



## Article

# Exposing Video Compression History by Detecting Transcoded HEVC Videos from AVC Coding

Shan Bian <sup>1,\*</sup>, Haoliang Li <sup>2</sup>, Tianji Gu <sup>1</sup> and Alex Chichung Kot <sup>2</sup>

<sup>1</sup> College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China; terrify0608@yeah.net

<sup>2</sup> School of Electrical and Electronic Engineering, Nanyang Technology University, Singapore 639798, Singapore; lihaoliang@ntu.edu.sg (H.L.); eackot@ntu.edu.sg (A.C.K.)

\* Correspondence: bianshan@scau.edu.cn

Received: 30 November 2018; Accepted: 3 January 2019; Published: 8 January 2019



**Abstract:** The analysis of video compression history is one of the important issues in video forensics. It can assist forensics analysts in many ways, e.g., to determine whether a video is original or potentially tampered with, or to evaluate the real quality of a re-encoded video, etc. In the existing literature, however, there are very few works targeting videos in HEVC format (the most recent standard), especially for the issue of the detection of transcoded videos. In this paper, we propose a novel method based on the statistics of Prediction Units (PUs) to detect transcoded HEVC videos from AVC format. According to the analysis of the footprints of HEVC videos, the frequencies of PUs (whether in symmetric patterns or not) are distinguishable between original HEVC videos and transcoded ones. The reason is that previous AVC encoding disturbs the PU partition scheme of HEVC. Based on this observation, a 5D and a 25D feature set are extracted from I frames and P frames, respectively, and are combined to form the proposed 30D feature set, which is finally fed to an SVM classifier. To validate the proposed method, extensive experiments are conducted on a dataset consisting of CIF (352 × 288) and HD 720p videos with a diversity of bitrates and different encoding parameters. Experimental results show that the proposed method is very effective at detecting transcoded HEVC videos and outperforms the most recent work.

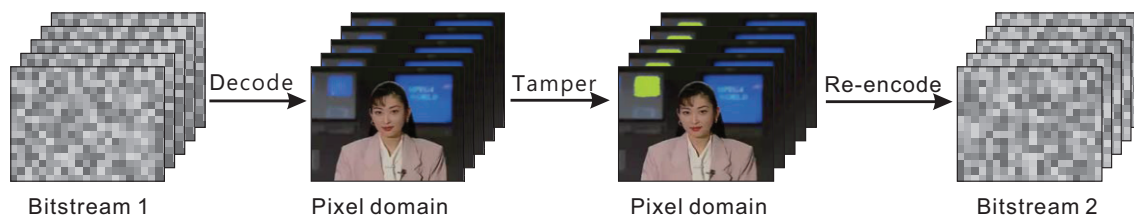
**Keywords:** video forensics; compression history; transcoded AVC/HEVC; prediction units

## 1. Introduction

The widespread utilization of inexpensive and portable video capture devices, such as camcorders and mobile phones, coupled with the surge in surveillance cameras, has led to a remarkable increase in the amount of digital video data. Besides, thanks to the ubiquitous Internet and various video sharing sites (e.g., YouTube, Douyin, TikTok, etc.), many users are pleased to upload their videos on the web as social media. To make videos more appealing, video owners usually edit their videos before uploading, such as scaling, cropping, frame-deletion/insertion, etc. Each of these alterations needs to be done in the pixel domain, so an altered video has to be decoded and then re-encoded inevitably, as illustrated in Figure 1.

When a video sequence is re-encoded, only the information about the most recent coding could be obtained by parsing the bitstream; namely, the re-encoding operation overwrites the previous encoding information. Therefore, a video can be disguised as any format with any encoding parameters by means of re-encoding. For instance, old or lower quality AVC content could be re-encoded by a new video codec, such as HEVC (High-Efficiency Video Coding), as if it were of natively high quality in HEVC format. (Please note that AVC stands for the H.264/AVC standard, and HEVC represents the H.265/HEVC standard in this paper.) This transcoded HEVC video would be easily tagged as “HEVC”

or “high quality” after being uploaded to video sharing sites, and then, it could attract more attention than deserved, earning advertising revenue.



**Figure 1.** Process of tampering with a video. A video is firstly decoded to the pixel domain in which the alteration is done. After editing operations, the video sequence in the pixel domain is re-encoded to a new bitstream. Please note that the decoded pictures are from the “Akiyo” video sequence downloaded from [1].

However, re-encoded videos are not totally untraceable. Each coding step inevitably leaves invisible characteristic clues/footprints in compressed video, which are very useful in investigating video compression history [2]. The knowledge of video compression history can help forensic analysts in many ways. For example, it can be used in video forensic research, e.g., to determine whether a video is original, or to identify the source device of a video since some encoding format might be incompatible with some devices [3]. Besides, it might be employed to detect video splicing since different video clips might have different encoding history. Moreover, the analysis of the processing history of the video compression could be introduced in video quality assessment because the encoding information obtained from a video bitstream might not represent the actual visual quality [4], e.g., a “high”-quality HEVC video re-encoded from an originally low-quality AVC content.

To expose video compression history, scholars and researchers have put in a lot of effort to study potential characteristics in compressed videos [5,6], such as DCT coefficient distribution [7–9], blocking artifacts of MPEG videos [10,11], the first-digit law of DCT coefficients [12,13], Markov-based features [14], prediction unit-based characteristics of HEVC videos [15–17], etc. Besides, video transcoding is also one of the common operations in video compression, so the issue of detecting transcoded videos also attracts some attention in the existing literature. For example, Bestagini et al. [3] proposed a method to estimate the codec and GOP size used in the first coding step based on the idempotency property that the video codec inherits from quantizers. However, its “recompress-and-observe” scheme leads to a high computational complexity, and the latest HEVC codec is not taken into consideration in their work. Costanzo et al. [18] observed that AVC encoding influences the decision strategy of the subsequent HEVC encoding regarding the motion prediction modes in B frames, so they presented a method to reveal AVC/HEVC transcoding and to estimate the previous Quantization Parameter (QP). However, this method can only be adopted in the special case that the encoding mode is set as constant QP mode and the QP in HEVC encoding is smaller than that of AVC. Besides, the feature set of their method is extracted from B frames that are rarely used in many practical applications based on the investigation. Yu et al. [19] proposed a detection method based on the histogram of the PU (Prediction Unit) amount in the first P frames of all GOPs. However, this method extracts features from only one P frame in each GOP, which might be unstable for different video content.

In this paper, we address the problem of the detection of transcoded HEVC videos from AVC format by presenting an effective method based on the statistics of PU types. According to the analysis of extensive experiments, it is found that the previous AVC compression influences the PU partition strategy in the subsequent HEVC encoding. As a result, the frequencies of PUs are distinguishable between original HEVC videos and transcoded ones. Based on this property, a 5D and a 25D feature set are extracted from I frames and P frames, respectively, and then, both sets are combined to form a 30D feature vector, which is finally fed to an SVM classifier. To validate the effectiveness of the proposed method, extensive experiments are conducted on CIF and high definition videos with

different encoding parameters and four bitrates. The experimental results show that the detection rates are satisfying and the method is robust to different encoding parameters.

The rest of the paper is organized as follows. Section 2 briefly introduces the HEVC codec and its PU partition strategy. Section 3 analyzes the footprints of PUs and presents the detection method. A thorough experimental validation is presented in Section 4. Finally, Section 5 draws the conclusion of this paper, along with future challenges.

## 2. Analysis of HEVC Footprints

High-Efficiency Video Coding (HEVC), known as H.265 or MPEG-H Part 2, is regarded as one of the successors to the widely-used AVC (H.264 or MPEG-4 Part 10). Currently, HEVC is supported by many implementations and products, such as Apple iOS and macOS [20], Samsung mobile phones [21], Cannon camcorders [22], etc. In this section, some features of HEVC are briefly introduced, and the footprints of PU partition are analyzed.

### 2.1. High-Efficiency Video Coding

The first edition of the HEVC standard was finalized in 2013. Since then, it has been continually extended, improved, and published by both ITU-T [23] and ISO/IEC [24]. The video coding layer of HEVC adopts the same hybrid technique (inter-/intra-picture prediction and 2D transform coding) used in all video compression standards [25]. For example, HEVC follows the classic approach of inter-picture and intra-picture prediction to exploit temporal and spatial statistical dependencies, respectively. However, compared to AVC, HEVC introduces various new features to enable significantly improved compression performance and bit-rate reduction [26].

Firstly, HEVC defines Coding Tree Units (CTU) as the basic processing unit to specify the decoding process. Different from the previous macroblock-based coding layer, the analogous structure in HEVC is CTU, which has a size selected by the encoder rather than a fixed size.

Secondly, Coding Units (CUs) are introduced in the HEVC standard. A CTU may contain only one CU or may be split into multiple CUs, each of which may also be split.

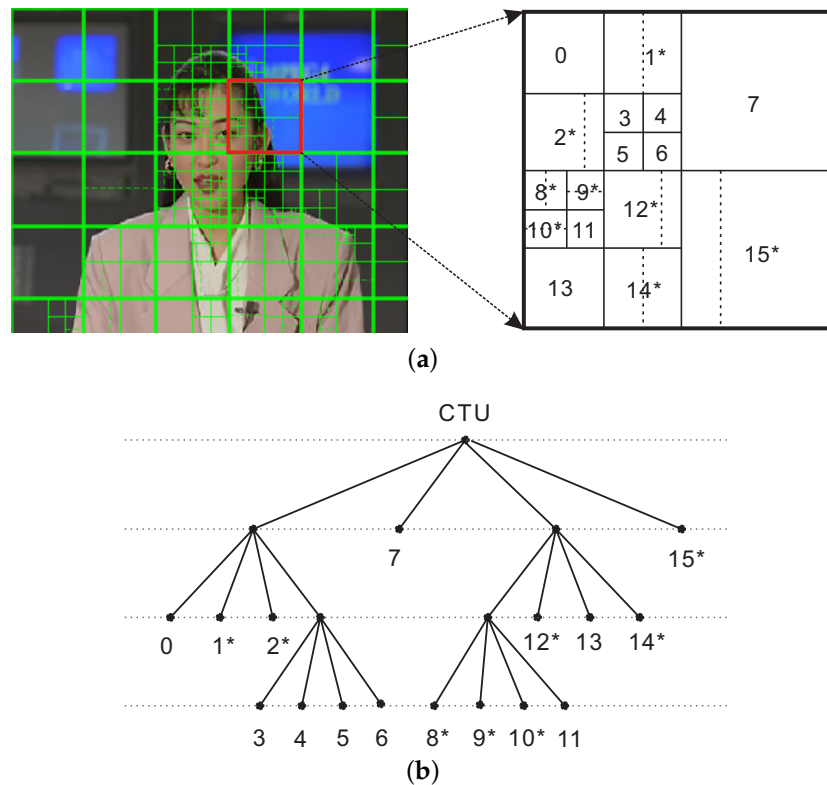
Thirdly, HEVC employs Prediction Units (PUs) for inter-/intra-picture prediction. PUs are split from its root CU level once, or not split. The supported PU sizes range from  $64 \times 64$  down to  $4 \times 4$ .

Moreover, there are a plenty of new features involved in HEVC coding, such as Transform Units (TUs) with various sizes, advanced motion vector prediction, motion compensation in quarter-sample precision, simplified in-loop deblocking filtering, etc.

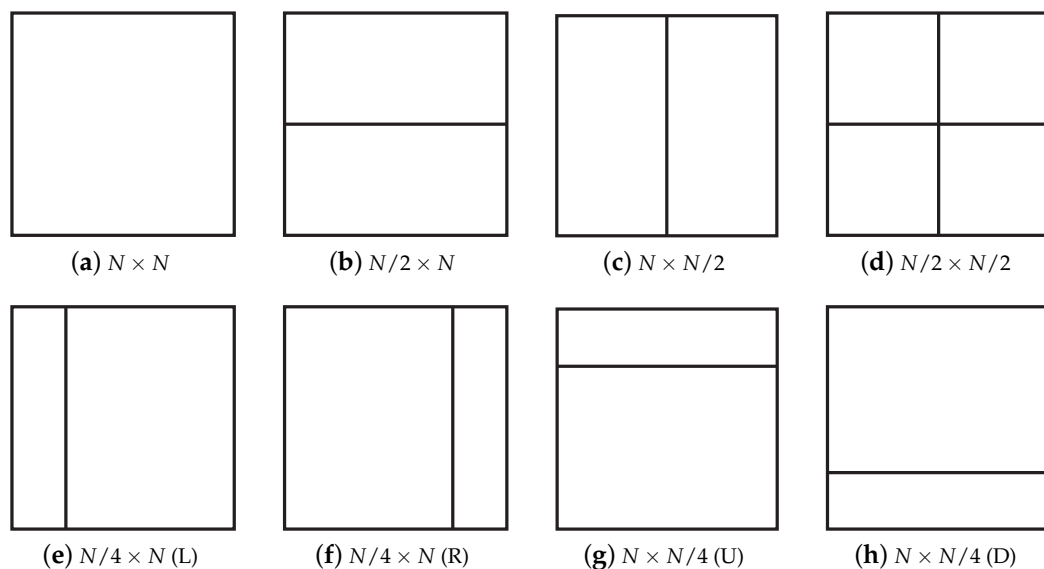
### 2.2. CU and PU Partition

As introduced above, HEVC employs a few types of units, e.g., CTU, CU, and PU, to construct the coding structures. The CTU is the basic processing unit and may be split into CUs in quadtree iteratively. Figure 2 illustrates the subdivision of a CTU of  $64 \times 64$  in size into CUs. It is observed from this figure that the root CTU is split into 15 CUs (leaves in quadtree) with four levels in total. Each CU decides a prediction mode according to the encoder, and then, it may continue to split into one or multiple PUs depending on the prediction mode.

As shown in Figure 2, the CU nodes marked with asterisks, e.g., 1\*, 2\*, 8\*, etc., represent that there exist subsequent PU partitions. Besides, different prediction modes provide different optional partitioning modes, as illustrated in Figure 3. If the intra-picture mode is chosen by a CU, the PU size is the same as the CU size, or the CU is split into four PU quadrants when the CU size is the smallest (refer to Modes (a) and (d) in Figure 3, respectively). For example in Figure 2a, the CU nodes 0, 3, 4, 5, etc., are split into one PU in Mode (a), or they are regarded as not split. On the other hand, when the prediction mode is signaled as inter-picture, the CU can split into one, two, or four PUs in all eight modes. For instance, the CU nodes 1\*, 8\*, and 14\* in Figure 2a are split in Mode (c), and nodes 2\* and 12\* are split into asymmetric PUs in Mode (f).



**Figure 2.** Subdivision of a CTU (Coding Tree Unit) into CUs (Coding Units) (with subsequent Prediction Unit (PU) partitioning marked with an asterisk (\*)). Solid lines indicate CU boundaries, and dotted lines indicate PU boundaries. The picture is one of the P frames from the HEVC-encoded “Akiyo” video by HM12.0 [27]. (a) CTU with its partitioning. (b) Corresponding quadtree.

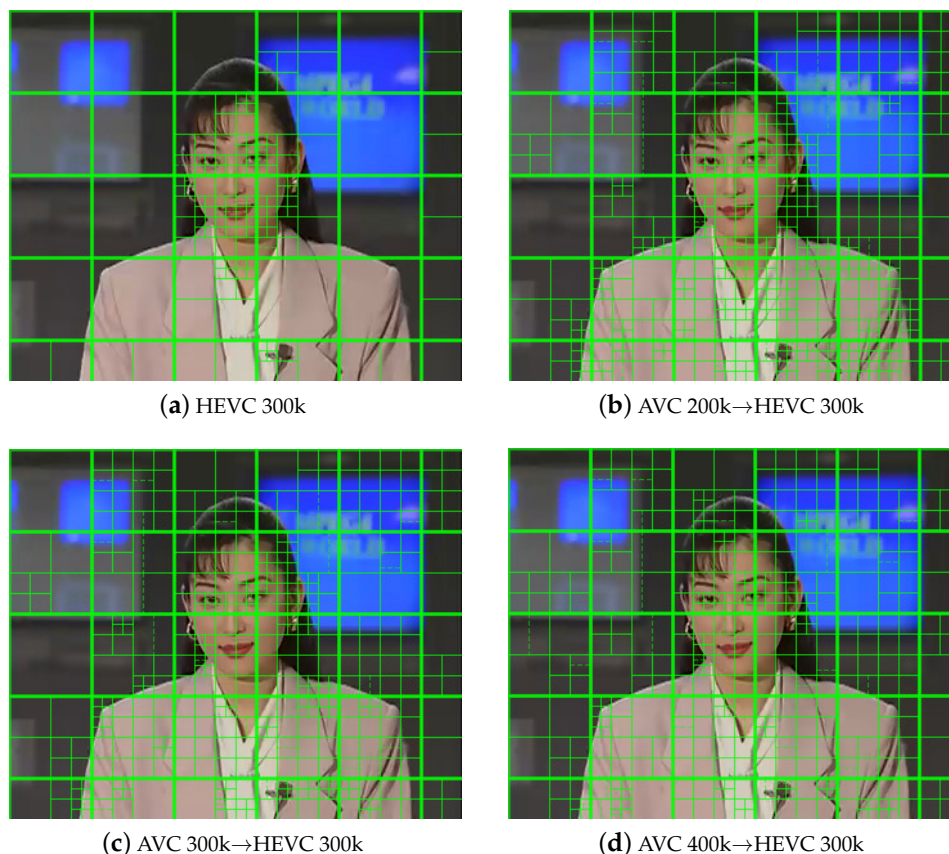


**Figure 3.** Modes for splitting a CU into PUs, subject to certain size constraints. For intra-picture-predicted CUs, only (a) and (d) are supported.

### 2.3. Footprints of HEVC

The quadtree-based CU block partitioning structure is designed to adapt to various texture characteristics of images in different encoding cases [28]. In other words, the quadtree-based coding structure can select appropriate CU partitions for regions of different texture complexity. Figure 4a

illustrates the CU and PU partitions of one frame from the “Akiyo” video encoded by HEVC. It is observed from the figure that the more complex the texture, the smaller the CU size of the partition, e.g., the face area of “Akiyo”. In this way, the complex region can be encoded with finer coding units and prediction units, thus effectively improving the visual performance of videos.



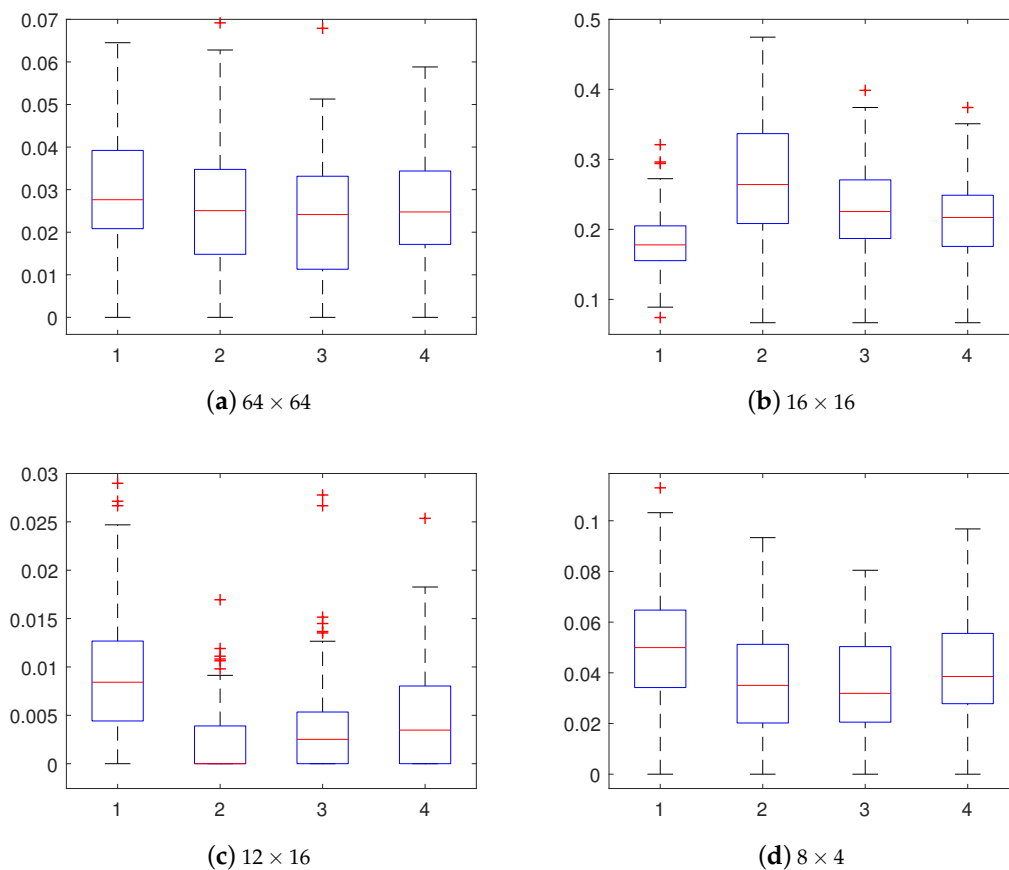
**Figure 4.** PU partitions for “Akiyo” videos of 300k (bps). Solid lines indicate CU boundaries, and dotted lines indicate PU boundaries. (a) HEVC-encoded video. (b–d) Re-encoded HEVC videos from various AVC coding.

However, according to extensive experiments and analysis, it is found that the PU partition of an HEVC video is easily impacted by its compression history. In other words, if a video is re-encoded to HEVC from AVC format, its PU partition is different from that of an original HEVC video. Figure 4b–d illustrates the PU partitions of re-encoded HEVC videos from AVC formats with different bitrates. Compared to Figure 4a, the PU partitions of transcoded HEVC videos (Figure 4b–d) change significantly, even though these videos are of the same content. It is also observed that there exist more finer-grained PUs, i.e., smaller-sized blocks, in transcoded HEVC videos. Besides, the amount of coarse-grained PUs is much less accordingly in transcoded HEVC videos. Therefore, this property can be exploited as the footprints of HEVC videos.

To further analyze the footprints, the frequencies of PUs are calculated for these four types of videos (Figure 4a–d). Figure 5 shows the box diagrams for some PU sizes with the horizontal axis corresponding to these four types of videos. For comparison purposes, four PU sizes are illustrated in this figure, i.e., a large size of  $64 \times 64$ , a small size of  $16 \times 16$ , a vertical splitting unit of  $12 \times 16$ , and a horizontal partitioning unit of  $8 \times 4$ . From Figure 5, it is observed that the distributions of PUs are distinguishable between original HEVC videos and transcoded ones. Taking Figure 5b for example, it shows the frequency of PUs of  $16 \times 16$  in size for four videos (Video #1–4). As can be seen, the frequencies of PUs differ between the original HEVC videos (Video #1) and the transcoded



ones (Video #2–4). Namely, the median value of Video #1 is below 0.2, while the medians of Video #2–4 are all above 0.2. Besides, the amount of PUs of  $16 \times 16$  in size in Video #2–4 is more than that in Video #1, signifying that transcoded HEVC videos tend to split into smaller PUs, which agrees with the observations in Figure 4. However, there are more types of PUs in the statistical analysis of Figure 5, compared to Figure 4, which is based on one single picture. For instance, it can be seen from Figure 5c,d that there are more PUs of  $12 \times 16$  and  $8 \times 4$  in size in the original HEVC video (#1) than that in the transcoded videos (#2–4). Based on the above analysis of PUs of various sizes, it can be concluded that the frequencies of PUs are distinguishable between original HEVC videos and transcoded ones.



**Figure 5.** Box plots of the frequencies of PUs for four types of videos, i.e., originally-encoded HEVC videos, re-encoded HEVC videos from AVC formats with bitrates of 200, 300, and 400 Kbps, denoted as 1, 2, 3, and 4 in horizontal axis, respectively. Four different PU sizes are considered, i.e., (a)  $64 \times 64$ ; (b)  $16 \times 16$ ; (c)  $12 \times 16$ ; (d)  $8 \times 4$ .

#### 2.4. Footprint Analysis

The CU and PU partition scheme in the HEVC coding standard is determined in the sense of the minimum Rate-Distortion (RD) costs [29]. RD cost for each CU size is defined as follows:

$$J_x = B_x + \lambda_x \cdot SSE \quad (1)$$

$$SSE = \sum_{i,j} (Block_x(i,j) - Block_{ref}(i,j))^2 \quad (2)$$

where  $B_x$  specifies the bit cost to be considered and  $\lambda_x$  is the Lagrange multiplier. For a given bitrate, the RD cost is influenced by the SSE, which is determined by the difference between the current encoding CU  $Block_x$  and its reference block  $Block_{ref}$ .

The reason for the difference in PU frequency between the original and transcoded videos depends on the distinct video encoding chains. The original HEVC video is directly encoded to HEVC format from a sequence of raw uncompressed images, while the transcoded HEVC video is generated by re-encoding the decoded content from AVC coding. Therefore, the distortion of the current encoding image is much more obvious in the transcoded video than that in the original video. This leads to a larger *SSE* and a larger RD cost in the transcoded video compared to the counterpart in the original HEVC video. As a result, the PU partition of the transcoded video differs from that of the original video since the transcoded video has to try other partition patterns to achieve the least RD cost. Therefore, it can be observed that the PU frequencies are distinguishable between the original and the transcoded videos. Based on this property, we present the feature extraction method and propose the detection method in the next section.

### 3. Proposed Method

Based on the analysis of the characteristics of PU partition in HEVC, we found that the frequencies of PUs of different sizes are distinguishable between original HEVC videos and transcoded ones. Therefore, we exploit the statistics of PUs for feature extraction and algorithm design. In this section, we firstly depict the feature extraction method and then present the proposed method.

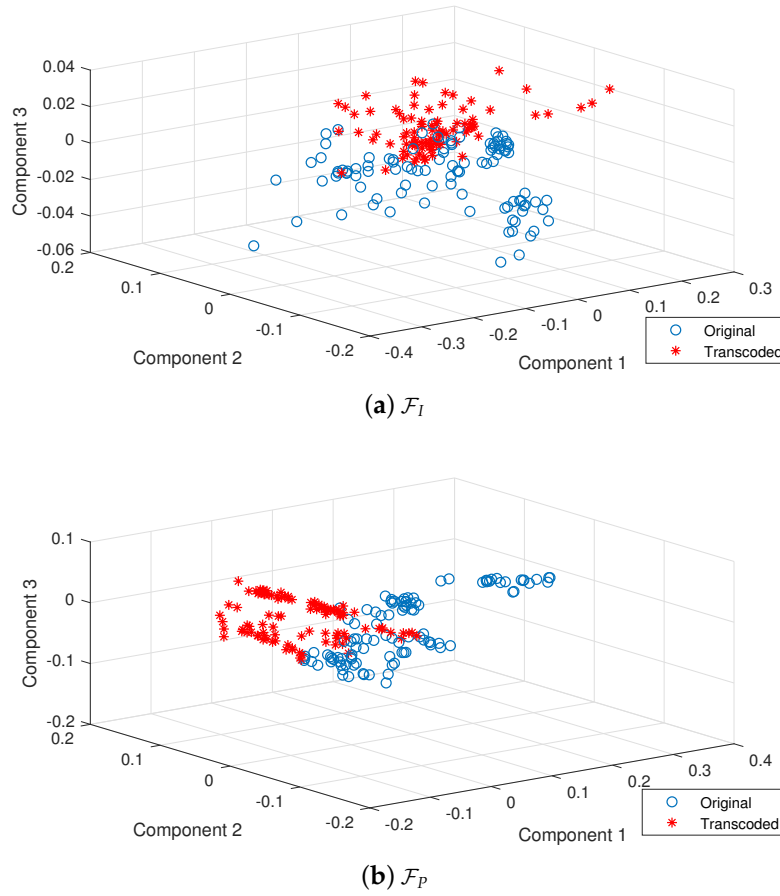
#### 3.1. Feature Extraction

As introduced above, the quad-tree-based CU partition strategy makes HEVC highly efficient to cope with video content of different texture complexities. Besides, according to the coding scheme, all CUs in I frames are coded using only intra-picture prediction, while some CUs of P frames can also be coded using inter-picture prediction. Considering quad-tree-based CU splitting and two alternative PU partition modes, there are five PU sizes, i.e.,  $64 \times 64$ ,  $32 \times 32$ ,  $16 \times 16$ ,  $8 \times 8$ , and  $4 \times 4$ , in I frames in total. Similarly, there are twenty-five different PU sizes in P frames giving eight PU partition modes that can be selected. Table 1 shows all twenty-five different PU sizes that can be adopted in P frames. The underlined sizes can be also employed in I frames.

**Table 1.** List of twenty-five PU sizes in HEVC. P frames can adopt any of them according to the encoder, while I frames can only employ the underlined ones.

<u><math>64 \times 64</math></u>	$64 \times 32$	$32 \times 64$	<u><math>32 \times 32</math></u>	$16 \times 64$
$48 \times 64$	$64 \times 16$	$64 \times 48$	$32 \times 16$	$16 \times 32$
<u><math>16 \times 16</math></u>	$8 \times 32$	$24 \times 32$	$32 \times 8$	$32 \times 24$
$16 \times 8$	$8 \times 16$	<u><math>8 \times 8</math></u>	$4 \times 16$	$12 \times 16$
$16 \times 4$	$16 \times 12$	$8 \times 4$	$4 \times 8$	<u><math>4 \times 4</math></u>

According to the alternative PU sizes, the frequencies of PUs from I frames and P frames are calculated to generate two feature sets of 5D and 25D, respectively, denoted as  $\mathcal{F}_I$  and  $\mathcal{F}_P$ . For display purposes, the dimension of the feature sets is reduced by principal component analysis [30] to obtain the first three principal components. Figure 6 shows the representations of these components for feature sets  $\mathcal{F}_I$  and  $\mathcal{F}_P$  in the 3D axis. To further analyze the effectiveness of the proposed feature sets, samples of original HEVC videos and that of transcoded ones are both included in this figure for comparison purposes. From these figures, it is observed that the representations of original HEVC videos and transcoded ones can be separated to some extent in the principal component space, which means that the proposed two feature sets are effective at distinguishing original HEVC videos and transcoded ones. More validation experiments are provided in Section 4.



**Figure 6.** Representations of the first three principal components for original HEVC videos and transcoded ones.

### 3.2. Detection Method

Based on the analysis of  $\mathcal{F}_I$  and  $\mathcal{F}_P$ , we further combine these two feature sets as  $\mathcal{F}$ , which is the feature vector for subsequent classification. Figure 7 displays the whole diagram of the proposed method. The specific explanation for the algorithm is as follows:

- Step i. Assuming a given HEVC video  $\mathcal{V}$ , firstly decode it, and parse the bitstream for each frame  $v_i$  to obtain the data of PU partitions.
- Step ii. For each frame  $v_i$ , calculate the frequencies of PUs depending on the frame type denoted as  $p_{v_i}$ . In I frames, five sizes of PUs are taken into account, while in P frames, twenty-five sorts of PUs are calculated. Please note that we only count once for a PU split to prevent double counting. For example, we only count once for PU of  $16 \times 12$  in size when a root CU is split into two PUs of  $16 \times 12$  and  $16 \times 4$  in size.
- Step iii. Divide frames of  $\mathcal{V}$  into the set of I frames  $\mathcal{V}_I$  and the set of P frames  $\mathcal{V}_P$ . Then, compute the mean of  $p_{v_i}$  for  $\mathcal{V}_I$  and  $\mathcal{V}_P$ , respectively, denoted as  $\mathcal{F}_I$  and  $\mathcal{F}_P$ . These two feature sets are 5D and 25D, respectively.

$$\mathcal{F}_I = \frac{1}{|\mathcal{V}_I|} \sum_{i \in \mathcal{V}_I} p_i \quad (3)$$

$$\mathcal{F}_P = \frac{1}{|\mathcal{V}_P|} \sum_{i \in \mathcal{V}_P} p_i \quad (4)$$

- Step iv. Combine  $\mathcal{F}_I$  and  $\mathcal{F}_P$  as feature vector  $\mathcal{F}$  of 30D.



Step v. The 30D feature vector  $\mathcal{F}$  is fed to a pre-trained classifier (support vector machine) for identification.

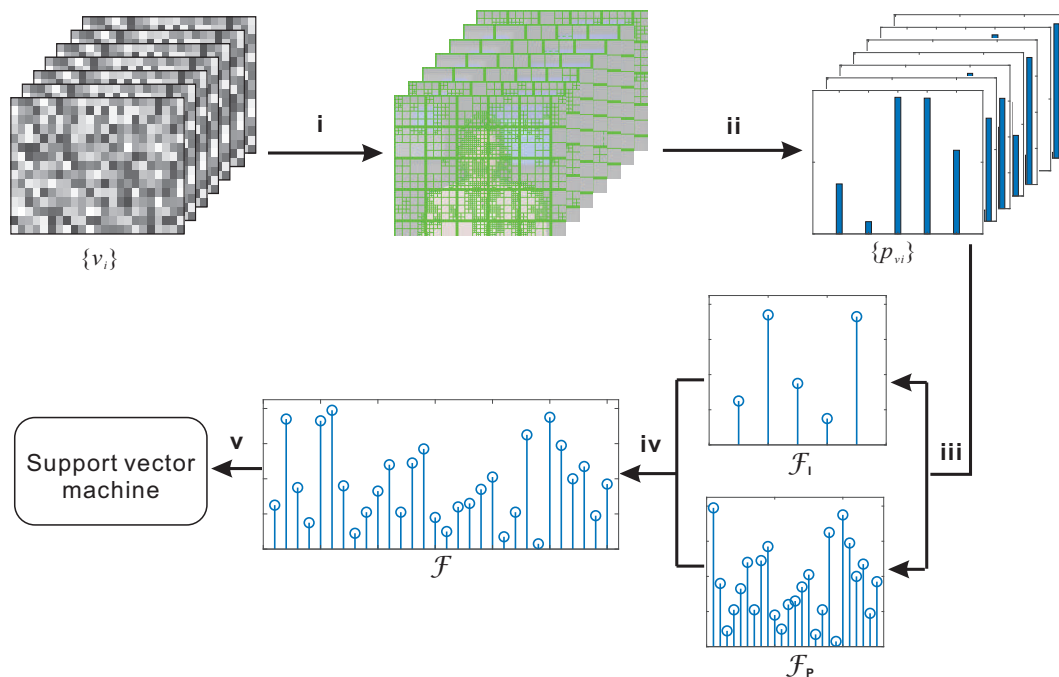


Figure 7. Flowchart of the proposed method.

With regard to the classifier, the support vector machine needs to be trained before the detection process. In the training procedure, original HEVC videos and transcoded ones are tagged as negative and positive instances, respectively. Then, operations in Steps i–iv are applied to each of these instances to generate feature sets. Finally, these feature sets of two classes are fed to a support vector machine to train a model for distinguishing between the original HEVC videos and the transcoded ones. After the training procedure, a subsequent query video can be processed by Steps i–v similarly, to decide whether it is original or transcoded with the help of the pre-trained SVM model.

#### 4. Experimental Results

In order to validate the proposed method, in this section, we present extensive experimental results conducted on a dataset with a wide range of video sequences of different resolutions and encoding parameters.

##### 4.1. Experiment Setup

The dataset used in the experiments is composed of originally-encoded HEVC videos and transcoded HEVC videos as negative and positive instances, respectively. To generate these instances, all videos are encoded from uncompressed YUV sequences downloaded from [1]. Due to the limited amount, these sequences are cut into short clips with 100 frames; thus, 105 CIF clips are obtained from 18 YUV sequences. With respect to coding tools, HM 12.0 [27] is employed with low-delay coding mode for HEVC encoding/decoding, and JM 19.0 [31] is applied with the main profile for AVC encoding/decoding. For positive instances, each of these YUV clips is encoded to AVC format using JM, then decoded, and finally re-encoded using HM to HEVC format. For the negative instances, the YUV clips are directly encoded using HM to simulate original HEVC videos. In generating various encoded videos, four different bitrates are used in the experiments, i.e., 100 Kbps, 200 Kbps, 300 Kbps, and 400 Kbps. Considering that four bitrates are employed in the encoding process, 1680 ( $105 \times 4 \times 4$ )

transcoded videos are generated as positive instances, and 420 ( $105 \times 4$ ) original HEVC videos are obtained as negative instances.

In the feature extraction step, we modified GitlHEVCAnalyzer [32] to parse the CU and PU partition data in the decoding process of HEVC videos. The subsequent data analysis and model training was implemented in MATLAB R2017b. Besides, Support Vector Machine (SVM) was applied in the model training step with the Radial Basis Function (RBF) as the kernel function, and 10-fold cross-validation was employed in the experiments to assess the performance of the proposed model. With regard to the 10-fold cross-validation, all the videos in the dataset  $D$  were randomly divided into ten mutually exclusive equal-size subsets tagged as  $D_i$  for  $i \in [1, 10]$ . Then, the SVM was trained and tested ten times; each time,  $D_k$  was retained as the test set, and the union of the other nine sets  $\bigcup_{i \neq k} D_i$  served as the training set. The detection accuracies were computed by averaging the ten detection rates generated from ten times of training-testing. Extensive experimental results are presented in the following subsections.

#### 4.2. Results on Different Feature Sets

As discussed in Section 3, the features sets  $\mathcal{F}_I$  and  $\mathcal{F}_P$  are distinguishable between original and transcoded HEVC videos; thus, we combined these two features as the feature vector  $\mathcal{F}$  in the proposed method. To further analyze these feature sets, extensive comparative experiments were conducted on the dataset. Table 2 shows the experimental results in various cases with different AVC/HEVC bitrates. The bitrates in the column and row headings indicate the bitrates applied in the former AVC and the latter HEVC encoding, respectively.

**Table 2.** Detection accuracies (%) on three feature sets. The header row indicates the bitrates of re-encoded HEVC videos, while the header column shows the original bitrates of previous AVC encoding. Bold numbers represent the best results in each case.

Bitrate (AVC\HEVC)	Feature Set	100 Kbps	200 Kbps	300 Kbps	400 Kbps
100 Kbps	$\mathcal{F}_I$	91.54	94.77	93.15	91.43
	$\mathcal{F}_P$	90.08	97.86	98.39	98.99
	$\mathcal{F}$	<b>92.73</b>	<b>98.47</b>	<b>99.52</b>	<b>99.55</b>
200 Kbps	$\mathcal{F}_I$	89.03	94.92	93.44	92.40
	$\mathcal{F}_P$	76.32	96.36	98.55	99.86
	$\mathcal{F}$	<b>89.52</b>	<b>98.39</b>	<b>100.00</b>	<b>100.00</b>
300 Kbps	$\mathcal{F}_I$	<b>81.27</b>	94.64	94.67	90.61
	$\mathcal{F}_P$	68.51	92.08	97.47	98.90
	$\mathcal{F}$	<u>77.08</u>	<b>95.67</b>	<b>98.52</b>	<b>99.52</b>
400 Kbps	$\mathcal{F}_I$	<b>79.02</b>	94.76	95.25	91.10
	$\mathcal{F}_P$	65.84	85.95	95.51	98.24
	$\mathcal{F}$	<u>72.78</u>	<b>95.19</b>	<b>96.40</b>	<b>99.55</b>
<b>Average</b>	$\mathcal{F}_I$	<b>85.22</b>	94.77	94.13	91.39
	$\mathcal{F}_P$	75.19	93.06	97.48	99.00
	$\mathcal{F}$	83.03	<b>96.93</b>	<b>98.61</b>	<b>99.66</b>

From Table 2, it is observed that the proposed method (feature set  $\mathcal{F}$ ) was effective at detecting transcoded videos, no matter whether bitrates went up or down after transcoding. Besides, the results on the feature set  $\mathcal{F}$  were almost the best compared to those on the feature set  $\mathcal{F}_I$  or  $\mathcal{F}_P$ , except for a few cases, as underlined in the table. In some low-bitrate cases, e.g., 100 Kbps of HEVC, the average detection rate of  $\mathcal{F}_I$  was about 2% and 10% higher than that of  $\mathcal{F}$  and  $\mathcal{F}_P$ , respectively. The reason is that I frames can keep more information than P frames in video encoding due to the intra-picture prediction mode, so features from I frames are more robust to low-bitrate compression than P frames. Therefore, it can be concluded that  $\mathcal{F}_I$  played an important role in the proposed feature set  $\mathcal{F}$ . On the other hand, when the bitrates of HEVC encoding were higher, e.g., 200 Kbps, 300 Kbps, or 400 Kbps,

the detection accuracies on  $\mathcal{F}_P$  outperformed the results on  $\mathcal{F}_I$ , indicating that P frames contributed more in high-bitrate cases. Therefore, in the proposed method,  $\mathcal{F}_I$  and  $\mathcal{F}_P$  were combined as the feature vector under the overall consideration.

#### 4.3. Comparison with Existing Methods

In this part, contrast experiments are conducted for the purpose of comparative analysis with the existing methods. In the most recent literature, Yu et al. [19] proposed a method based on the histogram of the PU amount to detect transcoded AVC/HEVC videos. In their method, the PU partition data were calculated from the first P frame in each GOP. Then, all the histograms of the PU amount were averaged to generate a 25D feature vector, which was then fed to an SVM classifier. For comparison, Yu's method was applied to the same dataset with 1680 positive and 420 negative instances. The comparative results of the experiments are presented in Table 3, in which the differences in detection accuracies between the proposed method and Yu's are given.

From Table 3, it is observed that our method outperformed Yu's method in all cases with the average improvement being about 5%. The reason for the increase is that the feature set  $\mathcal{F}_P$  in our method was extracted from all P frames rather than only one in Yu's method, making our method more adaptable to different video contents. Besides, it can be seen from the averaged values in Table 3 that the lower the HEVC video bitrate, the more improved the performance. This is due to the employment of the feature set  $\mathcal{F}_I$  in our method, which is characterized as the lower the bitrate, the greater the effect. (Please note that the "effect" here means the relative effect compared to feature set  $\mathcal{F}_P$ .) These result also imply that our proposed combination of features from I frames and P frames played a vital role in the detection of transcoded HEVC videos.

**Table 3.** Comparative results (%) with Yu's method [19]. The numbers represent the differences of detection accuracies between our method and Yu's. Up-arrows indicate improvement of detection accuracies.

Bitrate (AVC\HEVC)	100 Kbps	200 Kbps	300 Kbps	400 Kbps
100 Kbps	3.74 ↑	2.33 ↑	4.81 ↑	2.38 ↑
200 Kbps	8.50 ↑	5.98 ↑	4.79 ↑	1.89 ↑
300 Kbps	4.62 ↑	5.66 ↑	5.67 ↑	3.81 ↑
400 Kbps	5.59 ↑	8.21 ↑	4.17 ↑	5.31 ↑
<b>Average</b>	<b>5.61 ↑</b>	<b>5.54 ↑</b>	<b>4.86 ↑</b>	<b>3.35 ↑</b>

#### 4.4. Robustness to Encoding Parameters

Video compression is usually impacted by many encoding parameters. In this part, therefore, we test the proposed method to analyze its robustness to different encoding cases. To this end, the experiments were conducted on a new dataset with different parameters applied in AVC and HEVC encodings. In the former AVC encoding, the B frame number was set as two, and the distance between I frames was set as 12, so the frame structure of AVC encoding was "IBBPBBPBBPBBBI". In the latter HEVC encoding, a different frame structure was employed with the B frame number set as zero and intra-frame distance set as 20. Similarly, four bitrates were tested in AVC/HEVC encodings, including 100 Kbps, 200 Kbps, 300 Kbps, and 400 Kbps. Therefore, 1680 positive and 420 negative instances were generated in the dataset.

Table 4 shows the detection accuracies in different cases. From this table, it can be seen that the overall results were still satisfying, with the average detection rate above 90%, but the detection accuracies decreased compared to the results of Table 2. The reason for the decline is that in cases of frame structure changing, frame types and prediction modes varied, as well. Consequently, this impacts the frequencies of PUs in the partition scheme and disturbs the proposed feature sets  $\mathcal{F}_I$  and  $\mathcal{F}_P$ . However, the influence was very limited, with the average decline of the detection rates as low as 2.8%, which means that the proposed method was robust to different encoding parameters.

**Table 4.** Detection accuracies (%) on compressed videos with different parameters. The average detection rate is 91.40%, and the average decrease is 2.8% compared to Table 2.

Bitrate (AVC\HEVC)	100 Kbps	200 Kbps	300 Kbps	400 Kbps
100 Kbps	90.58	97.07	97.19	97.66
200 Kbps	79.55	95.17	97.31	98.00
300 Kbps	75.79	93.63	95.00	97.14
400 Kbps	70.47	88.46	94.14	95.18
<b>Average</b>	<b>79.10</b>	<b>93.58</b>	<b>95.91</b>	<b>97.00</b>

#### 4.5. Robustness against Video Enhancement

In video processing history, video enhancement might be performed as well, together with video transcoding. Therefore, the performance of the proposed method was also evaluated by testing its robustness against video enhancement. To this end, the open-source video encoding tool FFmpeg was employed to process the extensive videos in the dataset automatically. The saturation enhancement was performed on the AVC decoded videos before HEVC re-encoding. For each frame in the video, the saturation was increased by 50% to make the image color purer and brighter. Table 5 presents the detection accuracies of the experiments conducted on the dataset with saturation enhancement. It can be seen from this table that the proposed method remained effective, with most of the average accuracies above 93%. The experimental results indicate that the proposed model was robust to video filtering operations such as saturation enhancement.

**Table 5.** Detection accuracies (%) on compressed videos with saturation enhancement. The average detection rate is 93.96%, and the average decrease is 1.15% compared to Table 2.

Bitrate (AVC\HEVC)	100 Kbps	200 Kbps	300 Kbps	400 Kbps
100 Kbps	94.71	98.14	99.52	99.55
200 Kbps	88.72	98.14	98.59	100
300 Kbps	77.65	94.78	96.69	96.23
400 Kbps	76.29	91.94	94.8	97.59
<b>Average</b>	<b>84.34</b>	<b>95.75</b>	<b>97.4</b>	<b>98.34</b>

#### 4.6. Results on High Definition Videos

HEVC has been designed to address essentially the issue of coding efficiency for High Definition (HD) videos [25]. Therefore, HD 720p (1280 × 720 progressive) videos were also taken into consideration in the experiments for the performance test. To this end, an HD video dataset was established by encoding uncompressed YUV sequences downloaded from [1]. Due to the limited YUV sequence sources, we cut these sequence into short clips with 100 frames, and 75 clips were obtained in total. Similarly, JM and HM were employed for AVC and HEVC encoding, respectively. Considering the large resolution of HD videos, new bitrates were selected in this part, i.e., 2 Mbps, 3 Mbps, 4 Mbps, and 5 Mbps. To evaluate the performance on HD videos, a new model was trained and tested based on the dataset of HD 720p videos, and 10-fold cross-validation was also employed to assess the average accuracy.

The experimental results on HD videos are shown in Table 6. It is observed from the table that the detection rates were all greater than 95%, indicating that the proposed method was very effective at detecting HD videos. Besides, it can be seen from the average detection rates that the accuracy went up along with the HEVC bitrate. The reason is that the more slight the HEVC compresses, the more information obtained in the feature set. Therefore, the results on high-bitrate videos were usually better. Moreover, it is observed that the average detection accuracy was as high as 98.17%, which is even better than that of CIF videos compared to Table 2. The reason is that HD videos contain more PUs than CIF videos due to the large resolution, making the feature sets more stable to different

contents. This also implies that the proposed method is very suitable for HD videos and thus practical in real applications.

**Table 6.** Detection accuracies (%) on HD 720p videos. The average detection rate is 98.17%.

Bitrate (AVC\HEVC)	2 Mbps	3 Mbps	4 Mbps	5 Mbps
2 Mbps	97.90	98.66	98.71	99.33
3 Mbps	97.99	98.62	98.71	98.75
4 Mbps	97.29	98.04	98.75	98.71
5 Mbps	95.23	97.50	97.90	98.62
<b>Average</b>	<b>97.10</b>	<b>98.21</b>	<b>98.52</b>	<b>98.85</b>

## 5. Conclusions

In this paper, we presented an effective method to detect transcoded HEVC videos from AVC format. The algorithm exploits the frequencies of prediction units in I frames and P frames. Based on this, a 5D and a 25D feature set are obtained from I frames and P frames, respectively, and then are combined as the 30D feature vector, which is finally fed to an SVM classifier. The validation experiments were conducted on a dataset consisting of a diversity of videos of CIF or HD 720p resolutions with four types of bitrates. Extensive experiments show that the proposed method was effective at detecting transcoded HEVC videos and robust to different encoding parameters.

In future work, more coding types will be taken into consideration in the previous encoding step to cover most of the popular coding standards, such as MPGE-2, MPEG-4, VP8/9, etc. Other encoding tools, e.g., x265, will also be employed in the experimental validation in future research. Besides, the method is expected to be further improved by introducing some new features, e.g., quantization coefficients prediction errors, etc., to enhance the adaptability of the method.

**Author Contributions:** S.B. conceived of the work, proposed and implemented the proposed methods, and wrote the manuscript. H.L. provided methodology suggestions and polished the manuscript. T.G. performed the experiments and analyzed the data. A.C.K. was the research advisor and provided methodology suggestions.

**Funding:** This work was supported by the National Natural Science Foundation of China (Nos. 61702199, 61672242, 61772209), the Science and Technology Planning Project of Guangdong Province (2016A050502050) and the China Scholarship Council.

**Acknowledgments:** The authors highly appreciate the authors of [19], L.Yu, Z. Zhang, et al., for their kind help in implementing their method. The authors would like to thank both the Editor and the reviewers for the invaluable comments and suggestions that helped greatly to improve this paper.

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

## References

1. Xiph.org Video Test Media [Derf's Collection]. Available online: <https://media.xiph.org/video/derf/> (accessed on 4 November 2018).
2. Milani, S.; Fontani, M.; Bestagini, P.; Barni, M.; Piva, A.; Tagliasacchi, M.; Tubaro, S. An overview on video forensics. In Proceedings of the APSIPA Transactions on Signal and Information Processing, Bucharest, Romania, 27–31 August 2012; Volume 1.
3. Bestagini, P.; Milani, S.; Tagliasacchi, M.; Tubaro, S. Codec and GOP Identification in Double Compressed Videos. *IEEE Trans. Image Process.* **2016**, *25*, 2298–2310. [[CrossRef](#)] [[PubMed](#)]
4. Bian, S.; Luo, W.; Huang, J. Exposing fake bit rate videos and estimating original bit rates. *IEEE Trans. Circuits Syst. Video Technol.* **2014**, *24*, 2144–2154. [[CrossRef](#)]
5. Stamm, M.; Wu, M.; Liu, K. Information forensics: An overview of the first decade. *IEEE Access* **2013**, *1*, 167–200. [[CrossRef](#)]
6. Singh, R.D.; Aggarwal, N. Video content authentication techniques: A comprehensive survey. *Multimed Syst.* **2018**, *24*, 211–240. [[CrossRef](#)]

7. Wang, W.; Farid, H. Exposing Digital Forgeries in Video by Detecting Double MPEG Compression. In Proceedings of the 8th Workshop on Multimedia and Security, Geneva, Switzerland, 26–27 September 2006; ACM: New York, NY, USA, 2006; pp. 37–47, doi:10.1145/1161366.1161375. [CrossRef]
8. Wang, W.; Farid, H. Exposing digital forgeries in video by detecting double quantization. In Proceedings of the MMandSec'09—11th ACM Multimedia Security Workshop, Princeton, NJ, USA, 7–8 September 2009; pp. 39–47.
9. Liao, D.; Yang, R.; Liu, H.; Li, J.; Huang, J. Double H.264/AVC compression detection using quantized nonzero AC coefficients. In *Media Watermarking, Security, and Forensics III*; International Society for Optics and Photonics: Bellingham, WA, USA, 2011; Volume 7880, p. 78800Q.
10. Luo, W.; Wu, M.; Huang, J. MPEG recompression detection based on block artifacts. In *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*; International Society for Optics and Photonics: Bellingham, WA, USA, 2008; Volume 6819, p. 68190X.
11. He, P.; Jiang, X.; Sun, T.; Wang, S. Detection of double compression in MPEG-4 videos based on block artifact measurement. *Neurocomputing* **2017**, *228*, 84–96. [CrossRef]
12. Chen, W.; Shi, Y. Detection of double MPEG compression based on first digit statistics. In *Digital Watermarking. IWDW 2008*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2009; Volume 5450, pp. 16–30.
13. Sun, T.; Wang, W.; Jiang, X. Exposing video forgeries by detecting MPEG double compression. In Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan, 25–30 March 2012; pp. 1389–1392.
14. Jiang, X.; Wang, W.; Sun, T.; Shi, Y.; Wang, S. Detection of double compression in MPEG-4 videos based on Markov statistics. *IEEE Signal Process. Lett.* **2013**, *20*, 447–450. [CrossRef]
15. Jia, R.S.; Li, Z.H.; Zhang, Z.Z.; Li, D.D. Double HEVC Compression Detection with the Same QPs Based on the PU Numbers. *ITM Web Conf.* **2016**, *7*, 02010. [CrossRef]
16. Xu, Q.; Sun, T.; Jiang, X.; Dong, Y. HEVC Double Compression Detection Based on SN-PUPM Feature. In Proceedings of the International Workshop on Digital Watermarking, Magdeburg, Germany, 23–25 August 2017; pp. 3–17.
17. Liang, X.; Li, Z.; Yang, Y.; Zhang, Z.; Zhang, Y. Detection of Double Compression for HEVC Videos with Fake Bitrate. *IEEE Access* **2018**, *6*, 53243–53253. [CrossRef]
18. Costanzo, A.; Barni, M. Detection of double AVC/HEVC encoding. In Proceedings of the 2016 24th European Signal Processing Conference (EUSIPCO), Budapest, Hungary, 28 August–2 September 2016; pp. 2245–2249.
19. Yu, L.F.; Zhang, Z.Z.; Yang, X.; Li, Z.H. P Frame PU Partitioning Mode Based H.264 to HEVC Video Transcoding Detection. *J. Appl. Sci.* **2018**, *36*, 278–286.
20. Apple Inc. Using HEIF or HEVC Media on Apple Devices. Available online: <https://support.apple.com/en-us/HT207022> (accessed on 4 November 2018).
21. Sammobile. Galaxy S9 and Galaxy S9+ Support Video Recording in HEVC Format. Available online: <https://www.sammobile.com/news/galaxy-s9-series-support-video-recording-in-hevc-format/> (accessed on 4 November 2018).
22. Cannon Inc. 4K UHD/50P 4:2:2 10-Bit Support with HEVC. Available online: <https://www.canon.co.uk/pro/video-cameras/xf705-4k-uhd-support/> (accessed on 4 November 2018).
23. International Telecommunication Union. H.265: High Efficiency Video Coding. Available online: <https://www.itu.int/rec/T-REC-H.265> (accessed on 4 November 2018).
24. International Organization for Standardization. ISO/IEC 23008-2:2017. Information Technology—High Efficiency Coding and Media Delivery in Heterogeneous Environments—Part 2: High Efficiency Video Coding. Available online: <https://www.iso.org/standard/69668.html> (accessed on 4 November 2018).
25. Sullivan, G.J.; Ohm, J.; Han, W.; Wiegand, T. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1649–1668. [CrossRef]
26. Ohm, J.; Sullivan, G.J.; Schwarz, H.; Tan, T.K.; Wiegand, T. Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC). *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1669–1684. [CrossRef]
27. HEVC Test Model (HM). Available online: <https://hevc.hhi.fraunhofer.de/> (accessed on 4 November 2018).
28. Cho, S.; Kim, M. Fast CU Splitting and Pruning for Suboptimal CU Partitioning in HEVC Intra Coding. *IEEE Trans. Circuits Syst. Video Technol.* **2013**, *23*, 1555–1564. [CrossRef]



29. Ahn, S.; Lee, B.; Kim, M. A Novel Fast CU Encoding Scheme Based on Spatiotemporal Encoding Parameters for HEVC Inter Coding. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *25*, 422–435. [[CrossRef](#)]
30. Jolliffe, I. Principal Component Analysis. In *International Encyclopedia of Statistical Science*; Lovric, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 1094–1096.
31. JM Software. Available online: <http://iphome.hhi.de/suehring/> (accessed on 4 November 2018).
32. Li, H.; Chao, H. GitlHEVCAnalyzer. Available online: <https://github.com/lheric/GitlHEVCAnalyzer/> (accessed on 4 November 2018).



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).