applied sciences

MDPI

Article

Video-Based Recognition of Human Activity Using Novel Feature Extraction Techniques

Obada Issa and Tamer Shanableh *

Department of Computer Science and Engineering, American University of Sharjah, Sharjah P.O. Box 26666, United Arab Emirates; b00071518@aus.edu
* Correspondence: tshanableh@aus.edu

Abstract: This paper proposes a novel approach to activity recognition where videos are compressed using video coding to generate feature vectors based on compression variables. We propose to eliminate the temporal domain of feature vectors by computing the mean and standard deviation of each variable across all video frames. Thus, each video is represented by a single feature vector of 67 variables. As for the motion vectors, we eliminated their temporal domain by projecting their phases using PCA, thus representing each video by a single feature vector with a length equal to the number of frames in a video. Consequently, complex classifiers such as LSTM can be avoided and classical machine learning techniques can be used instead. Experimental results on the JHMDB dataset resulted in average classification accuracies of 68.8% and 74.2% when using the projected phases of motion vectors and video coding feature variables, respectively. The advantage of the proposed solution is the use of FVs with low dimensionality and simple machine learning techniques.

Keywords: activity recognition; high-efficiency video coding; machine learning; motion vectors

Citation: Issa, O.; Shanableh, T. Video-Based Recognition of Human Activity Using Novel Feature Extraction Techniques. *Appl. Sci.* **2023**, *13*, 6856. https://doi.org/10.3390/app13116856

Academic Editor: Vincent A. Cicirello

Received: 29 April 2023
Revised: 22 May 2023
Accepted: 1 June 2023
Published: 5 June 2023

1. Introduction

Although it is simple for humans to perceive and identify activities in videos, automating this process is challenging. Activity recognition is crucial to interpreting videos. Applications of activity recognition include automated surveillance, monitoring elderly behavior, human–computer interaction, content-based video retrieval, and video summarization. Activity recognition can be broken down into the following activities: First, activity classification that aims to categorize an image or video with the appropriate title (such as "cooking", "writing", etc.). Second, given a specific action and a video as input, action localization attempts to pinpoint the precise location and timestamp in the video at which the action is occurring.

Activity recognition has recently been conducted using a variety of techniques, such as wearable sensors, wireless sensors, or network-based sensors. However, compared to other modalities, video-based action recognition systems attract interest due to their high recognition rate and huge availability of video datasets. In this work, we focus on categorizing a video with an appropriate activity title using video-based solutions.

A decade ago, researchers mostly relied on features extracted from the motion history image (MHI) [1] or optical flow fields [2]. The MHI method represents the motion information of an action in a grayscale image, whereas optical flow represents the displacement of pixels between consecutive frames. These features were fed into classifiers, such as support vector machines (SVMs) or decision trees, to recognize the action. Traditional approaches to HAR relied on handcrafted features and classifiers, such as SVMs and hidden Markov models (HMMs) [3]. These methods were computationally efficient, but their performance was limited by the quality of the handcrafted features.

On the other hand, the literature shows that the usage of deep learning enhances activity recognition performance. Due to its enhanced performance in applications such as

surveillance, object categorization, biometrics, and a few others, deep learning is receiving a lot of attention in the field of computer vision. Recently, there has been a noticeable rise in the use of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) in the research of HAR [4]. CNNs are commonly used for the extraction of spatial information from video frames, while RNNs are usually used to capture temporal dependencies between features. Graph neural networks (GNNs) and attention methods have also been studied in conjunction with existing methods to achieve better interpretability of human activities in videos [4]. With these methods or techniques, HAR performance has significantly increased, especially for longer videos with complex moves.

There are several research papers in the literature that use deep learning for activity recognition, specifically, human action recognition or "HAR". For instance, in [5], the authors introduced "Dynamic Images", which are temporal representations of videos based on rank pooling, and then use these images on pre-trained CNNs. A framework was developed by the authors of [6] where they combined spatial and temporal contexts in one self-supervised framework without the need for extensive preprocessing. They divided video frames into grids of patches and trained a network to solve jigsaw puzzles on them. Due to the lack sufficient dataset samples at the time, in [7] it was proposed extending existing video datasets by applying action recognition on millions of videos over social platforms and categorizing them into the existing categories on mentioned datasets. In [8], the authors extended on the work conducted by the authors of [7] and proposed a neural operation which encodes spatiotemporal features by imposing a weight-sharing constraint on learnable parameters. They performed 2D convolution along three views of volumetric video data (H $\times$ W, T $\times$ H, W $\times$ T) to learn spatial appearance and temporal motion cues. In [9], the authors used VGG19 as their choice for a pre-trained CNN along with multi-view features computed from vertical and horizontal gradients. The best features were then used in a Naïve Bayes model for recognition.

A unique training procedure was introduced by the authors of [10], who used a pre-trained CNN for the spatial stream on single frames of a video, and for the temporal stream, they extracted three different RGB frames at different times in the video, converted them to grayscale, and assigned each to an RGB plane. Afterwards, the three images were merged into a Stacked Grayscale 3channel Image (SG3I) which then could be used with a conventional pre-trained CNN. In [11], the work looked into improving motion representations by tuning the spatial stream to predict the outputs of the temporal stream, eliminating the need for a two-stream architecture, and combining them into a single stream. In [12], the authors opted to use dilated CNNs to extract salient discriminative features and then feed them into a bidirectional LSTM to learn dependencies followed by an attention mechanism to enhance the recognition performance. Researchers in [13] combined spatial and temporal features into one set of spatiotemporal features and used the OFF CNN for feature extraction.

In a recent work [14], Optical Flow was used to represent the motion between video frames on a pixel basis. This result was arranged as an Optical Flow image and then converted into a feature vector using either VGG-Net or GoogleNet. Since such algorithms result in high-dimensionality feature vectors, the dimensionality of the results was reduced, and a multi-channel 1D-CNN was used to classify the resultant time series.

In this work, on the other hand, we utilized the High-Efficiency Video Coder (HEVC) [15] to compute feature variables for each video frame. During the HEVC coding process, many variables were computed to optimize the processes of intra- and inter-frame coding, taking into account the spatiotemporal activities for the video frames. We then eliminated redundancies in the temporal domain of the feature vector sequences by means of computing the measures of central tendencies and depression of each variable and in terms of PCA projections. Random forests (RFs) [16] were then used for model generation and classification.

In short, HAR is a major field of study that has advanced over time. Traditional methods depended on manually created features and classifiers, but recent developments in deep learning have resulted in considerable performance gains. The creation of more

precise and reliable models has also been aided by the availability of carefully curated large-scale datasets.

The rest of the paper is organized as follows. Section 2 introduces the system overview of the proposed solution including feature extraction and temporal domain elimination. Section 3 describes the used dataset, Section 4 presents the experimental setup and results, and Section 5 concludes the paper.

2. System Overview of Proposed Solution

In the proposed system, the AVI videos of the used dataset are first converted into YUV images, followed by HEVC video coding. We modified the HEVC coder to store feature variables and motion vectors for each video frame. These variables and motion vectors are then used to generate the feature vectors of each video frame as detailed in the sections to follow. The reason we extracted feature variables from HEVC is that such coders compute a rich set of variables that depend on motion estimation and compensation, hence capturing the activity across the temporal domain. In HEVC, this is performed on 64×64 blocks of pixels which are known as coding units. Such units are recursively partitioned into smaller blocks of pixels to capture both temporal and spatial variations in a video frame. The proposed system overview is illustrated in Figure 1.

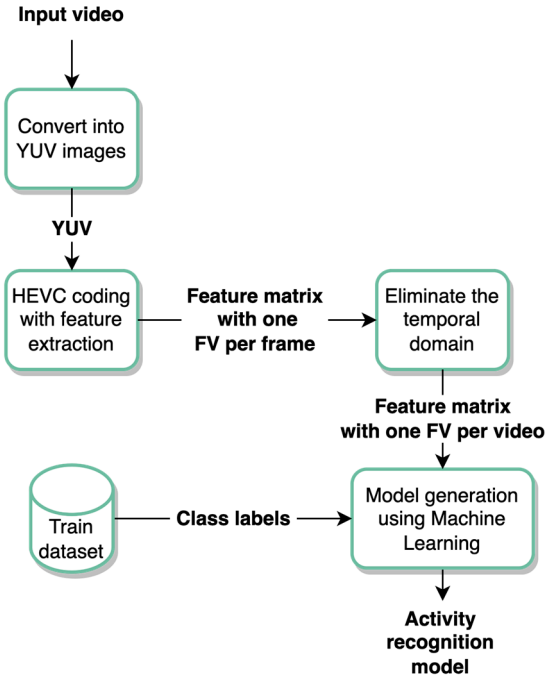


Figure 1. System overview of proposed solution.

2.1. Proposed Features Using HEVC

In this work, we propose the use of video coding standards to extract features from videos containing human activities. Namely, we modified the HEVC video coder to generate feature variables as detailed next. Rich feature sets can be generated from the HEVC coding process as the process of motion estimation is based on quadratic recursive splitting of the coding units. Such units vary in depth from 0, which is a block of 64×64 pixels, to 3, which is a block of 16×16 pixels. Coding units are further split into prediction units varying in size from 32×32 to 4×4 pixels.

Consequently, the extracted feature variables are based on coding unit partitioning, count of coding bits spent on coding unit splits, percentage of intra-coded and skipped coding units, histogram of coding unit depths, sum of absolute differences in prediction errors (i.e., the difference between a current pixel block and its best-match location in a previous frame), coding distortion of coding units, histograms of motion vectors, variances

in motion vectors, and count of coding bits spent on coding motion vector differences. This results in 67 feature variables per video frame.

The authors previously used a similar set of features in their work on identifying key frames in video sequences [17]. The feature extraction solution is illustrated in Figure 2.

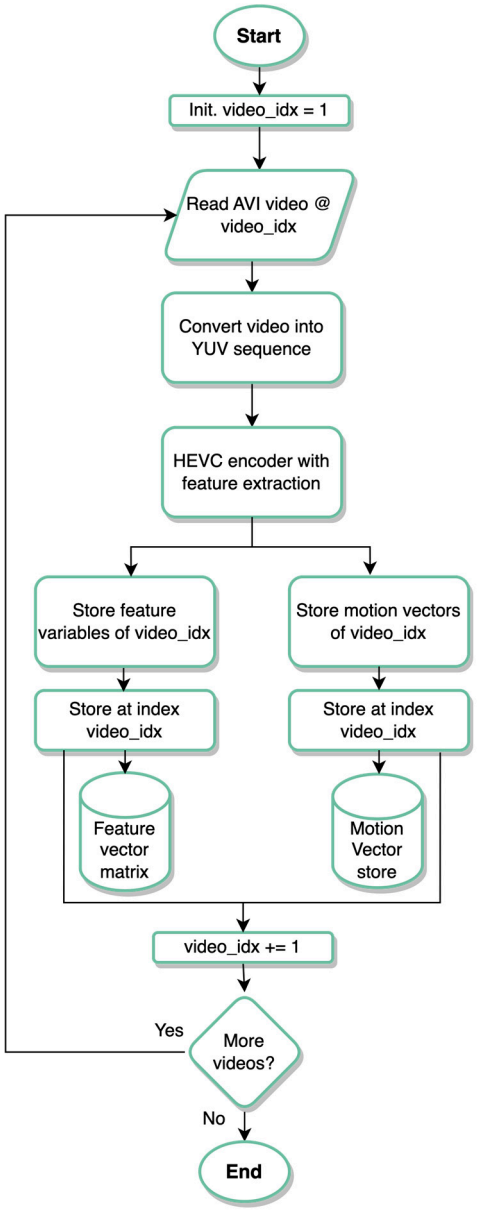


Figure 2. Flowchart of overall proposed feature extraction.

The original videos of the dataset used in this work are in avi format; hence, as detailed in the flowchart in Figure 2, avi videos are converted into YUV sequences suitable for video coding. A HEVC video coder is used to encode the YUV sequences and generate feature variables and motion vectors which constitute the basis of our proposed solution. This process is repeated for every frame in human activity video.

2.2. Proposed Features Using Motion Vector Phases

As illustrated in Figure 2, during HEVC coding, the motion vectors of each video frame are stored. These motion vectors are then used for generating feature vectors for the video sequences as follows. The motion vectors are listed in V_x, V_y pairs for each video frame in a video sequence. The phases of the motion vectors are computed as $\tan^{-1}(V_y/V_x)$;

thus, each V_x, V_y pair is converted into a scalar representing the phase of the motion vector. The resulting lists of phases pertaining to one video frame are then projected using PCA. This results in a list of projected phases with a length equal to the number of video frames in a video minus one (as motion estimation starts from the second video frame). The reason phases of motion vectors are computed is that each human activity is mainly characterized by the phases of the motion as opposed to its the magnitude, as the later depends more on the speed at which an activity is carried out. This arrangement is further illustrated in Figure 3. Once the motion vectors are projected, the generated feature vectors are stored in a feature matrix which can then be used for generating the classification model.

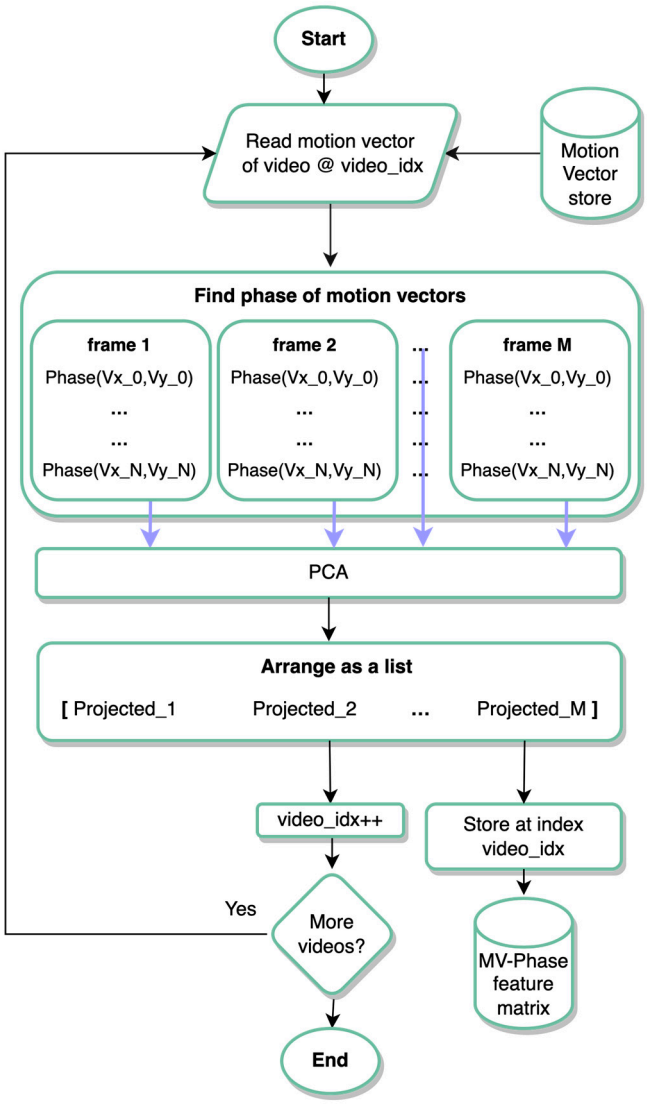


Figure 3. Appending HEVC features with motion-vector-based features.

2.3. Proposed Elimination of Temporal Domain

It is worth noting that the proposed feature extraction solutions in Sections 2.1 and 2.2 generate a sequence of feature vectors per video sequence. A classification model for such a sequence can be generated using deep learning or, more specifically, long short-term memory (LSTM) [18] (or even 1D-CNNs). However, in this work, to reduce the computational complexity, we propose to collapse all feature vectors of a video sequence into one feature vector. For the HEVC feature variables, the temporal domain can be eliminated by computing the averages and the standard deviations of each feature variable. This results in a feature vector with a length of 67x2 variables for each video sequence.

As for the phases of the motion vectors of a given video frame, the temporal domain is eliminated by projecting them using PCA, as illustrated in Figure 3. Consequently, all phases are represented using a single 1D list with a length equal to the number of frames in a video sequence. The length of the final feature vector per video sequence is equal to the number of video frames in the sequence minus one, as motion estimation starts from the second video frame.

Once the temporal dependencies of the feature vector sequence are eliminated using this approach, classical machine learning techniques can be used for model generation. In this work, we used random forests.

2.4. Model Generation and Classification Setup

In this work, we used random forests for classification. The number of trees grown was 180, which was determined imperially. Feature variable selection was enabled, where a set of RF trees are generated against the class of a human activity. Such trees were trained on a subset of feature variables. For each variable, an informative subset of feature variables was found by leveraging the usage statistics. Namely, if a feature variable is frequently selected, then it is considered as a good variable to retain.

The importance of each of the retained variables in predicting the correct human activity establishes out-of-bag data used to select the feature variables, with importance scores making up 90% of the total importance score. The out-of-bag data represent the feature vectors that were left out during the model generation process of a given tree in the random forest.

3. Dataset

In this work, we used the JHMDB dataset [19] which contains 928 one-to-two-second videos categorized into 21 classes of daily human activities, such as brushing hair, catching an object, clapping, playing golf, pushing objects, running, sitting down, walking, waving, and so forth. The speed of body parts are different across video sequences in the database. The dataset has three splits of 70% for training and 30% for testing. The classification results are based on the average accuracies of those splits. The reason this dataset was used is that it contains a smaller number of video sequences, which makes it challenging for classification. Figure 4 contains sample images from the JHMDB videos.

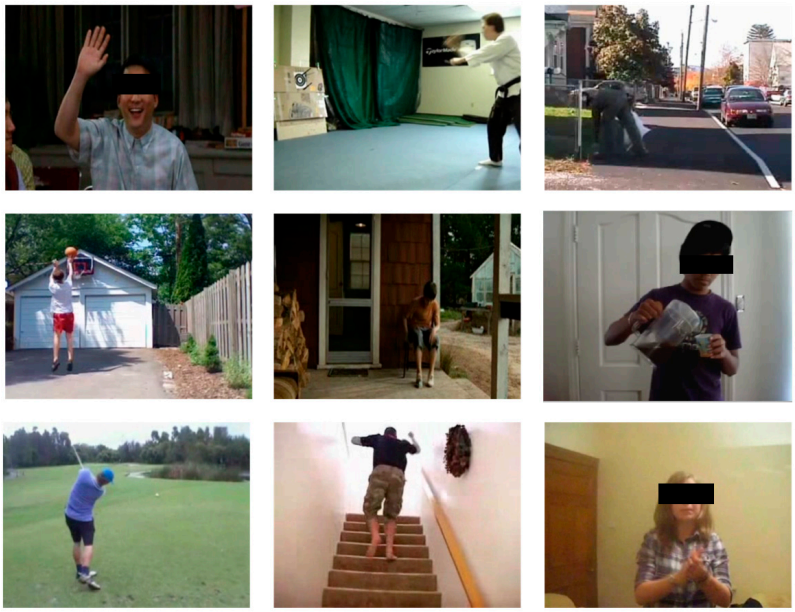


Figure 4. Sample images from JHMDB dataset of different activities such as (from top left to right): wave, throw knife, pick, throw ball, sit, pour, golf, climb stairs, and clap.

4. Results

We start by reporting the classification accuracies of the proposed solutions. As practiced in the literature, we report the average classification accuracy of the three data splits of the JHMDB dataset. We also compare our results against similar solutions reported in the literature. Our comparison against the existing literature is based on what is summarized in the recent work [10] which presented recent and outstanding results using an innovative solution, as summarized in Section 1.

In the proposed solution, the number of trees used in the random forests classifier was 180, which was set empirically. The length of the feature vector of the HEVC solution of Section 2.1 was 67x2, pertaining to the mean and standard deviation of the HEVC features. In contrast, the length of the feature vector of the projected motion vector phases in Section 2.2 was equal to the number of video frames in a sequence (minus one, as motion estimation starts from the second frame).

In the HEVC video coding, we used typical coding parameters such as a maximum coding unit size of 64 × 64 pixels and a motion estimation range of 32 pixels. The quantization scale was set to 22, which is neither coarse nor fine.

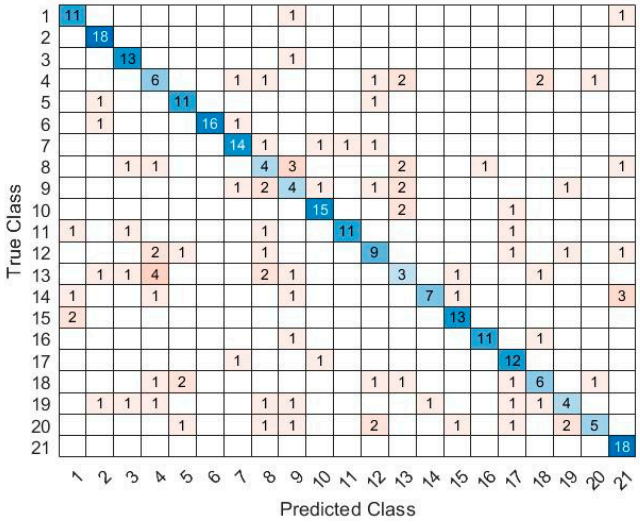
As presented in Table 1, in comparison with the reviewed work, the proposed solution, which is based on projecting motion vector phases, scored with an average accuracy of 68.8%. This can be justified by the fact that each video frame in this solution was projected into one scalar point, as elaborated upon in Section 2.2. On the other hand, the solution based on feature variables from HEVC encoding ranked the second with an average accuracy of 74.2%. The latter solution achieved higher classification accuracy, as motion vectors are an integral part of the HEVC features in addition to many other coding features, as listed in Section 2.1.

Table 1. Comparison of average classification accuracy of the proposed solutions against state-of-the-art solutions using the JHMDB dataset.

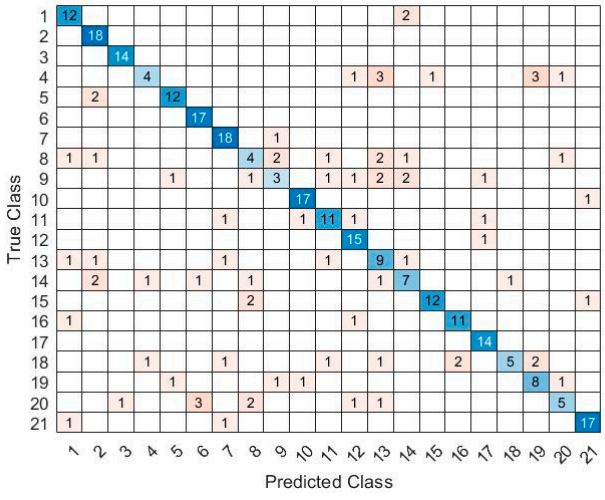
Method	Accuracy
PoTion [20]	57.0%
P-CNN [21]	61.1%
Action Tubes [22]	62.5%
EleAtt-GRU [23]	62.9%
Zero Padding (with features from VGG-Net) [14]	68.8%
Proposed projection of motion vector phases	**68.8%**
PA3D [24]	69.5%
Temporal Pyramid (with features from VGG-Net) [14]	69.8%
MR Two-Stream R-CNN [25]	71.1%
DR2N [26]	71.8%
Zero Padding (with features from GoogleNet) [14]	72.3%
Generalized Rank Pooling with the improved Trajectory Features [27]	73.7%
Proposed feature variables based on HEVC coding	**74.2%**
Temporal Pyramid (with features from GoogleNet) [14]	74.8%

In deep learning, typically, multi-solutions can be fused to generate more accurate results. Fusion can be achieved on score level or on a feature level. The former is irrelevant to RFs, as classification scores are not generated, and the fact that only two solutions are proposed means that the solutions cannot be fused on a class label level, as ties cannot be resolved with an even number of classifiers. This leaves the option of feature fusion, where the features of the projected motion vector phases and HEVC features are concatenated. Having tested that, we noticed that the classification accuracy did not improve. This is justified by the fact that HEVC features already include histograms of motion vectors generated from the motion estimation process of the coder. The fact that the second proposed solution is mainly based on motion vectors necessitates that fusing both features does not add class discriminating information and thus does not enhance the classification accuracy.

Confusion matrices pertaining to both proposed solutions are displayed in Figure 5. The proposed solution using HEVC feature variables works well for some videos but not for others. For example, as shown in the confusion matrix (Figure 5b), videos belonging to Class 7 are well classified, yet videos belonging to Class 8 are mainly misclassified. In the dataset, Class 7 is "Climb stairs" and class 8 is "Dive". With a closer look at the videos, it is observed that videos in Class 7 mainly have static backgrounds, whereas videos in Class 8 have backgrounds with noticeable temporal activities. When compressed in HEVC, the coder does not distinguish between foreground and background temporal activities; hence, the feature variables capture information irrelevant to the main human activity in the video. This can be listed as a limitation of the proposed solution. A potential solution is to segment and track the motion of the foreground object in a video sequence as proposed [28].



(**a**)



(**b**)

Figure 5. Confusion matrices pertaining to both proposed solutions; (**a**) projection of motion vector phases; (**b**) feature variables based on HEVC coding.

The advantage of this proposed solution, however, lies in its simplicity, where RFs are used instead pf deep learning, and FVs with low dimensionalities are generated from the video coding process.

The proposed solutions are mainly based on video coding. Taking into account that videos need to be compressed regardless of the application they are used in, feature

variables can be generated from the compression process without extra processes in the classification pipeline. As such, the computational cost is mainly restricted to the model generation and inference.

In Table 2, we present the average train and test of the proposed solution against the use of LSTM. The results were generated using MATLAB R2021b, and the time was measured using its cputime() function. The machine used to generate the results runs Windows 10 with a 10th gen Intel Core i9 processor, 16 GB RAM, and NVIDIA Quadro T2000 GPU with Max-Q Design.

Table 2. Average train and test times of the proposed solution against the use of LSTM.

	With Temporal Domain Elimination and RFs	**Without Temporal Domain Elimination and LSTM**
	HEVC Variables	
Train time (s)	181.4	1091.1
Test time (s)	2.26	2.96
	MV Phase Projections	
Train time (s)	74.79	2416.3
Test time (s)	2.27	2.68

As mentioned in Section 2.3, we proposed the elimination of the temporal of the feature vectors such that we end up with a single feature vector per video sequence. This allowed the proposed solution to use classical machine learning approaches for model generation and classification at a fraction of the computational cost of using deep learning, in this case, LSTM. We used a moderate 100 epochs in the LSTM training parameters with an initial learning rate of 1×10^{-4}. The layers used in the LSTM architecture include an LSTM layer with 2000 nodes followed by a 50% dropout layer. These layers are followed by softmax and classification layers. The size of the mini batch is 16, and the number of iterations per epoch is equal to the number of feature vectors divided by the size of the mini batch. These were chosen as they are the default settings in MATLAB's sequence classification.

Note that our proposed solutions are not designed for LSTM; the latter is used only to show the advantage in computational processing time of the proposed solutions.

The proposed solutions in Sections 2.1 and 2.2 are referred to here as "HEVC Variables" and "MV Phases". A number of observations are made from the results in Table 2. First, the proposed MV Phases solution has a faster model generation time than the proposed HEVC Variables solution; this is so as the later generates FVs with a length of 67 and the former generates FVs with a length of 48. Second, this observation is reversed when LSTM is used; with LSTM, the proposed temporal elimination solution is not used, and therefore the phases of all motion vectors of a video frame are presented, as is the classifier with a large dimensionality, which depends on the number of blocks in the frame. In contrast, in the HEVC Variables solution, the length of the FV remains 67, but without temporal elimination, we have one FV per frame as opposed to one FV per video sequence. Third, with the proposed temporal elimination, the model generation time of the proposed solutions are a fraction of those generated using LSTM. More specifically, when the HEVC Variables solution is used, the model generation time is 17% of that when using LSTM. Moreover, when the MV Phases solution is used, the model generation time is 3.1% of that when using LSTM. These are clear savings in terms of computation time.

5. Conclusions

In this paper, we proposed two approaches for feature extraction for the purpose of video-based activity recognition. HEVC video coding is used to generate feature variables and motion vectors. The HEVC coder computes a rich set of variables that depends on motion estimation and compensation, hence capturing the activity across the temporal domain. The first solution uses HEVC coding variables, resulting in 67 features per video

frame. The second solution uses the phases of the motion vectors of each video frame as a feature vector. In both solutions, we proposed to eliminate the temporal domain such that we end up with only one feature vector per video sequence, thus eliminating the need for complex deep learning techniques. This was achieved by computing the measure of central tendency and dispersion of HEVC variables and by projecting the motion vectors using PCA. Model generation and classification was performed using random forests. Lastly, the model generation time of the proposed solutions is a fraction of that generated using LSTM. More specifically, when the HEVC Variables solution is used, the model generation time is 17% of that when using LSTM, and when the MV Phases solution is used, the model generation time is 3.1% of that when using LSTM.

Author Contributions: Conceptualization, T.S.; methodology, T.S.; software, T.S.; funding acquisition, T.S.; validation, O.I. and T.S.; formal analysis, O.I. and T.S.; investigation, O.I. and T.S.; resources, O.I. and T.S.; data curation, O.I. and T.S.; writing—original draft preparation, O.I. and T.S.; writing—review and editing, O.I. and T.S.; visualization, O.I. and T.S.; supervision, O.I. and T.S.; project administration, O.I and T.S. All authors have read and agreed to the published version of the manuscript.

Funding: The work in this paper was supported, in part, by the Open Access Program from the American University of Sharjah. This paper represents the opinions of the author(s) and does not mean to represent the position or opinions of the American University of Sharjah.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset used in this work: http://jhmdb.is.tue.mpg.de/ (accessed on 1 October 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, H.; Klaser, A.; Schmid, C.; Liu, C.-L. Action Recognition by Dense Trajectories. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 3169–3176.
2. Wang, J.; Liu, Z.; Wu, Y.; Yuan, J. Learning Actionlet Ensemble for 3D Human Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 914–927. [CrossRef] [PubMed]
3. Laptev, I.; Marszalek, M.; Schmid, C.; Rozenfeld, B. Learning Realistic Human Actions from Movies. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
4. Le, V.-T.; Tran-Trung, K.; Hoang, V.T. A Comprehensive Review of Recent Deep Learning Techniques for Human Activity Recognition. *Comput. Intell. Neurosci.* **2022**, *2022*, 1–17. [CrossRef] [PubMed]
5. Bilen, H.; Fernando, B.; Gavves, E.; Vedaldi, A.; Gould, S. Dynamic Image Networks for Action Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3034–3042.
6. Ahsan, U.; Madhok, R.; Essa, I. Video Jigsaw: Unsupervised Learning of Spatiotemporal Context for Video Action Recognition. In Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa Village, HI, USA, 7–11 January 2019; pp. 179–189.
7. Ghadiyaram, D.; Tran, D.; Mahajan, D. Large-Scale Weakly-Supervised Pre-Training for Video Action Recognition. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 12038–12047.
8. Li, C.; Zhong, Q.; Xie, D.; Pu, S. Collaborative Spatiotemporal Feature Learning for Video Action Recognition. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 7864–7873.
9. Khan, M.A.; Javed, K.; Khan, S.A.; Saba, T.; Habib, U.; Khan, J.A.; Abbasi, A.A. Human Action Recognition Using Fusion of Multiview and Deep Features: An Application to Video Surveillance. *Multimed. Tools Appl.* **2020**. [CrossRef]
10. Kim, J.-H.; Won, C.S. Action Recognition in Videos Using Pre-Trained 2D Convolutional Neural Networks. *IEEE Access* **2020**, *8*, 60179–60188. [CrossRef]
11. Stroud, J.C.; Ross, D.A.; Sun, C.; Deng, J.; Sukthankar, R. D3D: Distilled 3D Networks for Video Action Recognition. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, CO, USA, 1–5 March 2020; pp. 614–623.
12. Muhammad, K.; Mustaqeem; Ullah, A.; Imran, A.S.; Sajjad, M.; Kiran, M.S.; Sannino, G.; de Albuquerque, V.H.C. Human Action Recognition Using Attention Based LSTM Network with Dilated CNN Features. *Future Gener. Comput. Syst.* **2021**, *125*, 820–830. [CrossRef]

13. Xu, J.; Song, R.; Wei, H.; Guo, J.; Zhou, Y.; Huang, X. A Fast Human Action Recognition Network Based on Spatio-Temporal Features. *Neurocomputing* **2021**, *441*, 350–358. [CrossRef]
14. Javidani, A.; Mahmoudi-Aznaveh, A. Learning Representative Temporal Features for Action Recognition. *Multimed. Tools Appl.* **2022**, *81*, 3145–3163. [CrossRef]
15. Sullivan, G.J.; Ohm, J.-R.; Han, W.-J.; Wiegand, T. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1649–1668. [CrossRef]
16. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
17. Issa, O.; Shanableh, T. CNN and HEVC Video Coding Features for Static Video Summarization. *IEEE Access* **2022**, *10*, 72080–72091. [CrossRef]
18. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
19. Jhuang, H.; Gall, J.; Zuffi, S.; Schmid, C.; Black, M.J. Towards Understanding Action Recognition. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 3192–3199.
20. Choutas, V.; Weinzaepfel, P.; Revaud, J.; Schmid, C. PoTion: Pose MoTion Representation for Action Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7024–7033.
21. Chéron, G.; Laptev, I.; Schmid, C. P-CNN: Pose-Based CNN Features for Action Recognition. *arXiv* **2015**, arXiv:1506.03607.
22. Gkioxari, G.; Malik, J. Finding Action Tubes. *arXiv* **2014**, arXiv:1411.6031.
23. Zhang, P.; Xue, J.; Lan, C.; Zeng, W.; Gao, Z.; Zheng, N. Adding Attentiveness to the Neurons in Recurrent Neural Networks. *arXiv* **2018**, arXiv:1807.04445.
24. Yan, A.; Wang, Y.; Li, Z.; Qiao, Y. PA3D: Pose-Action 3D Machine for Video Recognition. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 7914–7923.
25. Peng, X.; Schmid, C. Multi-Region Two-Stream R-CNN for Action Detection. In *Computer Vision–ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2016; Volume 9908, pp. 744–759. ISBN 978-3-319-46492-3.
26. Sun, C.; Shrivastava, A.; Vondrick, C.; Sukthankar, R.; Murphy, K.; Schmid, C. Relational Action Forecasting. *arXiv* **2019**, arXiv:1904.04231.
27. Cherian, A.; Fernando, B.; Harandi, M.; Gould, S. Generalized Rank Pooling for Activity Recognition. *arXiv* **2017**, arXiv:1704.02112.
28. Bertasius, G.; Torresani, L. Classifying, segmenting, and tracking object instances in video with mask propagation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9739–9748.