IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 29 September 2023, accepted 21 November 2023, date of publication 5 December 2023,
date of current version 12 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3339575

RESEARCH ARTICLE

Transfer Learning Models for CNN Fusion With Fisher Vector for Codebook Optimization of Foreground Features

MOHAMED GAMAL M. KAMALELDIN[1,2], SYED A. R. ABU-BAKAR[2], (Senior Member, IEEE), AND USMAN ULLAH SHEIKH[3]

[1]Electronics and Communications Engineering Department, Arab Academy for Science, Technology and Maritime Transport, Cairo 11799, Egypt
[2]Computer Vision, Video and Image Processing Research Laboratory, School of Electrical Engineering, Universiti Teknologi Malaysia, Skudai, Johor 81300, Malaysia
[3]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Johor Bahru 81310, Malaysia

Corresponding authors: Mohamed Gamal M. Kamaleldin (mohamed.gamal.1@gmail.com) and Usman Ullah Sheikh (usman@fke.utm.my)

This work was supported by the Ministry of Education Malaysia and Universiti Teknologi Malaysia (UTM) under the UTM Fundamental Research Grant under Grant Q.J130000.3823.22H29.

ABSTRACT Human action recognition has become one of the main topics in the computer vision field due to its high demand and competitiveness in real-world applications. The main goals of human action recognition are to improve classification accuracy and reduce computational complexity. Previous studies have mainly used two approaches: the hand-crafted feature extraction approach and the deep learning approach. The hand-crafted approach is simple, which confers it with an added advantage in terms of computational complexity. However, this method is low in accuracy. Conversely, the deep learning approach achieves high accuracy even for complex datasets, but it suffers in terms of computational complexity and long training time as it needs to process huge datasets during training. Other approaches include the use of pre-trained deep learning networks to fuse both methods. In this paper, we will introduce a combination of pre-trained convolutional neural networks (CNN) to extract features, an improved Fisher vector (iFV) codebook, and an optimized support vector machine SVM to achieve improved human action recognition. We leveraged three pre-trained CNNs, namely, Inception-ResNet-v2, NASNet-Large, and Xception, to extract the features. Then, we applied the improved Fisher vector codebook to encode them. We subsequently trained the codebook using SVM for classification and re-adjusted the SVM weights using five different optimization techniques, which are SGD, Adadelta, ADAM, Adamax, and Nadam. To evaluate the performance, we utilized UCF101 and HMDB51 datasets. The results demonstrate that the accuracy and computational complexity of our approach are comparable to state-of-the-art techniques.

INDEX TERMS Human action recognition, pre-trained convolutional neural networks, long short-term memory (LSTM), features encoding, optimization.

I. INTRODUCTION

Over the past few decades, many researchers have directed their research interests toward human action recognition (HAR) due to its wide spectrum of applications and associated challenges. Some of these applications include video surveillance in public and indoor places, home care for elderly people, smart homes, virtual reality, video entertainment systems, video indexing, anomalous traffic detection, gesture control and video monitoring for suspicious people. In most available datasets, the main challenges of HAR are different viewpoints, different movement speeds, scale factors, non-uniform illumination, cluttered backgrounds, and occlusions. Two different approaches have been applied in tackling HAR: the hand-crafted approach and the deep learning approach.

The associate editor coordinating the review of this manuscript and approving it for publication was Seifedine Kadry.

The hand-crafted approaches are mainly based on manually extracting the spatiotemporal features. They can be categorized into interest points-based methods such as the use of scale-invariant *feature* transform (SIFT) [1], speeded-up robust features (SURF) [2], and the dense features-based methods that include Hessian matrix, Gabor filter, space-time interest points (STIP), motion scale-invariant feature transform (MoSIFT), trajectory (TRA), histogram of oriented gradients (HOG), histograms of optical flow (HOF), and motion boundary histogram (MBH). The approach that combines HOG, HOF, and MBH features to form improved dense trajectory (iDT) features has been proven to achieve the best results [3]. These features are then encoded using different codebooks such as the bag of visual words (BoVW) [4], vector of the local aggregated descriptor (VLAD) [5], locality-constrained linear coding (LLC) codebook [6], and Fisher vector codebook [7] methods. The final classification process is done using different machine learning techniques such as the support vector machine (SVM), which is the most commonly used technique. While the hand-crafted approach has low computational complexity and a training process that consumes less time, it suffers from low accuracy.

The deep learning approach utilizes the convolution neural network (CNN) frameworks and has thus far been the state-of-the-art approach in recent years due to its superior performance. In its simple form, the CNN consists of an input layer, a large number of hidden layers, and an output layer. When the video frames are fed into the network, it extracts the spatiotemporal features automatically and subsequently classifies the actions [8]. For better performance, 2D Convolutional Neural Networks (2D CNNs) have been widely used for spatial feature extraction and classification. However, this method has poor motion detection which makes it more suitable only for image classification. A recurrent layer is added to the last layer of the 2D CNN to extract the temporal information for video classification [9]. Nevertheless, this approach fails to identify the inferior motion. In addition, more training is required because the network has to keep track of all video sequences, which translates to an increase in computational complexity [10]. Another approach is the use of 3D CNNs [11] that combine two-stream networks; one stream is to extract the space features while the second stream is to track features through successive frames using optical flow. This approach requires the network to be trained with a massive number of data compared to the 2D CNN-based methods. Hence, it needs intense computing power and precise network architecture. It is also very time-consuming and is not applicable to small datasets. Furthermore, transfer learning has been used to improve the implementation of CNN [12]. Transfer learning massively speeds up the training process as the target domain model does not require training from scratch with a huge amount of data. Subsequently, this has significantly reduced the training data size and training time, which in return improves accessibility, the learning process, and network generalization.

Transfer learning network can be directly applied to a given dataset as fixed weights pre-trained CNN. Yet, another scenario is to fine-tune the networks' weights by retraining the weights of some selected layers on the new dataset while freezing the remaining layers weights, or use the network as a feature extractor by replacing the last layer with a conventional codebook and a classifier. While the last layer which is a fully connected (FC) layer followed by a non-linear activation function that is the SoftMax layer (and loss function of cross entropy in case of multi class problems), can be deployed straightaway for classification, those of convolutional layers are nonlinear and considered typically too high dimensional to be employed directly [13]. Hence, it requires the adoption of feature extraction or dimensionality reduction techniques on those CNN activations before the classification process.

In order to reduce the computational complexity, the final fully connected layer and multinomial logistic regression used by SoftMax layer and classifier are replaced by a feature encoder followed by a linear classifier [14], Encoding features helps to cluster, group, handle, and recognize these features. Additionally, it guarantees the coherency of the classified features which accordingly helps in improving the accuracy.

In the past, Bag of visual Words (BOvW) has been used as the most widespread encoding approach. However, this method lacks optimal feature selection, resulting in the inclusion of numerous background features leading to an increase in the computational complexity. In addition, several local features are encoded via the same visual word, resulting in inaccurate k-means clustering issues. Moreover, because BOvW describes a video using a random set of local descriptors, the spatial data arrangement is missing. Finally, whereas some visual words may be more significant than others, BOvW assigns all visual words the same weight thereby reducing the technique's discriminative ability [7], [15]. As an alternative, Vector of Locally Aggregated Descriptors (VLAD) and Fisher vector encoding methods were presented as an alternative compact codebook representation. Both Fisher vector and VLAD encoding methods capture variations in the distribution of local descriptors supplied by a cluster center. VLAD models its codebook using k-means, with a codebook size of $K \times D$, where K is the number of cluster centers and D is the feature dimension. This technique relies only on 1st order statistics (means) and each feature descriptor is associated with the nearest visual word in the codebook [16]. However, it has never achieved the Fisher vector accuracy. In addition, the sum of residue vectors between local descriptors and cluster centers could not provide sufficient diversity, thereby reducing the ability to differentiate between different classes [17]. On the other hand, the Fisher vector technique uses Gaussian mixture model to represent its codebook with a size of $2 \times K \times D$ where the 1st order statistics (means) and 2nd order statistics (covariance) are used to represent each Gaussian element. Fisher vector technique has advantages

over other methods due to its efficiency in terms of computing the codebook visual representations, classifier training and testing processes especially for large scale datasets, this leads to excellent results even with efficient linear classifiers. In the work in [7], demonstrated that applying L2 normalization for each cluster or each Gaussian component can massively enhance the performance. Another advantage of Fisher vector over VLAD is that Fisher coding is viewed as a Gaussian kernel codebook that integrates probability and credibility, while VLAD is represented by a conventional codebook that has a fixed weight and hard-voting. Consequently, VLAD is regarded as a non-probabilistic form of GMM [18].

In this paper, we introduce a new method that leverages the fusion of pre-trained CNN extracted features and a hand-crafted codebook. First, we selected three pre-trained CNNs to extract features. Then, PCA was used as a feature selection method to select the dominant features and remove the redundant features. Next, we engaged in an improved Fisher vector codebook to encode these features. Finally, SVM was applied to classify the actions. The codebook gradients representing the foreground features were selected using five different optimization techniques under the inspection of the validation labels to reduce the codebook size. By removing the redundant codebook elements, we improved accuracy and reduced the training time.

The main contributions of this paper are summarized as follows:

- Introducing a new method that proves to enhance the model accuracy compared to the pre-trained models CNN and other state-of-the-art techniques.
- Developing an optimization addition to the codebook that removes the redundant elements which reduces the computational complexity.

The rest of the paper is organized as follows: Section II presents the related works. Section III introduces the three pre-trained CNNs used for feature extraction, the codebook generation using Fisher vector, classification using SVM, and the mathematical model for applying the optimization process to extract foreground components and retraining the optimized model using SVM. Section IV introduces the experimental results applied to two datasets and compares their performance to the state-of-the-art techniques. Finally, Section V concludes the paper along with recommendations for future work.

II. RELATED WORKS

A. HAND-CRAFTED METHODS

Previously, hand-crafted feature approaches have been a popular technique for action recognition [19]. For instance, 2D spatial descriptors have been extracted for image recognition while 3D spatiotemporal descriptors have been widely used for video classification. Wang and Schmid [3] were the first to propose the iDT descriptors leveraged on the combination of histograms of oriented gradients (HOG), histograms of optical flow (HOF), and motion boundary histograms (MBH) to extract spatiotemporal features. These local features were then encoded using Fisher vectors and classified using an SVM classifier. When tested on UCF101 and HMDB51 datasets, this method was reported to have a performance of 85.9% and 57.2%, respectively. Li et al. [20] developed a model of unified appearance and motion variation (UMAMV) that detects SURF feature points with unique appearance information in the spatial domain and reflects motion change in the temporal domain. They tested their method on Weizmann and UCF101 datasets. Aslan et al. [21] compared four different machine learning methods to classify both greyscale and binary images on KTH and Weizmann datasets. They extracted features using a SURF detector, encoded them using a bag of visual words (BOVW), and compared the results of four different classifiers viz. k-nearest neighbor, SVM, decision trees, and naïve Bayes. Bayesian optimization was used to improve the accuracy. Nazir et al. [22] proposed a new codebook design to represent the spatiotemporal visual words called Bag of Expression (BoE). After coupling each visual word with several neighbors in the spatiotemporal domain to obtain independent visual word pairs, they found that this technique improved the handling of view independence together with scale invariance and occlusion in real-world scenarios. Duta et al. [23] proposed a new descriptor using histograms of motion detection that captures motion information using a simple temporal derivation. They claimed that their proposed method could reduce computational costs compared to the optical flow approach. In addition, they developed the Shape Difference VLAD (SD-VLAD) encoding technique that attached the shape information. Based on UCF50, UCF101, and HMDB51 datasets, they found that their method yielded better accuracy and less computational costs. Peng et al. [24] proposed a new codebook representation from iDT features called a hybrid super vector that combines the outputs from a manifold of BoVW models. Tested on UCF50, UCF101, and HMDB51 datasets, the aggregation of these codebook generations improved the descriptive power for human action recognition. Yamada et al. [25] proposed a new approach for spatial segmentation based on hand-crafted trajectory features. Using iDT features, they segmented the spatial frames, identified the critical features (including human motion), computed the optical flows, and tracked them over the frames. Fisher vector was used for codebook generation and classification was carried out using RNN instead of SVM for better results. This method was applied to the J-HMDB and MPII Cooking Activities datasets.

B. FUSION METHODS

Simonyan and Zisserman [26] proposed a two-stream CNN for action recognition architecture that combines spatial networks for feature detection and temporal networks for optical flow. The results were compared to the handcrafted state-of-the-art methods using UCF101 and HMDB51 datasets, and the accuracies obtained were 88.0% and 59.4%, respectively. Bilen et al. [27] proposed a method that maps RGB images

to a single dynamic image and applies it to a CNN. Subsequently, Wang and Qiao [28] proposed a method that combines a two-stream CNN and the iDT method to extract features and forms a descriptor called trajectory-pooled deep-convolutional descriptor (TDD). Fisher vector is then used to represent each video into a global super vector which is then classified using SVM. In another development, Yang et al. [29] proposed a fully connected recurrent neural network (FC-RNN) to temporally track long videos. Since the features have different discriminative abilities depending on the action, they proposed a multilayer and multimodal fusion framework of CNN features. The method uses four integral structures involving 2D-CNN-SP (single spatial frame), an optical flow image along with 3D-CNN on a short clip of spatial frames, and optical flow images. VGG16 and C3D pre-trained networks with recurrent layers are used to extract the features, and these features are then mapped using iFV. Finally, a linear SVM solver is used for the classification boosted by the boost-u algorithm that jointly fuses CNN with multiple approaches. The technique was evaluated using UCF101 and HMDB51 datasets, and it achieved accuracies of 91.6% and 61.8%, respectively. Huang et al. [30] proposed a 2D inflated operation that converts 2D pre-trained CNN kernels to 3D CNN filters by reorganizing the 2D kernels in a parallel manner through temporal frames. As a result, they obtained 3D kernels that minimize the training cost. Additionally, they included the iDT features in the networks to improve the results and applied the accumulated gradient descent technique in the training process to boost its accuracy. In another attempt, Wang and Li [31] modified the ResNet to extract a multi-feature map by adding numerous up-sampling layers to the pre-trained network. The resulting expanded feature maps lead to an increment of feature numbers and thus improvements in the training process. In their seminal work, the trilinear interpolation method was used for up-sampling the feature. Finally, they used the weighted geometric means combination forecasting method based on the L1 norm to combine features of all up-sampled layers. Song et al. [32] proposed a simple yet effective technique called Temporal-Spatial Mapping (TSM). Utilizing pre-trained VGG16 and TSM with BN-Inception, they extracted convolutional features for each frame of a video sequence. Next, they generated a video map and a 2D feature map by encoding the temporal-spatial information using TSM, which was later used to build a temporal attention module for predicting the final action categories. Zebhi et al. [33] proposed an alternative approach to represent video sequence spatiotemporal information. They leveraged the use of gait history images (GHI), which are similar to motion history images (MHI). For temporal representation, they used a time-sliced average gradient boundary magnitude (TAGBM) descriptor. Each video is split into N and M groups of consecutive frames, and the GHI and TAGBM are computed for each group, resulting in spatial and temporal templates, subsequently used as features. A pre-trained VGG16 framework together with transfer learning was selected for classification. Aly et al. [34] compared the performance of two pre-trained CNN models with different network architectures, i.e., GoogleNet and AlexNet on human action recognition datasets. Feature vectors are extracted from a video and trained via the transfer learning (TL) paradigm using Long-Short Term Memory (LSTM) framework to predict the video action labels. Next, Abdulazeem et al. [12] proposed a TL-based approach for human action recognition that utilizes a two-stream temporal CNN architecture along with LSTM. Tu et al. [35] proposed a multi-stream CNN composed of three networks, with each network as a two-stream network (TS-Nets). The first network extracts features of a bounding box around the detected human using an improved Block-sparse Robust Principal Component Analysis (IB- RPCA) method. The second network extracts features of human moving body parts captured using motion saliency measure. The third network extracts features from the whole RGB image using a combination of VGG 16 pre-trained network and iDT features. Based on these three networks, three motion streams are organized from the optical flow field. These features are integrated efficiently using a spatiotemporal 3D convolutional fusion approach. The CNN is tested using UCF101 and HMDB51 datasets with accuracies of 94.5% and 69.8%, respectively. Zamri et al. [36] also proposed a vision-based human action recognition via transfer learning. Their approach uses AlexNet as a pre-trained CNN to extract low-level features from three different image maps, i.e., motion history image that sustains spatiotemporal data, binary motion energy image that captures the motion region data, and optical flow information that holds accumulative motion speed data. Jerusha and Kumar [37] proposed a system that uses both RNN and LSTM units to classify activities in videos. Their proposed method contains pre-trained convolutional neural network (CNN) models recognizing actions using the transfer learning method. Various pre-trained CNN models including VGG16, InceptionV3, Resnet50, Resnet150, and Resnet152 were used to extract the visual features, and the videos of the UCF101 dataset were classified based on those features. The accuracy is comparable to the state-of-the-art methods. Muhammad et al. [38] proposed a bi-directional long short-term memory (BiLSTM) supported by an attention mechanism to detect and identify the most important features in sequential multi-frame data. To improve the classification process, they added a re-center loss function to the SoftMax layer. Dong et al. [39] proposed a generative model that evaluates human actions using still images. The model is based on two VGG16 pre-trained CNNs in which the first one is used for feature extraction, and the second one is used to keep the important features for each frame instead of tracking them through successive video frames using optical flow. Khan et al. [40] proposed a layout that comprises numerous steps including feature mapping, feature fusion and feature selection. The extracted deep features are combined by utilizing the Sequential-based Extended (SbE)

method. They employed the Kurtosis-controlled Weighted KNN method to select the best features. The chosen features are classified using several learning algorithms. Zhou and Zhang [41] proposed a method that fuses feature pyramid networks with a multi-scale feature fusion technology and attention mechanisms to improve the performance of human detection in crowded scenarios. Feature pyramid network (FPN) with an improved hierarchical split block is formulated. Next, the lateral connection in the FPN is replaced with an attention-based lateral connection (ALC) module with spatial and channel attention mechanisms. This allows detectors to focus on key aspects of occlusion patterns while also improving the representational ability of feature maps using massive spatial and semantic information. A bottom-up path augmentation (BPA) module was also used to take advantage of the Scale-FPN and ALC modules' characteristics. Scale-FPN, ALC, and BPA are combined to create SA-FPN and integrate it into the formulation of a crowded human detector to test the efficacy of the suggested method. Feichtenhofer et al. [42] proposed a multiplicative gating of the appearance stream for a two stream residual network (ResNet) pre-trained CNN that adds cross stream connections in the network early stages to ensure the connection between the spatial stream and the temporal stream during the training process. At the feature level, temporal filters are inserted between network layers that are created as identity mapping kernels. These temporal filters add new layers to the already established ResNet model while maintaining the identity of residual networks' features. This method is used to overcome the dominance of the spatial stream over the motion stream during the training process. This process results in an accuracy improvement. iDT features are added to the ResNet features to boost the performance. This performance is tested using ResNet-50 and ResNet-152 on UCF101 and HMDB51 datasets shows accuracies of 94.9% and 72.2% respectively. Hao and Zhang [43] proposed a two streams fusion method for video action recognition based on Dense Convolutional Networks (DenseNet). The multiplicative gate mechanism is used to connect between appearance and motion sequences at the building block level. In addition, knowledge distillation network is inserted to connect between the two streams and their final fusion. This network enables both streams to interact at the highest level layers. Its unique architecture enables successful connections between appearance and motion streams at various level layers, thereby promoting the formation of complex spatiotemporal features. Additionally, it allows the network to be trained as an end-to-end training, that guarantees the fusion process all over the entire network. The efficacy of the technique is evaluated using the UCF101 and HMDB51 datasets, with respective accuracies of 93.78% and 66.88% respectively.

III. METHODOLOGY

In transfer learning, the source and target domains depend on different datasets. A huge dataset is always used for training the source domain (e.g. ImageNet), while the input dataset for the target domain can be small.

Transfer learning can be used in three different ways: fixed feature extraction, freezing of layers and fine-tuning of weights, and the pre-trained models. In fixed feature extraction, the data is used as an input and the weights of the first fully connected layer are kept while the final fully-connected layer (also known as the SoftMax layer) and the classification stage are removed from the CNN network. In the fine-tuning and layers freezing, the pre-trained model structure is retained while the network weights are fine-tuned to fit in the target dataset. The process of weight fine-tuning can be conducted for all layers of the CNN network or only its higher layers.

The majority of popular architectures for pre-trained CNNs depend on huge datasets like the ImageNet dataset [44] to adjust the training weights. This study used the Xception [45], Inception-ResNet-v2 [46], and NASNet-Large networks [47]. These network structures yield excellent accuracy when supported by LSTM, which provides a temporal track for video frames beside the spatial stream for each image and naturally helps to massively improve the classification accuracy. Additionally, these networks are pre-trained using the ImageNet dataset which is an extensive large-scale dataset involving more than 15 million labelled high-resolution images across around 22,000 categories. The main reason for choosing these networks over other pre-trained CNNs is that they achieved the best performance on the ImageNet dataset [47], [48].

Figure 1 outlines our proposed framework. The data was first divided into three sets: training, validation, and testing. Next, we used the training dataset to extract the features using three deep pre-trained CNN network architectures. For each pre-trained network, the SoftMax and classification layers were removed to extract the network features after the last convolutional layer. These features were applied to the improved Fisher vector to form an elementary codebook composed of means and covariances values for each Gaussian component for features encoding. The codebook size was set up as $2 \times$ Gaussian elements number $\times$ features size since such encoding eases and improves the classification process. Elementary codebooks with different Gaussian element sizes and class labels were applied to train the support vector machine linear classifier in a *one versus all* strategy. Then, the normal vector of the hyperplane $\mathfrak{G}_{\phi}^{m}$ that separates class m from other SVM classes was extracted, and the codebook elements were rearranged so that the means and covariances elements of each Gaussian component $\mathcal{g}_{\phi}^{X}$ are multiplied by the variable B as shown in (4):

The Gaussian components matrix is written as:

$$\mathcal{g}_{\phi}^{X} = (\mu_1, \mu_2, \cdots, \mu_N, \beta_1, \beta_2, \cdots, \beta_N) \tag{1}$$

where μ represents the means, β represents the covariances, and N is the total number of Gaussian elements, the matrix is rearranged as:

$$\begin{aligned} \mathcal{g}_{\phi}^{X} &= \left(\mu_1, \beta_1, \mu_2, \beta_2, \cdots, \mu_n, \beta_n, \cdots, \mu_N, \beta_N\right) \\ &= \left(\mathcal{g}_1^X, \mathcal{g}_2^X, \cdots, \mathcal{g}_n^X, \cdots, \mathcal{g}_N^X\right) \end{aligned} \tag{2}$$

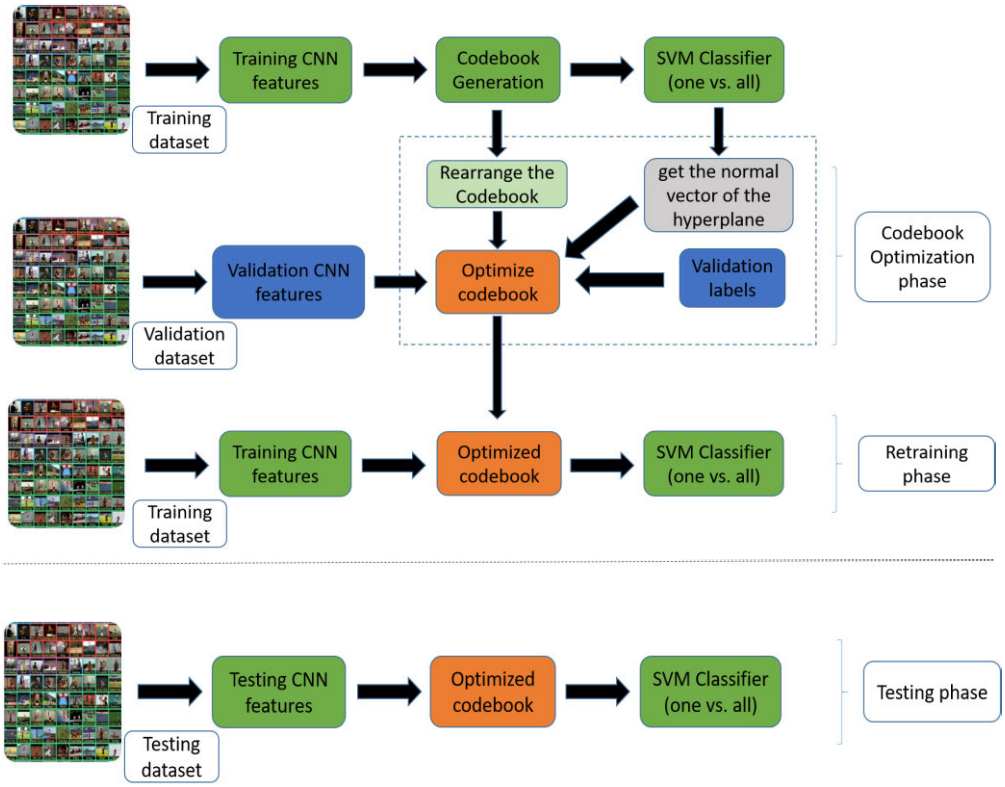


FIGURE 1. General model block diagram.

To solve the optimization problem, the Gaussian components matrix is multiplied by the optimization variable B as shown:

$$\widehat{g}_{\phi}^{X} = g_{\phi}^{X} B = (\left(g_{1}^{X} b_1, g_{2}^{X} b_2, \cdots, g_{n}^{X} b_n, \cdots, g_{N}^{X} b_N\right)) \quad (3)$$

The predicted output is written as:

$$\widehat{y}_X = \mathrm{L}\left(\frac{g_{\phi}^{X} B}{\left\| g_{\phi}^{X} B \right\|_2}, \mathfrak{G}_{\phi}^{m}\right) \quad (4)$$

where L is the linear kernel, g_{ϕ}^{X} is the Gaussian element, X represents the local descriptors, ϕ is the accumulation of all of the parametric distributions, and B is the optimization variable.

Validation data, validation labels y_k and $\mathfrak{G}_{\phi}^{m}$ were used to optimize the values of the Gaussian elements to fit the SVM hyperplanes, where $\mathfrak{G}_{\phi}^{m}$ is the classification hyperplane normal vector differentiating the selected class and other classes, y_X is the class label:

$$\widehat{y}_X = L\left(\frac{g_{\phi}^{X} B}{\left\| g_{\phi}^{X} B \right\|_2}, \mathfrak{G}_{\phi}^{m}\right) = \frac{\sum_{n=1}^{N} g_{n}^{X^T} b_n \mathfrak{G}_{n}^{m}}{\sqrt{\sum_{n=1}^{N} g_{n}^{X^T} g_{n}^{X} b_n^2}} \quad (5)$$

The optimized output, O_{out} is written as:

$$O_{out} = arg \max_{B} \sum_{X} L(y_X, \widehat{y}_X) \quad (6)$$

The optimization problem is solved depending on the following equation. If the predicted output and the validation class labels are equal, then the variable b is set to one and the Gaussian component is included in the codebook. If the predicted output and the validation class labels are different, b is optimized to zero so that the Gaussian component is eliminated from the codebook.

$$L\left(y_X, \widehat{y}_X\right) = \begin{cases} 1, & y_X = \widehat{y}_X \\ 0, & y_X \neq \widehat{y}_X \end{cases} \quad (7)$$

$$O_{out} = arg \max_{B} \sum_{X} L\left(y_X, \frac{\sum_{n=1}^{N} g_{n}^{X^T} b_n \mathfrak{G}_{n}^{m}}{\sqrt{\sum_{n=1}^{N} g_{n}^{X^T} g_{n}^{X} b_n^2}}\right) \quad (8)$$

Regarding the optimization process, stochastic gradient descent (SGD) has been widely used as the most efficient optimization method for reducing the cost function of large datasets [49]. In addition, recently improved SGD types have been used for CNN training. Here, we applied SGD [50], Adadelta optimizer [51], Adaptive moment estimation (Adam) optimizer, Adamax optimizer [52], and Nesterov accelerated Adam (Nadam) optimizer [53] methods. Their results are compared in the next section.

Following the optimization procedure, the new values of B were applied to the codebook to adjust the values of its elements. Finally, the training features were applied to the

optimized codebook to retrain the classifier. Then, the testing data were used to get improved accuracy.

IV. EXPERIMENTAL RESULTS

In this section, we discuss the performance of the proposed model applied to two established datasets and compare the experimental results with state-of-the-art techniques.

A. EXPERIMENTAL SETUP

We conducted the experiments on an AMD Ryzen Threadripper 3960X 24-Core Processor 3.79 GHz, 64 GB RAM, and Nvidia GTX 3080 with 12 GB GPU graphics card computer running on Windows 10 operating system. We used MATLAB programming language for the software.

The first dataset used was the UCF-101 dataset [54] with 13320 videos composed of 101 actions separated into five categories: human-object interaction, body-motion, human-human interaction, playing musical instruments, and sports. The video resolution was 360 × 240 and the frame rate was 25 frames/second.

The second dataset used was the HMDB51 dataset [55] with 6766 videos composed of 51 actions collected from sources such as Youtube and Google videos. The actions were categorized into five categories: facial actions, facial actions with object manipulation, general body movements, body movements with object interaction, and body movements for human interaction. The video resolution was 320 × 240 and the frame rate was 30 frames/second.

These datasets were used for testing because of their huge diversity of actions. Moreover, these actions covered a variety of different poses, object scaling, camera motion, viewpoints, cluttered backgrounds, and illumination conditions.

The datasets were divided into 70% for training, 15% for validation, and 15% for testing. Three different pre-trained CNNs were used for feature extraction during the training data. The encoding of these features was accomplished by the Fisher vector codebook using the VLfeat library [56]. The encoded features were then sent to the SVM for classification. Next, the validated pre-trained CNN features were applied to the codebook and the normal vector of the hyperplanes separating the SVM classes to optimize its elements under the validation labels inspection. Finally, the training features were applied to the optimized codebook to improve accuracy.

B. RESULTS AND DISCUSSION

We tested the three network architectures to extract the features as mentioned above and used four different codebook sizes for feature encoding, i.e., 64, 128, 256, and 512. For the optimization techniques, we implemented five different methods: SGD, Adadelta, Adam, Adamax, and Nadam. For classification, we leveraged the SVM technique.

C. EFFECT OF USING DIFFERENT OPTIMIZERS

Figures 2, 3, and 4 show the performance comparison of applying the five different optimizers on the UCF 101 dataset, while Figures 5, 6, and 7 show the same performance comparison for the HMDB51 dataset. As illustrated in Equation 5, the classifier optimization equation sets the value of B = 1 if the codebook descriptors are relevant and sets B = 0 if they are irrelevant. By doing this, redundant components are taken out from the codebook; hence, its size is reduced and it is made more efficient. The results demonstrate that the Adamax and Nadam optimizers produced the best results. Tables 1 and 2 show the accuracy vs. codebook size, where the codebook size "k" increased in a sequence of 64, 128, 256, and 512. The accuracy improved as the codebook size increased. However, the processing time increased as the codebook size increased (see Tables 3 and 4); this point will be illustrated in the next subsection.

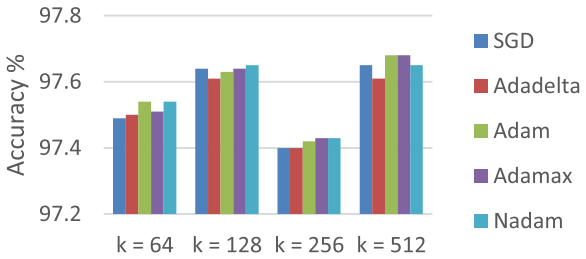


FIGURE 2. Accuracy comparison for five different optimizers for the Xception network on the UCF101 dataset.

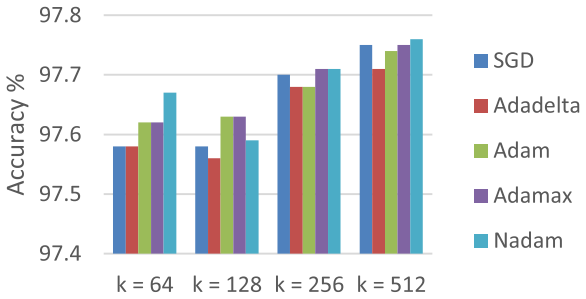


FIGURE 3. Accuracy comparison for five different optimizers for the Inception-ResNet-V2 network on the UCF101 dataset.

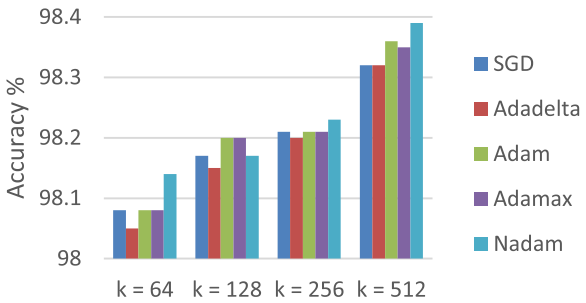


FIGURE 4. Accuracy comparison for five different optimizers for the NASNet-Large network on the UCF101 dataset.

D. EFFECT OF APPLYING PCA FOR THE FEATURES

Principal component analysis (PCA) is used to reduce the dimensionality of the video features. For Xception, Inception-ResNet-v2, and NASNet-Large pre-trained CNNs,

TABLE 1. Accuracy comparison based on the ucf101 dataset with and without using pca.

Codebook size	Accuracy %					
	Without using PCA			Using PCA		
	Xception	Inception-ResNet-v2	NASNet-Large	Xception	Inception-ResNet-v2	NASNet-Large
64	97.13	97.22	97.92	97.54	97.67	98.14
128	97.43	97.51	98.13	97.65	97.63	98.2
256	97.46	97.68	98.15	97.43	97.71	98.23
512	97.58	97.73	98.18	97.68	97.76	98.39

TABLE 2. Accuracy comparison based on the hmdb51 dataset with and without using pca.

Codebook size	Accuracy %					
	Without using PCA			Using PCA		
	Xception	Inception-ResNet-v2	NASNet-Large	Xception	Inception-ResNet-v2	NASNet-Large
64	65.42	67.32	67.26	66.91	68.89	71.83
128	65.87	67.76	67.78	66.24	69.18	72.54
256	65.76	67.65	68.12	66.79	70.21	72.46
512	65.89	67.24	68.33	67.16	69.33	73.13

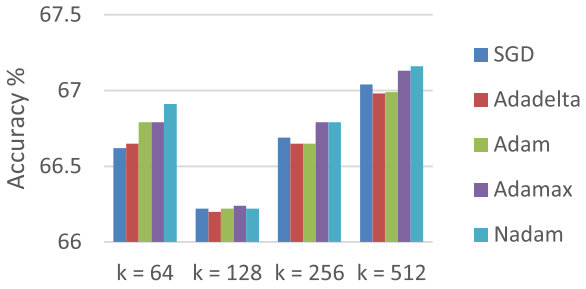


FIGURE 5. Accuracy comparison for five different optimizers for the Xception network on the HMDB51 dataset.

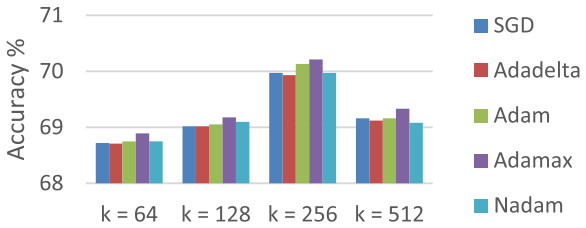


FIGURE 6. Accuracy comparison for five different optimizers for the Inception-ResNet-v2 network on the HMDB51 dataset.

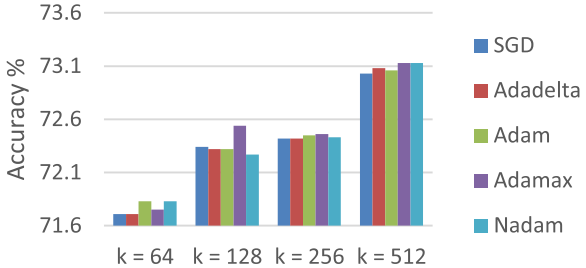


FIGURE 7. Accuracy comparison for five different optimizers for the NASNet-Large network on the HMDB51 dataset.

the extracted feature vector before applying PCA had a length of 2048, 1536, and 4032, respectively, which were reduced later using PCA. Tables 1 and 2 show a performance

TABLE 3. Time of execution with and without the optimization process for ucf101 dataset.

	Average time consumption (seconds)			
Codebook size	64	128	256	512
Pre-trained CNN without using optimizer	39.71	55.72	90.51	176.32
Pre-trained CNN using optimizer	40.68	57.12	94.32	182.61

TABLE 4. Time of execution with and without the optimization process for hmdb51 dataset.

	Average time consumption (seconds)			
Codebook size	64	128	256	512
Pre-trained CNN without using optimizer	21.82	39.99	80.91	164.49
Pre-trained CNN using optimizer	23.24	42.56	83.14	172.82

comparison with and without applying PCA on the training features for both datasets; the results show that the application of PCA has a positive influence on accuracy. This is because the redundant features are removed as PCA compresses them. Conversely, when no PCA was applied, the resulting accuracy performance was slightly lower than those in Tables 1 and 2. Nonetheless, the dimensions of the features must not be reduced excessively as this will cause deteriorations in accuracy.

E. EFFECT OF TIME EXECUTION

Tables 3 and 4 illustrate the processing times using the pre-trained features and SVM without and with the

TABLE 5. Comparison between codebook size after optimization for ucf101 dataset.

Codebook size after reduction for UCF101 dataset												
Network	Inception-ResNet-v2				Xception				NASNet-Large			
Codebook size	64	128	256	512	64	128	256	512	64	128	256	512
SGD	51	108	215	406	48	119	231	408	57	97	212	426
Adadelta	49	117	203	421	52	116	201	452	59	106	228	419
Adam	56	121	201	429	55	121	208	409	52	102	214	464
Adamax	55	115	213	409	52	115	214	493	49	119	231	426
Nadam	52	109	243	459	51	121	211	451	48	121	215	451

TABLE 6. Comparison between Codebook size after optimization FOR hmdb51 DATASET.

Codebook size after reduction for HMDB51 dataset												
Network	Inception-ResNet-v2				Xception				NASNet-Large			
Codebook size	64	128	256	512	64	128	256	512	64	128	256	512
SGD	59	121	228	501	56	102	214	498	49	118	212	451
Adadelta	57	119	219	469	52	113	207	425	48	97	226	468
Adam	54	118	234	472	56	103	214	421	46	109	241	428
Adamax	57	123	221	481	48	119	231	471	51	106	229	497
Nadam	58	119	235	469	49	107	233	471	50	114	246	425

TABLE 7. Accuracy performance comparison with the other state-of-the-art techniques.

Method	Accuracy (%) for UCF101	Accuracy (%) for HMDB51
iDT + IFV [3]	85.9 %	57.2 %
Two-stream CNN (fusion by SVM) [26]	88.0 %	59.4 %
Dynamic image (single RGB image) +CNN + iDT [27]	89.1 %	65.2 %
TDD +iDT [28]	91.5 %	65.9 %
FC-RNN [29]	91.6 %	61.8 %
3D ResNet 152 + iDT [30]	92.7 %	69.1 %
3D ResNext101 + iDT [31]	90.3 %	58.4 %
Temporal-Spatial Mapping [32]	94.3 %	72.7 %
ResNet + multiplicative gate mechanism [42]	94.9 %	72.2 %
DenseNet +Knowledge distillation network [43]	93.8 %	66.9 %
Proposed method	**98.4 %**	**73.2 %**

TABLE 8. Time of execution using softmax and cnn classifier for ucf101 and hmdb51 datasets.

	Average time consumption (seconds)		
Network	**Xception**	**Inception-ResNet-v2**	**NASNet-Large**
UCF101 dataset	264	347	516
HMDB51 dataset	191	285	415

optimization process, respectively. By using the optimization process, more steps are involved, such as retraining the optimized codebook using SVM and getting the SVM normal vector of the hyperplanes and the codebook optimization vector. The results indicate that although these steps take more time, the processing time is not significantly affected.

F. CODEBOOK REDUCTION SIZE

Tables 5 and 6 show the codebook size after removing the redundant elements using the optimization techniques and comparing it to the original codebook sizes. In all experiments, the codebook size was reduced compared to the original size $2 \times K \times D$ but still did not reach the size of the VLAD codebook of $K \times D$.

For the time consumption comparison to the state-of-the-art techniques, due to the different processors used by different authors, a numerical comparison is not practical. Instead, we summarize the comparison as follows, some authors use the traditional methods which does not include high computational complexity [19], [20], [21], [22], [23], [24]. While others improve on CNN by adding more trajectories to the network to improve the accuracy which increase the processing time [35], [37], [38], [39]. There are others who use 3D CNNs which include a huge training time which involve a huge processing time [29], [30]. On the other hand, the proposed method uses the optimization approach that does not affect the processing time.

Table 8 shows the processing time using the pre-trained CNNs. The first step is to freeze the weights of the layers for

each network. Then the final layers which are a fully connected layer followed by a SoftMax layer and the classifier are replaced to be suited to the new datasets. The table shows that the simulation time for the three networks is more than that of the proposed method.

G. COMPARISON WITH STATE-OF-THE-ART METHODS

Finally, to illustrate the superior performance of our proposed method, we compared it with other state-of-the-art methods such as the handcrafted approach [3] and methods based on fusion between CNN and iDT features [26], [27], [28], [29], [30], [31]. Table 7 shows that the proposed approach outperformed other methods for both UCF101 and HMDB51 datasets. Our proposed method achieved 98.4% classification accuracy for the UCF101 dataset, which is 5.7% better than the state-of-the-art method using 3D ResNet 152 + iDT [30]. For the HMDB51 dataset, the proposed method outperformed all methods and achieved an accuracy of 73.2%.

V. CONCLUSION

In this study, we have introduced a new technique that combines the pre-trained CNNs with the traditional techniques of codebook encoder and SVM classifier to extract the foreground features by optimizing the codebook elements. The optimized codebook was constructed based on identifying the discriminative Gaussians that represent the foreground while eliminating the non-effective codebook elements. The technique was tested on two benchmark datasets, i.e., UCF101 and HMDB51. The results show that using the optimization process to adjust the discriminative elements significantly improved the classification accuracy while reducing the codebook size compared to Fisher vector codebook size. In doing so, five different optimizers were applied and tested, and the results revealed that the time consumption did not increase much compared to other state-of-the-art algorithms. It is recommended for future studies to concentrate on improving the accuracy while reducing the training time to be applied for real-time applications.

REFERENCES

[1] F. Yang, Z. Ma, and M. Xie, "Codebook learning for image recognition based on parallel key SIFT analysis," *IEICE Trans. Inf. Syst.*, vol. 100, no. 4, pp. 927–930, 2017, doi: 10.1587/transinf.2016edl8167.

[2] D. Srivastava, R. Bakthula, and S. Agarwal, "Image classification using SURF and bag of LBP features constructed by clustering with fixed centers," *Multimedia Tools Appl.*, vol. 78, no. 11, pp. 14129–14153, Jun. 2019, doi: 10.1007/s11042-018-6793-8.

[3] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3551–3558, doi: 10.1109/ICCV.2013.441.

[4] C. S. Venegas-Barrera and J. Manjarrez, "Visual categorization with bags of keypoints," *Rev. Mex. Biodivers.*, vol. 82, no. 1, pp. 179–191, 2011. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.72.604

[5] F. Perronnin, M. Douze, P. Pe, C. Schmid, and J. Sa, "Into compact codes," *Analysis*, vol. 34, no. 9, pp. 1704–1716, 2012.

[6] G. J. Liu, Y. Liu, M. Z. Guo, P. N. Liu, and C. Y. Wang, "Non-negative locality-constrained linear coding for image classification," in *Proc. Int. Conf. Intell. Sci. Big Data Eng.*, Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 9242, 2015, pp. 462–471, doi: 10.1007/978-3-319-23989-7_47.

[7] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the Fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, Dec. 2013, doi: 10.1007/s11263-013-0636-x.

[8] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov, "Exploiting image-trained CNN architectures for unconstrained video classification," 2015, *arXiv:1503.04144*.

[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732, doi: 10.1109/CVPR.2014.223.

[10] N. Jaouedi, N. Boujnah, and M. S. Bouhlel, "A new hybrid deep learning model for human action recognition," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 32, no. 4, pp. 447–453, May 2020, doi: 10.1016/j.jksuci.2019.09.004.

[11] Z. Tu, W. Xie, D. Zhang, R. Poppe, R. C. Veltkamp, B. Li, and J. Yuan, "A survey of variational and CNN-based optical flow techniques," *Signal Process., Image Commun.*, vol. 72, pp. 9–24, Mar. 2019, doi: 10.1016/j.image.2018.12.002.

[12] Y. Abdulazeem, H. M. Balaha, W. M. Bahgat, and M. Badawy, "Human action recognition based on transfer learning approach," *IEEE Access*, vol. 9, pp. 82058–82069, 2021, doi: 10.1109/ACCESS.2021.3086668.

[13] A. Patil and M. Rane, "Convolutional neural networks: An overview and its applications in pattern recognition," *Smart Innov. Syst. Technol.*, vol. 195, pp. 21–30, Jan. 2021, doi: 10.1007/978-981-15-7078-0_3.

[14] Y. Tang, "Deep learning using linear support vector machines," in *Proc. ICML Workshop Challenges Represent. Learn. Workshop*, Jun. 2013, pp. 1–6.

[15] M. Dixit, S. Chen, D. Gao, N. Rasiwasia, and N. Vasconcelos, "Scene classification with semantic Fisher vectors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2974–2983, doi: 10.1109/CVPR.2015.7298916.

[16] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *Proc. CVPR*, Jun. 2010, pp. 3304–3311.

[17] J. Zhang, Y. Cao, and Q. Wu, "Vector of locally and adaptively aggregated descriptors for image feature representation," *Pattern Recognit.*, vol. 116, Aug. 2021, Art. no. 107952, doi: 10.1016/j.patcog.2021.107952.

[18] K. Simonyan, O. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild," in *Proc. Brit. Mach. Vis. Conf.*, 2013, pp. 1–11.

[19] S. A. R. Abu-Bakar, "Advances in human action recognition: An updated survey," *IET Image Process.*, vol. 13, no. 13, pp. 2381–2394, Nov. 2019, doi: 10.1049/iet-ipr.2019.0350.

[20] Y. Li, C. Yang, L. Zhang, R. Xia, L. Fan, and W. Xie, "A novel SURF based on a unified model of appearance and motion-variation," *IEEE Access*, vol. 6, pp. 31065–31076, 2018, doi: 10.1109/ACCESS.2018.2832290.

[21] M. F. Aslan, A. Durdu, and K. Sabanci, "Human action recognition with bag of visual words using different machine learning methods and hyperparameter optimization," *Neural Comput. Appl.*, vol. 32, no. 12, pp. 8585–8597, Jun. 2020, doi: 10.1007/s00521-019-04365-9.

[22] S. Nazir, M. H. Yousaf, J.-C. Nebel, and S. A. Velastin, "A bag of expression framework for improved human action recognition," *Pattern Recognit. Lett.*, vol. 103, pp. 39–45, Feb. 2018, doi: 10.1016/j.patrec.2017.12.024.

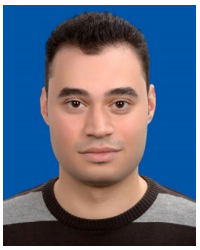
[23] I. C. Duta, J. R. R. Uijlings, B. Ionescu, K. Aizawa, A. G. Hauptmann, and N. Sebe, "Efficient human action recognition using histograms of motion gradients and VLAD with descriptor shape information," *Multimedia Tools Appl.*, vol. 76, no. 21, pp. 22445–22472, Nov. 2017, doi: 10.1007/s11042-017-4795-6.

[24] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Comput. Vis. Image Understand.*, vol. 150, pp. 109–125, Sep. 2016, doi: 10.1016/j.cviu.2016.03.013.

[25] K. Yamada, S. Ito, N. Kaneko, and K. Sumi, *Human Action Recognition via Body Part Region Segmented Dense Trajectories* (Lecture Notes in Computer Science), vol. 11367. Cham, Switzerland: Springer, 2019.

[26] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1, Jan. 2014, pp. 568–576.

[27] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3034–3042, doi: 10.1109/CVPR.2016.331.
[28] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 4305–4314.
[29] X. Yang, P. Molchanov, and J. Kautz, "Multilayer and multimodal fusion of deep neural networks for video classification," in *Proc. 24th ACM Int. Conf. Multimedia*, Oct. 2016, pp. 978–987, doi: 10.1145/2964284.2964297.
[30] Y. Huang, Y. Guo, and C. Gao, "Efficient parallel inflated 3D convolution architecture for action recognition," *IEEE Access*, vol. 8, pp. 45753–45765, 2020, doi: 10.1109/ACCESS.2020.2978223.
[31] H. Wang and J. Li, "Human action recognition algorithm based on multi-feature map fusion," *IEEE Access*, vol. 8, pp. 150945–150954, 2020, doi: 10.1109/ACCESS.2020.3017076.
[32] X. Song, C. Lan, W. Zeng, J. Xing, X. Sun, and J. Yang, "Temporal–spatial mapping for action recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 3, pp. 748–759, Mar. 2020, doi: 10.1109/TCSVT.2019.2896029.
[33] S. Zebhi, S. Almodarresi, and V. Abootalebi, "Human activity recognition based on transfer learning with spatio-temporal representations," *Int. Arab J. Inf. Technol.*, vol. 18, no. 6, pp. 839–845, 2021, doi: 10.34028/iajit/18/6/11.
[34] C. Aly, F. Salleh Abas, and H. Ann Goh, "Human action recognition using pre-trained convolutional neural networks," in *Proc. 2nd Int. Conf. Video, Signal Image Process.*, Dec. 2020, pp. 30–34, doi: 10.1145/3442705.3442710.
[35] Z. Tu, W. Xie, Q. Qin, R. Poppe, R. C. Veltkamp, B. Li, and J. Yuan, "Multi-stream CNN: Learning representations based on human-related regions for action recognition," *Pattern Recognit.*, vol. 79, pp. 32–43, Jul. 2018, doi: 10.1016/j.patcog.2018.01.020.
[36] N. M. Zamri, G. F. Ling, P. Y. Han, and O. S. Yin, "Vision-based human action recognition on pre-trained AlexNet," in *Proc. 9th IEEE Int. Conf. Control Syst., Comput. Eng. (ICCSCE)*, Nov. 2019, pp. 1–5, doi: 10.1109/ICCSCE47578.2019.9068586.
[37] S. A. Kumar, "Human action recognition based on spatio-temporal feature extraction using pre-trained CNN model and LSTM approach," *Int. J. Res. Advent Technol.*, vol. 7, no. 2, pp. 238–242, Mar. 2019, doi: 10.32622/ijrat.72201959.
[38] K. Muhammad, A. Ullah, A. S. Imran, M. Sajjad, M. S. Kiran, G. Sannino, and V. H. C. de Albuquerque, "Human action recognition using attention based LSTM network with dilated CNN features," *Future Gener. Comput. Syst.*, vol. 125, pp. 820–830, Dec. 2021, doi: 10.1016/j.future.2021.06.045.
[39] J. Dong, W. Yang, Y. Yao, and F. Porikli, "Knowledge memorization and generation for action recognition in still images," *Pattern Recognit.*, vol. 120, Dec. 2021, Art. no. 108188, doi: 10.1016/j.patcog.2021.108188.
[40] S. Khan, M. A. Khan, M. Alhaisoni, U. Tariq, H.-S. Yong, A. Armghan, and F. Alenezi, "Human action recognition: A paradigm of best deep learning features selection and serial based extended fusion," *Sensors*, vol. 21, no. 23, p. 7941, Nov. 2021, doi: 10.3390/s21237941.
[41] X. Zhou and L. Zhang, "SA-FPN: An effective feature pyramid network for crowded human detection," *Int. J. Speech Technol.*, vol. 52, no. 11, pp. 12556–12568, Sep. 2022, doi: 10.1007/s10489-021-03121-8.
[42] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7445–7454, doi: 10.1109/CVPR.2017.787.
[43] W. Hao and Z. Zhang, "Spatiotemporal distilled dense-connectivity network for video action recognition," *Pattern Recognit.*, vol. 92, pp. 13–24, Aug. 2019, doi: 10.1016/j.patcog.2019.03.005.
[44] B. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "CNN," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2012.
[45] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807, doi: 10.1109/CVPR.2017.195.
[46] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 4278–4284, doi: 10.1609/aaai.v31i1.11231.
[47] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710, doi: 10.1109/CVPR.2018.00907.
[48] N. Singh, Y. Hamid, S. Juneja, G. Srivastava, G. Dhiman, T. R. Gadekallu, and M. A. Shah, "Load balancing and service discovery using Docker swarm for microservice based big data applications," *J. Cloud Comput.*, vol. 12, no. 1, pp. 1–10, Jan. 2023, doi: 10.1186/s13677-022-00358-7.
[49] M. Gamal, S. Abu-Bakar, and U. U. Sheikh, "Stochastic recursive gradient descent optimization-based on foreground features of Fisher vector," *Proc. SPIE*, vol. 12342, p. 152, Oct. 2022, doi: 10.1117/12.2644640.
[50] L. Bottou, "Online learning and stochastic approximations," *Online Learn. Neural Netw.*, vol. 17, no. 9, pp. 1–35, 2018.
[51] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*.
[52] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15.
[53] T. Dozat, "Incorporating Nesterov momentum into Adam," in *Proc. ICLR Work.*, no. 1, 2016, pp. 2013–2016.
[54] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," 2012, *arXiv:1212.0402*.
[55] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2556–2563, doi: 10.1109/ICCV.2011.6126543.
[56] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proc. 18th ACM Int. Conf. Multimedia*, Oct. 2010, pp. 1469–1472, doi: 10.1145/1873951.1874249.



MOHAMED GAMAL M. KAMALELDIN received the B.S. and M.S. degrees in electronics and communications engineering from the Arab Academy for Science, Technology and Maritime Transport, Egypt, in 2008 and 2013, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with Universiti Teknologi Malaysia, Johor, Malaysia. Since 2013, he has been working as an Assistant Lecturer at the Arab Academy for Science, Technology and Maritime Transport. His research interests include computer vision, human action recognition, digital signal processing, and pattern recognition.

SYED A. R. ABU-BAKAR (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from Clarkson University, Potsdam, NY, USA, the MSEE degree from Georgia Tech, and the Ph.D. degree from the University of Bradford, U.K. He has been with the Faculty of Electrical Engineering, Universiti Teknologi Malaysia, since 1992. In 2004, he formed the Computer Vision, Video, and Image Processing Research Laboratory, and since then, he became the Head. He is currently a Full Professor with the Electronics and Computer Engineering Department. He has published more than 150 scientific articles, both at national and international levels. His research interests include computer vision and image processing with applications in video-based security and surveillance, medical image processing, and biometrics. He received the Meritorious Regional Chapter Service Award from the IEEE Signal Processing Society, in 2019. He was the Chair of the IEEE Signal Processing Society Malaysia Chapter, from 2014 to 2018.

USMAN ULLAH SHEIKH received the Ph.D. degree in image processing and computer vision from Universiti Teknologi Malaysia, in 2009. He is currently a Senior Lecturer with Universiti Teknologi Malaysia. His research interests include computer vision, machine learning, and embedded systems design.