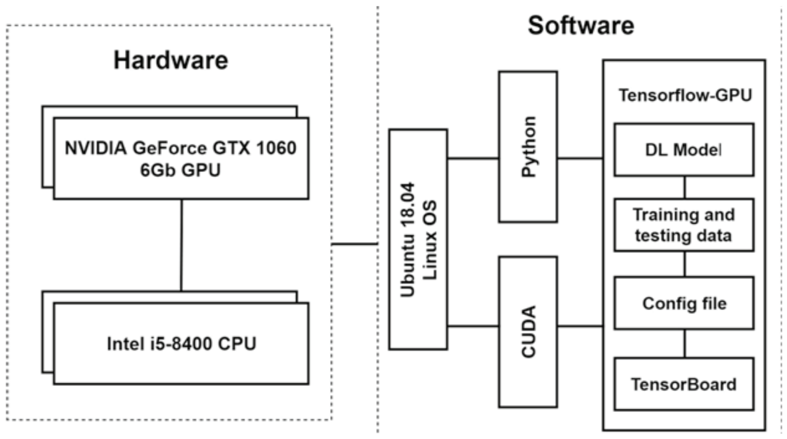


Fig. 10. Hardware and software setup

C) Results

In the first group of experiments, the researcher trained the models using multiple images of different human activities. For the CNN Inception architecture, she trained using 299 x 299 -pixel images and achieved a recognition accuracy of 86.36% on the HMDB data set and 89.9% on the Weizman data set. Additionally, she trained the CNN ResNet architecture using 299 x 299 -pixel images and used 224 x 224-pixel images for the CNN MobileNetV2 architecture, 331 x 331-pixel images for the NASNet architecture and 331 x 331 –pixel images for the PNASNet architecture. The training was performed over 30,000 steps, with 100 images per step. She found that by adjusting the input image scale, she was able to achieve smaller computational budgets and state-of-the-art performances. The best performance of the six CNN architectures for human activity recognition was achieved by the NASNet architecture, which was best on both the HMDB data set and Weizman data set.

The performance of PNASNet ranked second after the NASNet performance on the HMDB data set, but MobileNetV2 ranked second on the Weizman data set. Tables III and VI show recognition rate comparison with different approaches. The difference in performance between Weizman and HMDB data sets is due to the differences between the two data sets. The Weizman data set videos typically include only one person performing one activity in front of a static background, but the HMDB data set contains videos of group activities with variable backgrounds, which cause some occlusions. When objects are occluded by people or things, some missing measurements occur.

Therefore, the researcher preprocessed the images using a Kalman filter to counteract the missing measurements, which led to increases in recognition accuracy. Tables IV and V show the accuracy scores for human activity recognition in percentages by class. The highest accuracy percentages are defined by red rectangles.

TABLE III
COMPARISON OF CNN WITH DIFFERENT ARCHITECTURES WITH STATE- OF-THE-ART METHODS FOR WEIZMAN DATA SET

Method	Recognition accuracy
CNN+ Inception v3	89.9%
CNN+ResNet	89.3%
CNN+Inception ResNet	90.1%
CNN+MobileNet v2	90.4%
CNN+NASNet	91%
CNN+PNASNet	89.3%
Multilevel K-means[23]	98.9%
Fusing appearance and distribution information of interest points [24]	96.66%
Space time shape [25]	99.63%
Histogram of oriented rectangles and encoded with BoVW [26]	93.3%
3D SIFT + SVM classifier [27]	82.6%
Spatial- temporal words [28]	90%
Histograms of oriented 3D spatiotemporal gradients [29]	100%
mid-level motion feature [30]	100%
An Efficient Human Activity Recognition Technique [31]	100%
An information-rich Sampling Technique over Spatio-Temporal CNN [32]	95.78%

TABLE IV
ACTION RECOGNITION ACCURACIES (IN PERCENTAGES) BY CLASS ON THE WEIZMAN DATA SET

Method	Bending	Jumping jacks	Jumping	Jumping in place	Running	Skipping	Walking	Waving one hand	Waving two hands
CNN +Inception V3	98.3	89.7	87	97.8	86	86	94.7	85	85
CNN+ResNet	98.5	91	86	97.7	84	85	93.1	84.6	84.4
CNN+Inception ResNet	89.2	96.5	84	98	87	88	$\boxed{95}$	$\boxed{92}$	82
CNN+MobileNetV2	99	$\boxed{98.2}$	89	97.5	$\boxed{98}$	82.9	90	79	80
CNN+NASNet	99.5	98	$\boxed{92}$	99	90	$\boxed{89}$	89	82	81
CNN+PNASNet	$\boxed{99.9}$	90	83.4	$\boxed{99.5}$	88	81.9	90	85	$\boxed{86}$

TABLE V
ACTION RECOGNITION ACCURACIES (IN PERCENTAGES) BY CLASS ON THE HMDB DATA SET

Method	Falling to the floor	Punching	Kicking	Shooting a Gun
CNN +Inception V3	87.77	88.46	84.2	85
CNN+ResNet	88.77	87.38	89	89
CNN+Inception ResNet	81	92.31	88.33	89
CNN+MobileNetV2	85.8	88.4	88.3	90.1
CNN+NASNet	90	90.7	89.6	88
CNN+PNASNet	89.6	90.6	88.5	87.9

TABLE VI
COMPARISON OF CNN WITH DIFFERENT ARCHITECTURES WITH STATE-OF THE-ART-METHODS FOR HMDB DATA SET

Method	Recognition accuracy
CNN+Inception V3	86.36%
CNN+ResNet	88.5%
CNN+Inception ResNet	87.66%
CNN+MobileNetV2	88.15%
CNN+NASNet	89.57%
CNN+PNASNet	89.15%
STIP with HOG, HOF are encoded with various encoding methods [33]	29.22%
Improved dense trajectory [34]	57.2 %
iDT with HOG, HOF, MBHx, MBHy [35]	61.1%
Improved dense trajectory [36]	66.79 %
Spatial stream ConvNets and optical flow [37]	59.4%
Improved dense trajectory with HOG, HOF, MBHx [38]	65.1%
Trajectory-pooled deep-convolutional descriptor [39]	65.9%
STIP with HOG3D and encoded with various encoding methods [40]	30.5%
Three stream sequential deep trajectory descriptor [41]	65.2%
Human Action Recognition using Multi-Kernel Learning for Temporal Residual Network [42]	68.4%

IV. CONCLUSION

In this paper, the researcher presented a comparison between human activity recognition accuracy using convolutional neural networks of different architectures (ResNet, Inception V3, Inception- ResNet, MobileNetV2, NASNet, and PNASNet). She employed two data sets: the HMDB data set contains examples of group activities, and the Weizman data set contains examples of individual activities. The experimental results indicated that the CNN with the NASNet (Network architecture search) architecture achieves the best performance of the six CNN architectures on both the human activity data sets (HMDB and Weizman) where NasNet is certainly a more advanced technology for searching compact and efficient networks. In addition, she compared the results achieved with few of the state of the art methods.

REFERENCES

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, "Rethinking the inception architecture for computer vision,". in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA 2016, pp. 2818-2826,

[2] K. He, X. Zhang, S. Ren, J. Sun, "Identity Mappings in Deep Residual Networks," in European Conference in Computer Version (ECCV), Amsterdam, 2016, pp. 630-645

[3] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," in the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA, 2017, pp. 4278-4284.

[4] B. Zoph, V. Vasudevan, J. Shlens, V. Le. Quoc "Learning Transferable Architectures for Scalable Image Recognition," in Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City Utah US state, 2018, pp 8697-8710

[5] C. Liu, L. Jia Li, Barret Zoph, Maxim Neumann, et al., "Progressive Neural Architecture Search". Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 19-34.
[6] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City UT, USA 2018, pp4510-4520

[7] T. Plotz, Y. Hammerla, P. Olivier, "Feature learning for activity recognition in ubiquitous computing," in International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, 2011, pp.1729-1734.

[8] M. Zeng, L. T, Nguyen, "Convolutional neural networks for human activity recognition using mobile sensors," in Mobile Computing, Applications and Services (MobiCASE) 6th International Conference, Austin, USA, 2014, pp.197–205.

[9] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, "How transferable are features in deep neural networks?" in Advances in Neural Information Processing Systems 27 (NIPS '14), NIPS Foundation, Montreal, Canada, 2014, pp. 3320-3328.

[10] J. Yang, M. Nguyen, "Deep Convolutional Neural Networks On Multichannel Time Series For Human Activity Recognition," in 24th International Joint Conference on Artificial Intelligence (IJCAI), Argentina ,2015, pp. 3995-4001.

[11] M. Zeng, O. Mengshoel, "Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors," in 6th International Conference on Mobile Computing, Applications and Services (MobiCASE-16), Austin Texas US, 2014, pp197-205.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik,Rich "Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus ,Ohio 2014, pp1-21.

[13] J. Long, E. Shelhamer, T. Darrell, "Fully convolutional networks for semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA2015, pp 3431–3440.

[14] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei- Fei, "Large-scale video classification with convolutional neural networks," in Computer Vision and Pattern Recognition (CVPR) IEEE, Columbus, Ohio, 2014, pp 1725–1732.

[15] A. Toshev, C. Szegedy, "Deeppose, Human pose estimation via deep neural networks" in Computer Vision and Pattern Recognition (CVPR) IEEE, Columbus, Ohio 2014, pp 1653–1660.

[16] N. Wang, D.Yeung, "Learning a deep compact image representation for visual tracking," in Advances in Neural Information Processing Systems Conference, US, 2013, pp 809–817.

[17] C. Dong, C. Loy, K. He, X. Tang, "Learning a deep convolutional network for image super-resolution," in Computer Vision–ECCV Conference, Zurich, Switzerland, 2014, pp 184–199.

[18] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition", ICLR Conference, San Diego, May 2014, pp. 1-14.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, et al., "Going deeper with convolutions," in IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, June 2015, pp 1-9.

[20] B. Zoph, V. Vasudevan, J.Shlen,Le, Q.V, "Learning Transferable architectures for scalable image recognition," in IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1-14.

[21] HMDB Data set@http://serre-lab.clps.brown.edu/resource/ hmdb-a-large-human-motion-database.

[22] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri, "Actions as space-time shapes, IEEE Trans.Pattern Anal. Machine Intell, vol. 29, issue 12, pp. 2247-2253. Dec. 2007.

[23] M. Elshourbagy, E. Hemayed, M. Fayek, "Enhanced bag of words using k-means for human activity recognition,"Egyptian Informatics Journal, vol.17, issue2 ,PP 227-237, July (2016).

[24] M. Bregonzio, T. Xiang, S. Gong, "Fusing appearance and distribution information of interest points for action recognition". Pattern Recognition Journal (Elsevier), vol

45, issue 3, March 2012, pp 1220–1234.

[25] M. Blank, L. Gorelick, E. Shechtman, M. Irani, R. Basri, "Actions as space-time shapes," in Tenth IEEE International Conference on Computer Vision (ICCV'05), Beijing, China, 2005, pp. 1395–1402.

[26] N. Ikizler, P. Duygulu, "Human action recognition using distribution of oriented rectangular patches," in Human Motion-Understanding Modelling Capture and Animation Conference, Rio de Janeiro, Brazil, 2007, pp. 271- 284.

[27] P. Scovanner, A. Ali, M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in Proceedings of the 15th ACM international conference on Multimedia, Augsburg, Germany, 2007, pp.3 57–360.

[28] J.Carlos, H.Wang, L.Fei-Fei, "Unsupervised learning of human action categories using spatial- temporal words," Int J comput vision, vol. 79, pp.299-318, March 2008

[29] A. Klaser, M. Marszalek, C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in Proceedings of the British Machine Vision Conference Leeds, United Kingdom, 2008, pp. 271–275.

[30] A.Fathi, G.Mori, "Action recognition by learning mid-level motion features," in IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, 2008, pp. 1–8.

[31] A. Khelalef, Fakhreddine, N. Benoudjit, "An Efficient Human Activity Recognition Technique," HAL open science Journal, vol 29, issue 4, pp 702-715 , June 2020.

[32] SH.Shabbeer Basha, "An information-rich sampling Technique over spatio-temporal CNN for classification of human actions in video," Computer vision and pattern recognition Journal, arxiv:2002.02100v2, 7Feb (2020).

[33] X.Wang, L.Wang, Y.Qiao, "A comparative study of encoding, pooling and normalization methods for action recognition," in Computer Vision, Asian Conference on Computer Vision, Sydney, Australia., 2012, pp. 572–585.

[34] H.Wang, C.Schmid, "Action recognition with improved trajectories," in IEEE International Conference on Computer Vision, Sydney, Australia, 2013, pp. 3551–3558.

[35] X.Peng, L.Wang, X.Wang, Y.Qiao, " Bag of Visual Words and Fusion Methods for Action Recognition, Comprehensive Study and Good Practice: Elsevier Vol. 150 pp. 109–125, September 2016.

[36] X.Peng, C.Zou, Y.Qiao, Q.Peng, "Action recognition with stacked fisher vectors," in Computer Vision, Asian Conference on Computer Vision, ECCV, Zurich, 2014, pp. 581-595.

[37] K.Simonyan, A.Zisserman, "Two-stream convolutional networks for action recognition in videos," in Advances in Neural Information Processing Systems Conference, Montreal, Quebec, Canada. 2014, pp. 568–576.

[38] Z.Lan, M.Lin, X.Li, A.G.Hauptmann, B.Raj, "Beyond gaussian pyramid: multi-skip feature stacking for action recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA., 2015, pp. 204–212.

[39] L.Wang, Y.Qiao, X.Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 4305–4314.

[40] X.Zhen, L.Shao, "Action recognition via spatio-temporal local features: a comprehensive study," Image and Vision Computing, vol. 50, pp. 1–13, June 2016.

[41] Y.Shi, Y.Tian, Y.Wang, T.Huang, "Sequential deep trajectory descriptor for action recognition with three-stream CNN," IEEE Transactions on Multimedia, vol 19 issue7, pp. 1510–1520, July 2017.

[42] S.Nazir, M.Haroon, S.Velastin, "Human Action Recognition using Multi- Kernel Learning for Temporal Residual Network," in 14th International Joint Conference Vision, Imaging and Computer Graphics Theory and Applications, Prague Czech Republic, 2019, pp.420-426