DETECTION OF DOUBLE AND TRIPLE COMPRESSION IN VIDEOS FOR DIGITAL FORENSICS USING MACHINE LEARNING

by

Seba Youssef

A Thesis presented to the Faculty of the
American University of Sharjah
College of Engineering
In Partial Fulfillment
of the Requirements
for the Degree of

Master of Science in
Computer Engineering

Sharjah, United Arab Emirates

December 2020

Declaration of Authorship

I declare that this thesis is my own work and, to the best of my knowledge and belief, it does not contain material published or written by a third party, except where permission has been obtained and/or appropriately cited through full and accurate referencing.

Signature …Seba Youssef…………………………………………………………….

Date …23/12/2020……….…………………………………………………………

Approval Signatures

We, the undersigned, approve the Master's Thesis of Seba Youssef

Thesis Title: Detection of Double and Triple Compression in Videos for Digital Forensics Using Machine Learning

Date of Defense:
23/12/2020

Name, Title and Affiliation	Signature
Dr. Tamer Shanableh Professor, Department of Computer Science and Engineering Thesis Advisor	
Dr. Salam Dhou Assistant Professor, Department of Computer Science and Engineering Thesis Committee Member	
Dr. Mohamed Hassan Professor, Department of Electrical Engineering Thesis Committee Member	
Dr. Fadi Aloul Head, Department of Computer Science and Engineering	
Dr. Lotfi Romdhane Associate Dean for Graduate Affairs and Research College of Engineering	
Dr. Sirin Tekinay Dean, College of Engineering	
Dr. Mohamed El-Tarhuni Vice Provost for Graduate Studies Office of Graduate Studies	

Acknowledgement

I would like to thank my advisor, Dr. Tamer Shanableh, for providing the knowledge, guidance and support throughout my research stages. I’m highly grateful for his great assistance and encouragement throughout my thesis work. I would also like to thank the professors of the Computer Science and Engineering department for providing me with the knowledge and support throughout the master level courses. Finally, I would like to thank the Computer Science and Engineering department for providing me with the graduate assistantship and giving me the chance to develop my teaching skills while studying. I highly appreciate all their support and motivation.

Abstract

Digital video forensics is the process of analysing, examining and comparing a video for use in legal matters and court cases. In digital video forensics, the main aim is to detect and identify video forgery and manipulation to ensure a video's authenticity and reliability for use in court. This work focuses on passive forensics techniques, namely compression-based digital video forensics. When a video is edited by methods such as frame deletion, cropping, or duplication, the original encoded bitstream is first decoded, editing is applied and then the video is re-compressed before saving it. This means that by detecting re-compression in videos, we can interpret that the video has undergone some form of manipulation. The least number of recompressions a video can have is double compression, the first results from the device initially capturing the video which compresses it to store it in a suitable format and the second comes from the editing software or tool that re-compresses the video after it has been edited. Such editing can also be done multiple times leading to multiple compressions. Thus, finding out the compression history of a video becomes a very important mean for detecting any manipulation. Several techniques have been studied and investigated for the accurate classification of double and triple compression in videos based on machine learning and deep learning models with promising results being obtained. In this work, a number of experiments are conducted by using K-Nearest Neighbours (KNN), Random Forest (RF) or bi-directional Long Short-Term Memory (bi-LSTM) classifiers on a dataset of forged and unforged video sequences. In each of the experiments, performance is evaluated based on the classification accuracy and confusion matrix. Experiments are conducted on MPEG2 and HEVC coded videos using the same re-compression quantization parameter and the results of recompression detection are compared. Experiments are also conducted on HEVC coded videos with the same recompression bitrate and the results obtained are compared to existing solutions in literature. The experimental results revealed that both double compression and triple compression can be accurately detected using the proposed machine learning and deep learning solutions.

Keywords*: MPEG-2; HEVC; Recompression detection; Quantization parameter; Bitrate; Machine learning.*

Table of Contents

List of Tables

List of Figures

List of Abbreviations

B-frame	Bidirectional Frame
bi-LSTM	Bi-directional Long Short-term Memory
CU	Coding Unit
Eng	Energy of Prediction Residual
GoP	Group of Pictures
HEVC	High Efficiency Video Coding
I-frame	Intra Frame
KNN	K-Nearest Neighbours
LCU	Largest Coding Unit
MB	Macroblock
mbB	Backward MB
mbBi	Bidirectional MB
mbF	Forward MB
mbI	Intra MB
mbS	Skipped MB
ML	Machine Learning
MPEG	Moving Picture Experts Group
Mue	Mean
P-frame	Predicted Frame
PSNR	Peak Signal-to-Noise Ratio
PU	Prediction Unit
QB	Quantization Scale of B-frames
QCIF	Quarter Common Intermediate Format
QI	Quantization Scale of I-frames
QP	Quantization Scale of P-frames
RF	Random Forest
RNN	Recurrent Neural Network
Sig	Standard Deviation
TU	Transform Unit
VBR	Variable Bitrate

Chapter 1. Introduction

The field of digital video forensics refers to the scientific analysis, examination, evaluation and comparison of video to be used in legal matters. A video needs to be first validated before being used in legal cases to ensure its authenticity and suitability to court. Nowadays, there is a wide range of camera models available in the market such as smartphones, action camera, professional digital cameras and security system cameras. Each of them can use its own file format and video recording settings such as the resolution or number of frames per second. Such factors greatly impact how and what content is stored in the video. Because of the existence of digital forensics, such differences can be identified to avoid possible misinterpretation of what a video may appear to show to the public. These factors become very important in a court case as they can be the main evidence for proving someone’s guilt or innocence. It is also very evident that images and videos have become very significant tools in communication media. Due to the fact that professional knowledge is no longer required to edit or manipulate such data, a lot of research and interest has been put on digital forensics with the main focus on detecting image manipulation because of its widespread and relatively easier detection compared to videos.

However, in the past years, videos have been more widely used than ever in multiple domains such as security cameras in streets, trains, subways, malls, schools, companies and the like. Digital videos are also tremendously captured by cell phones which are then stored in compressed lossy formats with digital artifacts such as sampling and quantization. This means that fraud and manipulation of the compressed video through video editing, translation and transcoding can be easily applied without possible detection by the human eye. The same applies to the manipulation of digital video content present on social networking sites such as Facebook, Instagram, Twitter, LinkedIn and YouTube, making it very difficult to guarantee the trustworthiness and validness of such data without further examination. As a result, it becomes very important in court cases to assess the authenticity and quality of the video evidence being presented. From this, we can define video forensics as the logical examination and correlation of a video. Video forensics extracts the main features that distinguish between forged and original videos based on the type of forgery under examination.

Video forgery detection can be divided into two main classes, active forensics techniques and passive forensic techniques. In active forensics, approving data is used for the verification of the video, such as watermarks, fingerprinting and computerized watermarks. Passive forensics (aka blind forensics) does not rely on any validation information. It tries instead to extract features that differentiate between forged and unforced videos and possibly identify the location of forgery. Passive forgery detection techniques are further divided into two main categories; inter-frame forgery detection and intra-frame forgery detection. Inter-frame forgery detection involves the detection of forgery that occurs on an inter-frame level or in the temporal domain. Such forgeries can include frame insertion, frame deletion, frame duplication and double or multiple compressions. The detection techniques used for such types of forgeries are known as inter-frame forgery detection techniques. On the other hand, Intra-frame forgery detection involves the detection of forgery that occurs on each frame individually in the spatial domain. This includes copy-move of frames, splicing of frames, frame re-sampling or cropping and object-level manipulation. An input or unforged frame is first decoded, edited by either inter-frame or intra-frame forgery techniques and then re-encoded to be stored in its final compressed form. To identify such manipulation, the suitable forgery detection technique is used to extract the needed frames and features, which then uses a classification algorithm to distinguish between forged and original videos and possibly identify the exact type or location of forgery. This research focuses on passive forensic techniques due to the widespread of editing and manipulation of video content and thus there is much higher significance and need for such type of detection.

1.1. Motivation

Finding out the compression history of videos is one important way for identifying the authenticity and trustworthiness of a video. Over the past decade, several methods have been proposed by researchers in the forensics community for the detection of double compression in coded video sequences[1]. Multiple compressions occur when a video has undergone a series of compressions and decompressions. For instance, videos recorded on digital cameras are initially encoded by the camera itself and stored due to storage limitations, leading to the first compression in the series. The same video can then be decompressed, edited or manipulated and re-saved in the same format resulting in double compression. The number of compressions of a video sequence keeps on increasing every time the video undergoes any manipulation. This makes the efficient detection of double compression in videos extremely significant in the field of digital forensics due to the underlying manipulation they may have. In addition, due to the existence of social media platforms, even the authentic videos can have traces of double compression despite having no manipulation undergone. This is because a second compression is automatically applied whenever a video is uploaded on a social media platform such as Facebook, Messenger, Twitter, Instagram or WhatsApp. As a result, this makes focusing solely on double compression detection not enough and thus there is a need for building reliable triple or even multiple compression detection techniques. Multiple studies have focused on double compression detection in videos while, to our knowledge, none of them addressed the issue of triple compression detection. Therefore, the aim of this proposal is to address the issue of triple compression detection in MPEG and HEVC videos as well as to propose a new set of features that can be used to improve the existing accuracies for double compression detection in both coding techniques.

1.2. Problem Statement

The proposed solution aims to address the problem of forgery detection in MPEG2 and HEVC videos, based on the number of compressions they have undergone. This is done through the classification of double and triple compressions using machine learning and deep learning models. The detection of double and triple compression in coded videos can be viewed as a pre-step for forgery detection after which the existence and type of forgery can be identified. The solution aims to identify the effect of the more challenging case of having the same recompression quantization parameter on the detection of recompression in MPEG2 and HEVC coded videos. It also aims to identify the effect of having the same recompression bitrate on the detection of recompression in HEVC videos. A number of distinctive features are extracted from the original video bit stream, with single compression, and the re-compressed frames. In double compression detection, we classify the sequences as unforged which are the original sequences that have undergone single compression (class 1) and forged which are the sequences that have undergone two compressions (class 2). Whereas in triple compression detection, three classes are used for classification with class 1 representing the unforged samples, class 2 representing the forged samples with two compressions and class 3 representing the forged samples with 3 compressions. The features used for classification of MPEG2 coded videos are based on the prediction residuals of inter-coded macroblocks, the percentage of intra-coded macroblocks, and the estimated reconstruction quality or PSNR. The same features are extracted in HEVC videos but on a coding unit or CU level from which further statistics are obtained and used for classification. Machine and deep learning techniques are then used to report the classification accuracy, true positive and false negative rates of each of the conducted experiments.

Chapter 2. Background and Literature Review

This section discusses the systems presented in literature in the field of video compression based forensics as well as some background knowledge of the coding schemes, machine learning algorithms and the statistical approaches used throughout the experiments. Sections 4.1 and 4.2 present the systems proposed in literature related to compression detection in MPEG and HEVC videos. Sections 4.3 and 4.4 then briefly explain the two coding schemes used in the conducted experiments. Finally, in section 4.5, the three classifiers used, Random Forest (RF), K-Nearest Neighbors (KNN) and bi-directional Long Short-Term Memory (bi-LSTM) network are being discussed.

2.1. Existing Solutions for MPEG2

This section summarizes the systems presented in literature in the field of compression-based video forensics along with the feature sets used in each. Triple compression detection in videos is considered a relatively new area of research, but the problem of the detection of double MPEG-X (where X is 1,2 or 4) compression has been previously discussed in [2-10] and double HEVC compression detection has been discussed in [11-14]. With the focus on compression detection in MPEG-1 and MPEG-2 videos, multiple systems have been proposed to solve such issue. One of the most significant systems tackling double compression detection in MPEG videos was initially proposed in [8]. In their work, two techniques were used to identify the static and temporal artifacts that result from double MPEG compression after video tampering has occurred. Two main features were used, the histograms of quantized I-frames and the motion estimation errors of the predicted (P) and bidirectional (B) frames. In [7], Sun et al. detected double compression in MPEG videos by relying on the fact that compression will introduce disturbance in the DCT coefficients, where the first digit distribution of quantized AC coefficients must have the same characteristics if no double compression is introduced and different characteristics with double compression. Features based on the first digit AC coefficient of I-frames with varying bitrates are being considered. In [12] , the authors focused on the detection of double compression in MPEG2 videos but with the same bit rate. In their system, they relied on the statistical difference between single MPEG-2 compressed videos and double MPEG-2 compressed videos due to the fact that the number of different DCT coefficients between I-frames of single compression and double compression is larger than the number of different coefficients between the corresponding I-frames of the MPEG double compression and triple compression. This is because when an I-frame is recompressed multiple times using the same quantization matrix, the number of different coefficients between the frames will decrease with each compression. As explained in their work, if D’ represents the number of different DCT coefficients between the I-frames of single and double compressions, and D’’ represents the number of different DCT coefficients between I-frames of double and triple compression, the decision can be made as follows:

```
D’ = no. of different coefficients between single and double
D’’ = no. of different coefficients between double and triple
{
        if D’’ > D’, double compression
        if D’’ <= D’, single compression
}
```

In [3], Aghamaleki and Behrad proposed a system for the detection of double compression and localization of tampering in MPEG-X videos based on the traces of quantization error in residual errors of P-frames. Residual errors of P-frames contain peak values during compression history. Therefore, when multiple compressions or frame insertion or deletion occur, multiple P-frames in a group of pictures (GoP) will contain peak residual error values, which is used as an indication of forgery or double compression. The proposed system detects double compression with different GoP lengths and structures with an average detection rate of 92.73% for videos with high compression rates.

The systems proposed in literature focused mainly on solving the problem of double compression detection with much room for improvement still available in terms of the feature sets and classifiers used as well as the accuracies obtained. In addition, to the best of our knowledge, no existing work addresses the problem of triple compression detection in MPEG videos. Thus, the proposed solution aims to use machine learning models to tackle the problem of forgery detection in MPEG videos based on the number of compressions they have undergone, whether double or triple. Features are to be extracted from P frames and the impact of fixing the quantization parameter on the detection accuracy is to be examined. The same experiments are conducted in HEVC coded videos with the same features being extracted and the results of both coding schemes have been compared.

2.2. Existing Solutions for HEVC

In this thesis, we also focus on the detection of recompression in HEVC coded videos with the same re-compression bitrate. This section summarizes some of the most recent and best performing systems presented in literature for double compression detection in HEVC coded videos. To the extent of our knowledge, only systems related to double compression detection of HEVC videos were found. The issue of triple compression detection in the field of video forgery is yet considered new and no relevant experiments could be found. A number of parameters have been commonly used for double compression detection in HEVC videos such as the PU (Prediction Unit) type, the DCT coefficient and the TU (Transform Unit) type [12, 14-16]. Statistical data is obtained from these parameters and used to construct the feature sets which are then combined with machine learning classifiers for recompression detection. Multiple experiments were also conducted for double compression detection in shifted GoP structures where I-frames have been relocated after recompression. For such problem, the average prediction residual sequence was commonly used as the main feature for detection [2, 17, 18].

To tackle the problem of double compression detection with the same recompression bitrate, Jiang et al. [19] proposed a very efficient solution that uses the same re-compression bitrate and makes use of the GOP-based PU type statistics extracted from each frame. In the proposed solution, the authors rely on the temporal variation patterns of the PU type across GOPs in single and double compression instead of using separate frames or the full video sequence as common with most other experiments in previous works. Three types of PUs are first extracted from the video sequences (Intra, Skipped, Predicted) and then the ratio of the Intra and Skipped PU types is calculated so that for each GOP unit, a PU sequence is generated. Then, for each GOP or PU sequence, the mean and standard deviation of these ratios are calculated to obtain the final PU-type statistic used for the re-compression detection. Four different bitrates have been used for the double re-compression detection using the GOP-based PU type statistics. When the same re-compression bitrate is used, the proposed method achieves accuracies ranging from 93% to 96% with 93% for the lowest bitrate of 800kbps and 96% for the highest tested bitrate of 1400kbps. As another solution, the authors [11] managed to achieve accuracies ranging from 92% to 93% when using the same re-compression bitrates. In this method, the authors proposed a solution that detects double compression in HEVC videos using the Sequence of Number of Prediction Units of its Prediction Mode (SN-PUPM). In this method, the authors first extracted the number of PUs having different prediction modes from each frame of the video sequences. Then from each adjacent three frames, the SN-PUPM feature is calculated after which it is smoothed using an averaging filter. An SVM classifier is used and the period analysis method is implemented on each video sequence for the detection of double compression. In this method, the same GOP size has been used for single and double compression. In an experiment that uses the same re-compression bitrate for the detection of double compression, Liang et al [13] achieved accuracies ranging from 85% to 87% for the same re-compression bitrates where they extracted a 25-D feature using the histogram of the partition modes of PUs named as HPP features. In this method, the aim was to implement a fake bitrate HEVC video detection system. The first PU in each GOP was used to extract PU information from which the 25-D HPP features are extracted. The HPP features of all GOPs of a video sequence are then averaged to obtain the final detection feature. Finally, an SVM classifier with a polynomial kernel was used with the 25-D HPP features used as an input to obtain the final detection accuracy.

In our proposed solution, the same experimental setup has been done to mimic the experiments conducted by [11, 13, 19] for the recompression detection of HEVC coded videos when using the same re-compression bitrate. The same video sequences used in [19] and the same preprocessing steps have been applied. In our solution, a newly proposed feature set along with a Random Forest classifier and a bi-LSTM deep learning network have been used with results that significantly outperform the existing methods.

2.3. The MPEG2 Compression Standard

The MPEG video standard is one of the most commonly used schemes for video compression with the aim of reducing the spatial and temporal redundancy within and between frames. This proposal focuses on the detection of double and multiple compressions in MPEG-2 Variable Bit Rate videos, which are also MPEG-1 backward compatible. A brief overview of the MPEG-2 video coding scheme is first given as explained in [20]. The statistical differences that recompression can cause in forged and unforged MPEG2 coded videos are then described in the next sections. MPEG-2 contains 3 different types of frames known as I, P and B frames or intra, predicted and bidirectional frames. Each of the three types leads to a different amount of compression. These frames are organized in sequences, with each sequence containing N frames known as a group of pictures (GoP). The I-frames in MPEG-2 are encoded using intra-coding, without reference to the other frames in the sequence. P-frames and B-frames are encoded using inter-coding with reference to other frames. P-frames use the previous I-frame or P-frame as a reference and B-frames use the previous and the next I-frame of P-frame as a reference for encoding. B-frames allow for the highest level of compression, with P-frames a bit lower and I-frames the lowest of all. Next is a brief explanation of the intra-frame and inter-frame coding schemes used in MPEG-2. The full MPEG-2 coding process is represented below in figure 1.

2.3.1. Intra-frame coding. The intra coding process in MPEG-2 is very similar to that of individual JPEG images with the three main steps of DCT, quantization and variable length coding. DCT is first applied to each 8x8 block of the image to remove spatial redundancy between pixels. This produces 8x8 DCT coefficients with the DC-coefficient being the most significant along with the low frequency DCT values which are finely quantized. The remaining values are the AC-coefficients, representing high frequency components, which undergo more coarse quantization. The quantization is done by dividing the coefficients by one of the standard JPEG quantization matrices. The compression rate can also be altered by defining a quantization parameter as a user input to the encoder when Variable Bit Rate (VBR) is used. Finally, variable length coding such as Huffman coding it applied to the resulting quantized DCT coefficients. To decode a frame, the inverse process is applied.

2.3.2. Inter-frame coding. In MPEG-2, inter-frame coding is applied to P and B frames where motion estimation and motion compensation is applied, making use of the temporal redundancy between frames. For each macroblock in the current image, a spatial search algorithm tries to find a corresponding matching macroblock in a reference frame(s), the previous frame in P-frames or previous and next frames in B-frames. If a good match is found, the macroblock then goes through inter-coding, otherwise intra-coding is performed. In the case of inter-coding when a best match macroblock is found, motion estimation is applied where a motion vector is calculated, pointing to the position of the matching block in the reference frame. Since the matching block is most likely not an exact match of the current block, motion compensation is applied to compute the differences between the two, producing a prediction error frame. The prediction error frame then undergoes intra-coding with the exception of having the quantization matrix as a flat matrix with only one constant value. The DCT coefficients will now be mostly zeros due to the prediction error having much less information. To decode an inter-coded frame, the encoded motion vectors, representing the spatial difference between two frames and the encoded prediction error frames, representing the content difference between the frames, are sent to the decoder where the inverse process happens to re-construct the frame.

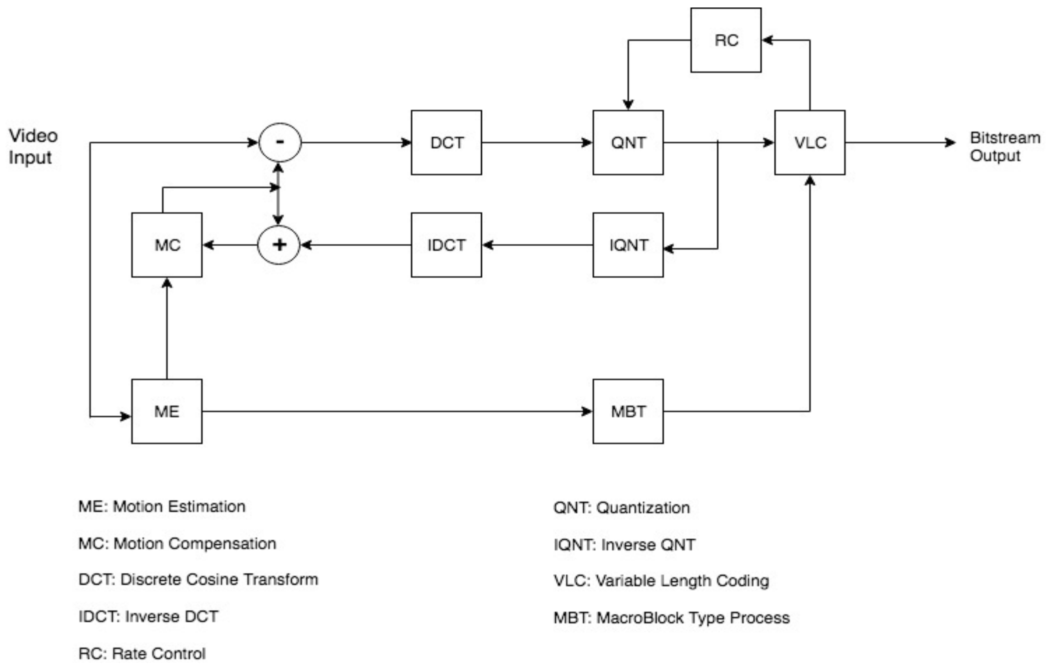

Figure 1: Overview of the MPEG-2 coding scheme.

2.4. The HEVC Compression Standard

HEVC was developed with the main aim of addressing the current need for high resolution videos with an improved coding efficiency [21]. In HEVC, inter and intra frame prediction are implemented along with transform coding, motion estimation and motion compensation. In this work, we rely on the fact that HEVC is a lossy coding technique so we attempt to examine the effect of HEVC re-compression on the coding parameter information from which we can differentiate between single, double and triple compression [15]. The HEVC coding standard replaces the known macroblock structure in H.264 with a coding tree unit (CTU) consisting of LCUs (Largest Coding

Units) as shown in Figure 2. These LCUs are divided into smaller sub-CUs in a recursive manner as shown in figure 3. The size L x L of a CTU can be chosen such that L=16, 32, or 64. These sub-CUs can be further divided into Prediction Units or PUs with sizes ranging from 4X4 to 32X32. The PUs represent the main block containing the prediction information which is also sent to the decoder. PUs are divided into three main types, skipped PUs (S-PU), inter-predicted PUs (P-PUs) and intra-predicted PUs (I-PUs) [22]. In this work, statistics related to PUs and CUs are being used. PUs are then divided into Transform Units or TUs which are used as an input to the Discrete Cosine Transform. HEVC supports four transform sizes for each N x N TU where N = 4,8,16 or 32. In intra prediction mode, HEVC defines 35 different intra prediction modes in which the mode with the lowest cost (Rate Distortion cost) is chosen. The intra-prediction modes are categorized into three types, DC Prediction Mode, Planar Prediction Mode and Angular Prediction Mode. DC prediction mode is used when predicting homogenous areas within a frame as it uses the average of the reference samples to predict a given sample. Planar prediction mode is used when needing to create smooth surfaces within an image. This method specifies an average value of two linear approximations using the four reference samples at all the corners of a given block. In the angular prediction mode, directional blocks of objects within an image are predicted by using the reference samples in a given direction and extrapolating them. In inter-prediction, either Discrete Cosine Transform (DCT) or Discrete Sine Transform (DST) is applied on TUs. The transform is applied on the residual coefficients which are calculated by finding the difference between the predicted block and the current block in the spatial domain after which quantization and entropy coding are applied. The full HEVC encoder process is shown below in Figure 4.

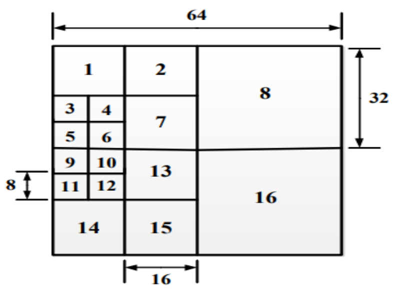

Figure 2: CU partitions [12].

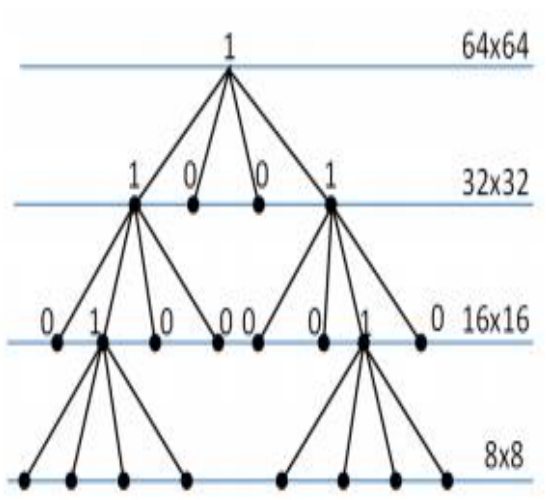

Figure 3: Example of CTU [21].

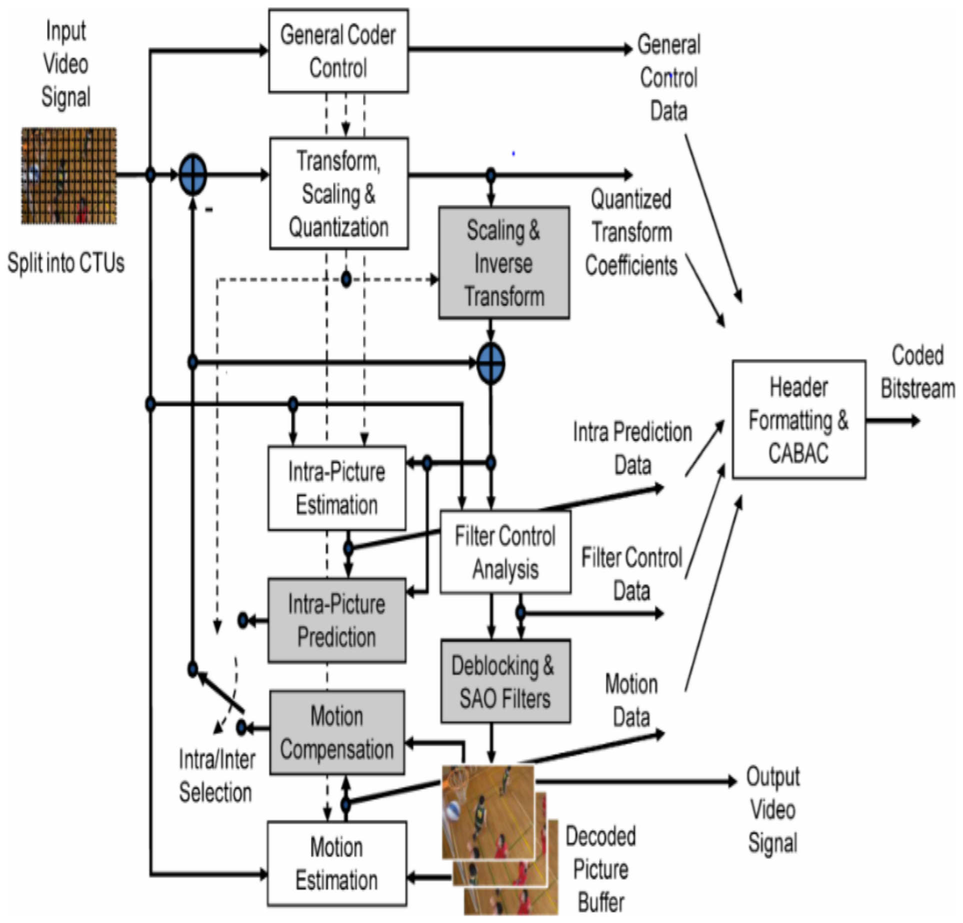

Figure 4: Typical HEVC video encoder [21].

2.5. Classifiers used in the proposed solution

To detect double and triple compressions, three machine learning techniques are used and compared in terms of the detection accuracy and true positive and false positive rates. The algorithms used are K-Nearest Neighbors (KNN), Random Forests (RFs) and bi-directional Long Short-Term Memory (bi-LSTM) networks. This section provides a brief explanation of the main functionalities of each of the three algorithms.

2.5.1. K-Nearest neighbors (KNN). As for KNN, the algorithm is used for classification based on feature similarity without any underlying knowledge or assumptions about the data distribution. The KNN algorithm is a very simple yet effective algorithm commonly used for classification [23]. An instance is classified based on the majority votes of its K nearest neighboring instances. The steps below show how the KNN works: If m is the number of samples in the training set and p is an unknown instance in the testing set,

1. The training samples are stored in an array *arr* of data points where each element in an array represents one of the instances.
2. For each of the training samples (*i=0 to m*), calculate the distance between *arr[i]* and that of p d(*arr[i],p*).
3. Store the classes of the instances with K smallest distances obtained from the previous step in a set S.
4. The majority label in S becomes the label of the test case p.
5. The type of distance used to find the k-nearest neighbors can be one of Euclidean, Manhattan or Minkowski. In the following experiments, Euclidian distance is being used as it is found to provide the highest results. Each of these distances is calculated as shown in Equations 1-3:

$$d = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{1}$$

$$d = \sum_{i=1}^{k}|(x_i - y_i)| \tag{2}$$

$$d = \left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q} \tag{3}$$

2.5.2. Random forest (RF). As for Random Forests (RFs), they are part of ensemble machine learning algorithms where a group of weak classifiers combine to form a strong classifier. The algorithm is based on the bagging approach where multiple decision trees are combined to create a new model with low variance in terms of classification and thus increase classification accuracy. The random forest approach is a combination consisting of a set of decision trees [24-27] used for classification. There are three types of nodes for each tree within the RF; root node, internal nodes and leaf or terminal nodes. The classification results of each separate tree are combined using the voting technique. One of the main advantages of RFs over decision trees is that individual decision trees tend to overfit. Thus, bagged decision trees or Random Forests combine the results of many decision trees to reduce overfitting and improve generalization. Each tree in the Random Forest is planted and grown as follows:

1. With N training samples, a sample of these N cases is selected at random with replacement and used as the training set for building 1 decision tree.
2. With M features, a number m<M is selected at random out of the M features and used for each tree being built. The value m is kept constant while growing the forest. The instances that were left out during the training phase of a certain tree is known as the out-of-bag data [28].
3. The above steps are repeated for building n trees (n is a user-defined value), each with a different subset of the training set N and a different subset of the M features. Each tree is then grown to the largest extent possible with no pruning.
4. New instances are predicted by combining the predictions of the n trees and classified using majority voting. In case of regression, the average predictions of trees is taken and used as the final output prediction.

2.5.3. Bi-directional long short-term memory (bi-LSTM). Long Short-Term Memory networks, also known as LSTMs, are a special version of the Recurrent Neural Network or RNN used for the prediction of sequential data and bi-LSTM is another version of the LSTM network. Recurrent neural networks differ from traditional feed-forward neural networks in the sense that they make use of past events to predict future ones. RNNs include additional complexity as they include loops that allow information to persist. An RNN can be viewed as one network having multiple copies with each copy passing a message to the following copy as shown in Figure 5 below. Thus, such networks can be very effective when used with sequential data where the output of one sample depends on that of previous samples. RNNs are very useful when it comes to language modeling, image captioning, translation, speech recognition, etc. The specific version known as LSTM network includes one input hidden and output layer with the hidden layer consisting of memory cells to retain previous data. The bi-LSTM is an improvement to the LSTM network where both previous and following data samples are used in the prediction of a current input. In other words, the bi-LSTM relies on past and future data to predict current data. Figure 6 shows a simple demonstration of the difference between an LSTM and a bi-LSTM network. This means that for every point in the input sequence, the network has full information about all the points preceding and following it. In our solution, we propose the use of the bi-LSTM network for recompression detection in HEVC coded videos using the fact that the features used to classify one frame will depend on that of the previous frames. This is because the use of features in previous frames will give a more clear understanding of the type of recompression in the present frame. To the extent of our knowledge, no existing solutions have used the bi-LSTM network for recompression detection in coded videos.

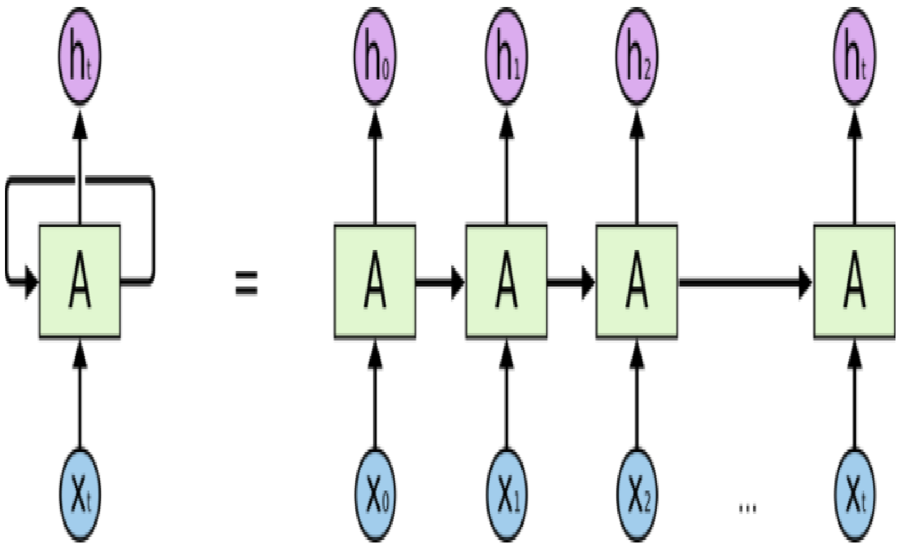

Figure 5: An unrolled recurrent neural network (RNN).

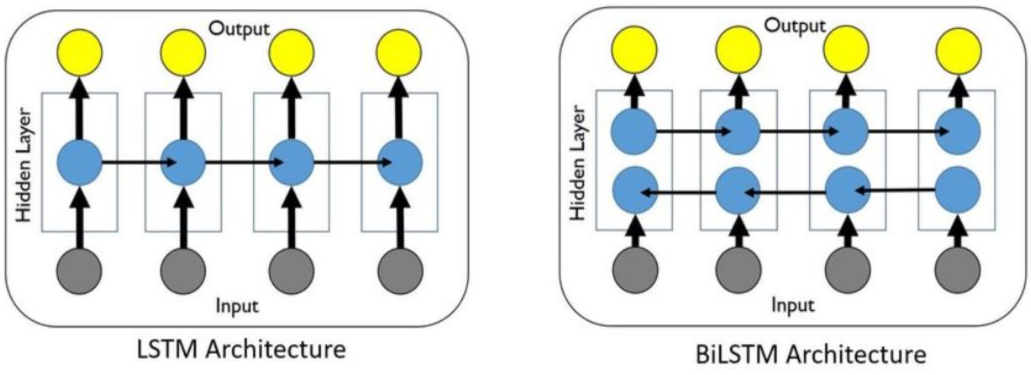

Figure 6: Difference between LSTM and bi-LSTM network [29].

Chapter 3. Proposed Solution

3.1. System Overview

3.1.1. MPEG2 & HEVC Recompression detection using same quantization parameter.

The existing video forgery detection solutions discussed earlier have a number of limitations. This is because most of the proposed solutions focused solely on the detection of double compression with the high majority relying on features extracted only from one frame type, disregarding the impact the forgery can have on multiple frame types. In addition, a focus was put on varying the bit rate of the re-compression, but none of the solutions have experimented the impact of varying the quantization parameter (QP), which is very common when any manipulation is done. Also, no existing work clearly addresses the issue of multiple compression detection in videos which is again very common especially with the spread of various social media platforms and automatic re-compression whenever any video is upload, making the detection of double compression not necessarily an indication of the existence of forgery.

Hence, in this work, the existing solutions reviewed are expanded in terms of the feature set used from different frame types, the quantization parameters and the number of compressions to be detected. A machine learning approach is used for the detection of double and triple compressions in MPEG-2 videos. The detection system is trained with a number of video sequences, consisting of both unaltered (i.e. singly compressed videos) and forged videos that have undergone double or multiple compressions. A number of features are extracted from the bitstreams of P-frames which are then used to train the model. The complete process of feature selection is further elaborated in the next section. The feature vectors are then normalized using z-scores due to the varying ranges they might have. A detection model is then generated which is capable of classifying a new unseen video sequence as forged or unforged as well as the number of compressions it has undergone. In this work, two machine learning techniques are used, K-Nearest Neighbours (KNN) and Random Forests (RF), from which we report the classification rate, true positive rate and false negative rate of each of the conducted experiments. Detection accuracy of the proposed model is then compared against the existing work. Experiments are conducted for MPEG2 coded videos that are re-compressed using the same quantization parameter since it is the most challenging case. The same exact experiments are then conducted using HEVC for the same dataset and same feature set and the results are compared. The proposed video forgery detection model system based on double and triple compression detection with same QP for both MPEG2 and HEVC is further illustrated in Figure 7.

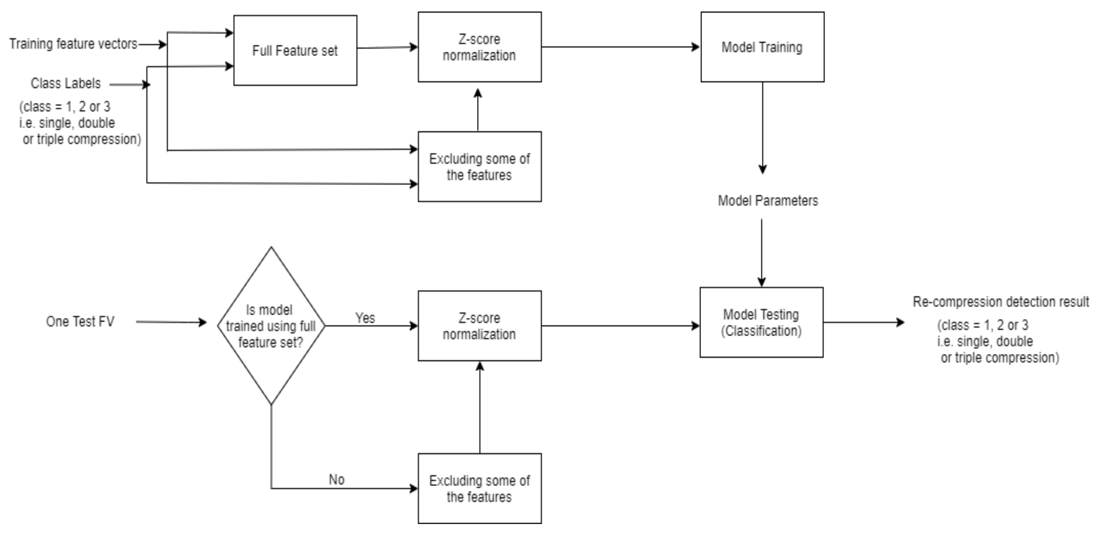

Figure 7: Block diagram of MPEG2 and HEVC re-compression detection with same QP.

3.1.2. HEVC Recompression detection using same bitrate. Multiple experiments are conducted to identify the re-compression detection accuracy for HEVC coded videos using the same re-compression bitrate. When using the same re-compression bitrate, it becomes more difficult to identify the traces of re-compression, making it more of a challenging task compared to when different bitrates are used and thus, more focus is put on such experiments. During the classification process, two different detection accuracies are obtained, one based on the extraction of frame-level features from the coded video sequences and the other based on the extraction of the Group of Pictures or GoP-level features. The results of both structures are obtained and compared with each other. Also, for each of the structures, classification is done using a Random Forest classifier as well as a deep learning bi-LSTM network. Figure 8 shows the experimental setup when using a RF classifier. Feature vectors of all the coded sequences are read on a frame-level or GoP-level, z-score normalization is applied and the feature vectors are split into training and testing data as discussed in the next section. Training feature vectors are downsampled and used as an input to the RF classifier.

After the training phase is complete, the testing dataset is used as an input to the model followed by sequence-level majority voting after which the sequence-level re-compression detection accuracies are obtained. As for the LSTM network, as shown in Figure 9, the same frame-level or GoP-level features are first read along with their class labels, which are then split into training and validation data. The bi-LSTM network is created, the training options are specified and the training dataset is used to train the network. The validation data is then used for testing and the final sequence-level accuracies are then obtained. Results from both classifiers, RF and bi-LSTM, and both feature sets, frame-level features and GoP-level features, for double compression detection with the same bitrate are reported in the Results section. The results have also been compared to existing work, demonstrating effectiveness of the proposed solution. Experiments are also conducted for triple compression detection with the same bitrate with promising results being obtained.

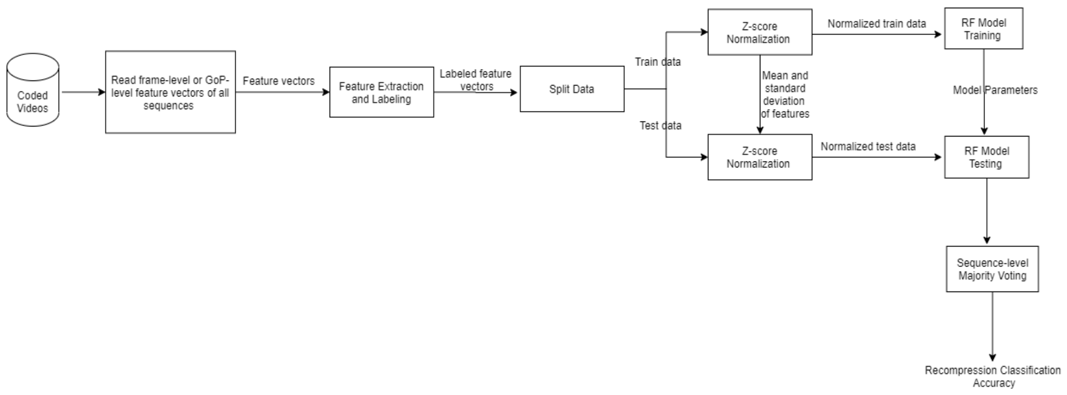

Figure 8: HEVC re-compression detection using RF classifier.

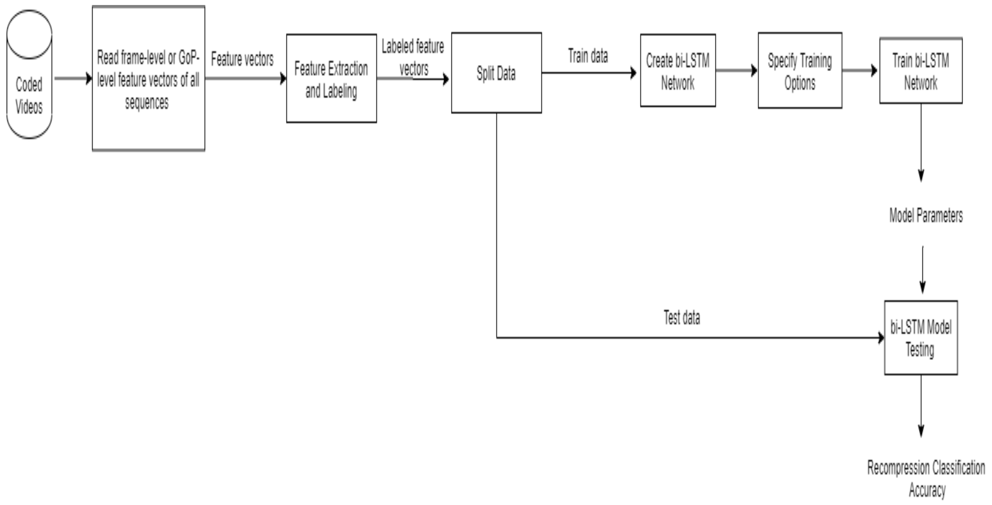

Figure 9: HEVC re-compression detection using bi-LSTM network.

3.2. Dataset

3.2.1. MPEG2 and HEVC Recompression detection using same quantization parameter. The MPEG2 codec is used to compress 36 YUV420 video sequences with QCIF or Quarter Common Intermediate Format resolution that are publicly available using Variable Bit Rate (VBR) coding. The group of picture (GoP) structure of all the MPEG2 coded sequences under examination has a value of N=12 (an I-frame every 12 frames) and M=3 (two B-frames between each pair of P-frames) represented by the structure IBBPBBPBBPBB….The video sequences used are Akiyo, Bowing, Bridge-Close, Bridge-Far, Carphone, City, Claire, Coastguard, Container, Crew, Deadline, Flower Gar- den, Football, Foreman, Galleon, Grandma, Hall, Harbour, Highway, Husky, Intros, Pamphlet, Mobile, Mother and Daughter, News, Paris, Salesman, Sign-Irene, Silent, Soccer, stefan, Students, Table, Tempete, Vtc1nw and WashDC. The same video sequences are also used for re-compression detection in HEVC coded videos. These video sequences have been chosen due to the different temporal activities they contain. Some of them contain very low activities while others contain medium to high levels of motion, providing a good representation of real-life video sequences. From these 36 videos, two sets of sequences are created, one representing the unaltered videos and the other representing the re-compressed or forged videos. Such sequences are available as raw YUV images and since original frames are not usually available in real life but are rather singly compressed and stored, we create the unforged dataset by first compressing the sequences using the MPEG-2 or HEVC coding scheme and storing their encoded bitstream. To create the forged dataset, we first decode the original encoded bitstreams, set the re-compression parameters and re-encode them using these new parameters, storing the re-encoded video bitstreams. For double compression detection, we re-compress the sequences only once while for triple compression detection, we recompress them three times, each compression depending on the previous one, before storing the final re-encoded bitstreams. The dataset consisting of forged and unforged video sequences is then used to train the classifiers and the detection rates are reported.

3.2.2. HEVC Recompression detection using same bitrate. In the experiments conducted for HEVC double and triple compression detection, 26 YUV420 sequences are used, thirteen of which are 1080p and thirteen are 720p.

Table 1 shows a detailed list of the YUV sequences used to generate the dataset, which is the same set of sequences used in [19]. All the sequences used are collected from the online video test media database[1]. To keep the spatial resolution the same for all the YUV sequences, the 1080p sequences are resized to 720p (1280x720) in the spatial domain in a lossless manner so that there are no traces of lossy compression. Also, to increase the sample size, each YUV sequence is divided into multiple non-overlapping sequences consisting of 100 frames each. As a result, a total of 127 shorter YUV sequences are obtained which are then used throughout the experiments. Four different bitrates are being used, selected from (800,1000,1200,1400) kbps. To obtain single compressed videos, the raw YUV sequences are compressed using one of the four bitrates. To obtain double compressed videos, the raw YUV sequences are first compressed using one of the bitrates then decompressed and re-encoded using the same bitrate. For triple compression, the same recompressions of double compression are conducted following by a third compression of the same bitrate. For a fair comparison, the x265 is selected as the HEVC codec, the Main Profile is applied in the coding process and B-frames are not considered to mimic the experiments setup used in [19]. All the obtained video sequences from the re-compressions are also divided into two groups for training and testing according to the splitting used in the previous work as shown in Table 2. This splitting structure ensures that no two YUV sequences originally belonging to the same sequence are included in both the training and testing datasets. As a result of the split, out of the 127 sequences, the training set will consist of 77 sequences and the testing set will consist of the remaining 50 sequences.

Table 1: YUV sequences used to generate dataset.

1080p YUV Sequences	**720p YUV Sequences**
blue_sky,crowd_run,pedestrian_area, riverbed,rush_field_cuts,rush_hour, snow_mnt,speed_bag,station2, sunflower,touchdown_pass,tractor, west_wind_easy	ducks_take_off,FourPeople,in_to_tree, Johnny,KristenAndSara,mobcal, old_town_cross,park_joy,parkrun, shields,stockholm,vidyo1,vidyo3

[1] [1] YUV sequences available at the online database: http://media.xiph.org/video/derf/.

Table 2: YUV sequences used in training and testing datasets.

Training Samples	**Testing Samples**
stockholm,KristenAndSara,Johnny, shields,vidyo3,ducks_take_off, sunflower,rush_hour,crowd_run, rush_field_cuts,blue_sky,speed_bag, touchdown_pass,west_wind_easy, pedestrian_area station2	in_to_tree,park_joy,old_town_cross,vidyo1, parkrun,mobcal,FourPeople,snow_mnt,tractor, riverbed.

Chapter 4. Proposed Feature Extraction and Classification

4.1. MPEG2 and HEVC Recompression Detection Using Same Quantization Parameter

This section discusses a new set of features to be used for the detection of double and triple compressions in MPEG2 and HEVC videos using the same re-compression quantization parameters. The features are extracted from the singly compressed decoded videos, representing the unaltered videos as well as the decoded videos after re-compression is applied, representing the forged videos. The MPEG2 feature set consists of 4 sequence-level features extracted from P-frames. A similar set of features were successfully used in detecting frame deletion in MPEG video [30]. In this work, we use this feature set for the detection of double compression and for the classification of triple compression. The same features used in MPEG2 were extracted in HEVC along with other HEVC-specific features. Table 3 shows the features extracted from MPEG2 coded videos and Table 4 shows the features extracted from HEVC videos. The proposed feature set consists of the mean and standard deviation of each of the extracted features. The equations in Table 5 and Table 6 have been used to compute the features extracted from MPEG2 and HEVC respectively.

Before conducting the experiments, it was observed that re-compression significantly affects the characteristics of P frames. It was also noticed that with each compression, the energy of prediction residuals, the percentage of intra MBs across frames and the quality of decoded frames noticeably change when re-compression is applied. Thus, such features play an important role in differentiating between forged and unaltered videos. It is also important to mention that in real-life, there is usually no access to the original raw video sequences, only the bitstreams after the first compression are available. Thus, to calculate the peak signal-to-noise ratio (PSNR) of the decoded videos, no reference quality assessment is to be used to estimate its value using the work reported in [31].

The results for HEVC have been obtained for three different feature sets; MPEG2-specific features, HEVC-specific features, and a full feature set in order to find the contribution of each of the feature sets on the results after which a comparison is done with MPEG2. The features used in each of the feature sets are subsets of the features shown in Table 3 and Table 4 such that the MPEG2-specific features are those presented in Table 3, the full feature-set are those presented in Table 4 and the HEVC-specific features are the difference between both. The results obtained for each of the feature sets have been reported and compared against those of MPEG2 in the Experimental Results section.

Table 3: An overview of the features computed in MPEG2 frames for same recompression QP.

Feature ID	Feature Description
1	*Prediction residual energy of non-Intra coded MBs.*
2	*Percentage of intra-coded MBs.*
3	*Estimated PSNR value.*
4	*Quantization parameter.*

Table 4: An overview of the features used in HEVC with same recompression QP.

Feature ID	Feature Description
1	*Number of sub-CU partitions*
2	*Ratio of MVD bits with reference to the total number of bits*
3	*Number of CU bits*
4	*Percentage of intra partitions in a CU*
5	*Percentage of skipped partitions in a CU*
6	*Percentage of inter (forward) partitions in a CU*
7	*Energy of prediction residual of CU*
8	*Estimated PSNR*

Table 5: Equations used in the computation of MPEG2 features with same QP.

Feature Name	Equation
Mean of prediction residual energy of non-intra coded MBs	$\mu_{\epsilon}={}^{1}/_{N}\sum_j\sum_i R_i(j)$ (4) *where i is the index of MB at the j*th *frame.* *N is the total number of predicted MBs in the video sequence for a P frame and* $R_i(j)$ *is the sum of absolute residual values for the i*th *MB at the j*th *frame.*
Standard deviation of prediction residual energy of non-intra coded MBs	$\sigma_{\epsilon}=\sqrt{E[(R_i(j)-\mu_{\epsilon})^2]}$ (5) *where E denotes the expected value.*
Mean percentage of intra-coded MBs	$\mu_{intra}={}^{1}/_{N}\sum_j I(j)$ (6) *where N is the total number of predicted P frames in the video sequence and I(j) is the percentage of intra coded MBs in the j*th *frame*
Standard deviation of percentage of intra-coded MBs	$\sigma_{intra}=\sqrt{E[(I(j)-\mu_{intra})^2]}$ (7) *where E denotes the expected value.*
Mean of estimated PSNR values	$\mu_{PSNR}={}^{1}/_{N}\sum_j P(j)$ (8) *where N is the total number of predicted P frames in the video sequence and P(j) is the estimated PSNR of the j*th *frame*
Standard deviation of estimated PSNR values	$\sigma_{psnr}=\sqrt{E\left[\left(P(j)-\mu_{psnr}\right)^2\right]}$ (9) *where E denotes the expected value.*
Mean of quantization parameter values	$\mu_{q}={}^{1}/_{N}\sum_j\sum_i Q_i(j)$ (10) *where N is the total number of MBs in the video sequence for a P frame and* $Q_i(j)$ *is the quantization parameter of the i*th *MB at the j*th *frame*
Standard deviation of quantization parameter values	$\sigma_{q}=\sqrt{E\left[\left(Q_i(j)-\mu_{q}\right)^2\right]}$ (11) *where E denotes the expected value.*

Table 6: Equations used in the computation of HEVC features with same QP.

Feature Name	Equation
Mean of *number of sub-CU partitions*	$\mu_{cuparts} = 1/_N \sum_j \sum_i A_i(j)$ (12) *where N is the total number of CUs per frame in the video sequence and $A_i(j)$ is the number of sub-CUs of the ith CU at the jth frame*
Standard deviation of *number of sub-CU partitions*	$\sigma_{cuparts} = \sqrt{E\left[\left(A_i(j) - \mu_{cuparts}\right)^2\right]}$ (13) *where E denotes the expected value.*
Mean of ratio of MVD bits	$\mu_{mvd} = 1/_N \sum_j \sum_i M_i(j)$ (14) *where N is the total number of CUs per frame in the video sequence and $M_i(j)$ is the ratio of MVD bits of the ith CU at the jth frame*
Standard deviation of ratio of MVD bits	$\sigma_{mvd} = \sqrt{E[(M_i(j) - \mu_{mvd})^2]}$ (15) *where E denotes the expected value.*
Mean of number of CU bits	$\mu_{cubits} = 1/_N \sum_j \sum_i B_i(j)$ (16) *where N is the total number of CUs per frame in the video sequence and $B_i(j)$ is the number of CU bits of the ith CU at the jth frame*
Standard deviation number of CU bits	$\sigma_{cubits} = \sqrt{E[(B_i(j) - \mu_{cubits})^2]}$ (17) *where E denotes the expected value.*
Mean of percentage of intra partitions in a CU	$\mu_{cuintra} = 1/_N \sum_j I(j)$ (18) *where N is the total number frames in the video sequence* *I(j) is the percentage of intra CUs in the jth frame*
Standard deviation of percentage of intra partitions in a CU	$\sigma_{cuintra} = \sqrt{E[(I(j) - \mu_{cuintra})^2]}$ (19) *where E denotes the expected value.*
Mean of percentage of skipped partitions in a CU	$\mu_{cuskipped} = 1/_N \sum_j S(j)$ (20) *where N is the total number of frames in the video sequence* *S(j) is the percentage of skipped CUs in the jth frame*

Standard deviation of percentage of skipped partitions in a CU	$\sigma_{cuskipped} = \sqrt{E\left[\left(S(j) - \mu_{cuskipped}\right)^2\right]}$ (21) *where E denotes the expected value.*
Mean of percentage of inter partitions in a CU	$\mu_{cuinter} = {}^{1}/_{N} \sum_j P(j)$ (22) *where N is the total number of frames in the video sequence* *I(j) is the percentage of inter CUs in the* j^{th} *frame*
Standard deviation of percentage of inter partitions in a CU	$\sigma_{cuinter} = \sqrt{E[(P(j) - \mu_{cuinter})^2]}$ (23) *where E denotes the expected value.*
Mean of prediction residual energy of CU	$\mu_{\epsilon} = {}^{1}/_{N} \sum_j \sum_i R_i(j)$ (24) *where i is the index of CU at the* j^{th} *frame.* *N is the total number of inter CUs in the video sequence* $R_i(j)$ *is the sum of absolute residual values for the* i^{th} *CU at the* j^{th} *frame.*
Standard deviation of prediction residual energy of CU	$\sigma_{\epsilon} = \sqrt{E[(R_i(j) - \mu_{\epsilon})^2]}$ (25) *where E denotes the expected value.*
Mean of estimated PSNR	$\mu_{PSNR} = {}^{1}/_{N} \sum_j P(j)$ (26) *where N is the total number of frames in the video sequence* *P(j) is the estimated PSNR of the* j^{th} *frame*
Standard deviation of estimated PSNR	$\sigma_{psnr} = \sqrt{E\left[\left(P(j) - \mu_{psnr}\right)^2\right]}$ (27) *where E denotes the expected value.*

4.2. HEVC Recompression Detection Using Same Bitrate

Experiments are conducted for HEVC coded videos having the same re-compression bitrate using two different feature sets and two types of classifiers. The same experiments are conducted for double and triple compression detection. The first feature set includes features extracted from the video sequences on a frame-level. The second feature set includes features extracted on a group of picture or GoP-level. For both feature sets, the features extracted are based on the 8 coding unit or CU-level features shown in Table 4 in the previous section which are extracted for each 64X64 CU for each frame. The quantization parameter is also added to the feature set since using the same re-compression bitrate will result in a different quantization parameter (QP) value for each CU, making it of much use in re-compression classification. The CU-level features initially extracted are QP of each CU, the number of sub-CUs in each 64X64 CU, the ratio of motion vector difference bits, the number of CU bits, the percentage of intra, skipped and inter sub-CUs and the energy of the prediction residual for the CU. Table 6 shows the equations used to calculate the CU-level features. These features are first extracted from the encoder for single, double and triple compression and are then summarized to obtain features on a frame level. The features obtained on a frame level are further summarized to obtain features for each GoP. Figure 4 presents an overview of the feature extraction phase where YUV images are first input to the encoder, CU-level features are extracted and summarized into frame-level features which are used once for training and testing and then further split into Gop-level features for classification using the GoP-level feature set.

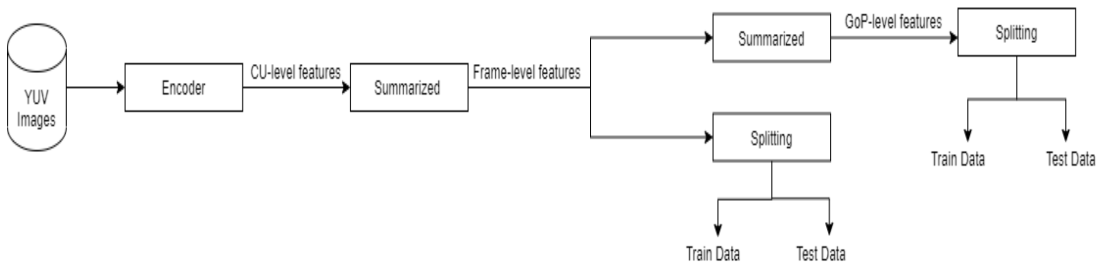

Figure 10: An overview of the feature extraction phase

4.2.1. Frame-level feature set. As for the frame-level feature set, the CU features corresponding to each frame are combined together and used to calculate the final set of frame-level features using the equations in Table 9. For each frame, there are 240 64X64 CUs for which features are extracted which means that each sequence will produce 2400 CU-level feature vectors. Since there are 127 sequences in the dataset, each re-compression will produce 304,800 feature vectors. The feature vectors are then summarized on a frame-level, meaning that for each 240 CUs of each frame in each sequence, the mean and standard deviation are calculated to obtain the value of the feature on a frame-level. The estimated PSNR for each frame is also calculated and added to the feature set. 100 feature vectors will be obtained for each frame and thus a total of 12700 (127 X 100) feature vectors will be obtained for each recompression.

Therefore, for double compression detection, a total of 25400 frame-level feature vectors are obtained and a total of 38100 feature vectors are obtained for triple compression detection.

Table 7: Equations used to calculate the frame-level features.

Feature Name	Equation
Mean of CU quantization parameter	$\mu_{q=1/N} \sum_i Q_i(j)$ (28) *where N is the total number of CUs per frame in the video sequence and $Q_i(j)$ is the quantization parameter of the ith CU in the current frame j*
Standard deviation of CU quantization parameter	$\sigma_q = \sqrt{E\left[\left(Q_i(j) - \mu_q\right)^2\right]}$ (29) *where E denotes the expected value.*
Mean of *number of sub-CU partitions*	$\mu_{cuparts=1/N} \sum_i A_i(j)$ (30) *where N is the total number of CUs per frame in the video sequence and $A_i(j)$ is the number of sub-CUs of the ith CU in the current frame j*
Standard deviation of *number of sub-CU partitions*	$\sigma_{cuparts} = \sqrt{E\left[\left(A_i(j) - \mu_{cuparts}\right)^2\right]}$ (31) *where E denotes the expected value.*
Mean of ratio of MVD bits	$\mu_{mvd=1/N} \sum_i M_i(j)$ (32) *where N is the total number of CUs per frame in the video sequence and $M_i(j)$ is the ratio of MVD bits of the ith CU in the current frame j*
Standard deviation of ratio of MVD bits	$\sigma_{mvd} = \sqrt{E[(M_i(j) - \mu_{mdv})^2]}$ (33) *where E denotes the expected value.*
Mean of number of CU bits	$\mu_{cubits=1/N} \sum_i B_i(j)$ (34) *where N is the total number of CUs per frame in the video sequence and $B_i(j)$ is the number of CU bits of the ith CU in the current frame j*
Standard deviation number of CU bits	$\sigma_{cubits} = \sqrt{E[(B_i(j) - \mu_{cubits})^2]}$ (35) *where E denotes the expected value.*

Mean of percentage of intra partitions in a CU	$\mu_{cuintra} = {}^{1}/_{N} \sum_i I_i(j)$ (36) *where N is the total number of CUs per frame in the video sequence and $I_i(j)$ is the percentage of intra partitions of the ith CU in the current frame j*
Standard deviation of percentage of intra partitions in a CU	$\sigma_{cuintra} = \sqrt{E[(I_i(j) - \mu_{cuintra})^2]}$ (37) *where E denotes the expected value.*
Mean of percentage of skipped partitions in a CU	$\mu_{cuskipped} = {}^{1}/_{N} \sum_i S_i(j)$ (38) *where N is the total number of CUs per frame in the video sequence and $S_i(j)$ is the percentage of skipped partitions of the ith CU in the current frame j*
Standard deviation of percentage of skipped partitions in a CU	$\sigma_{cuskipped} = \sqrt{E\left[\left(S_i(j) - \mu_{cuskipped}\right)^2\right]}$ (39) *where E denotes the expected value.*
Mean of percentage of inter partitions in a CU	$\mu_{cuinter} = {}^{1}/_{N} \sum_i P_i(j)$ (40) *where N is the total number of CUs per frame in the video sequence and $P_i(j)$ is the percentage of inter partitions of the ith CU in the current frame j*
Standard deviation of percentage of inter partitions in a CU	$\sigma_{cuinter} = \sqrt{E[(P_i(j) - \mu_{cuinter})^2]}$ (41) *where E denotes the expected value.*
Mean of prediction residual energy of CU	$\mu_{\epsilon} = {}^{1}/_{N} \sum_i R_i(j)$ (42) *where N is the total number of CUs per frame in the video sequence and $R_i(j)$ is the sum of absolute residual values of the ith CU in the current frame j*
Standard deviation of prediction residual energy of CU	$\sigma_{\epsilon} = \sqrt{E[(R_i(j) - \mu_{\epsilon})^2]}$ (43) *where E denotes the expected value.*
Estimated PSNR	$\mu_{PSNR} = P(j)$ (44) *where P(j) is the estimated PSNR of the current frame j*

4.2.2. Down sampling of frame-level feature vectors. The feature vectors obtained are divided into training and testing datasets according to the setup explained in the previous section. Since 77 of the sequences are allocated to training and 50 for testing, the training set shall contain 7700 feature vectors for each class and the testing set shall contain 5000 feature vectors for each class. The training dataset is down sampled to 5852 vectors to mimic the experimental setup used in [19] To ensure that none of the sequences are excluded all together from the training set while down sampling, for each sequence, 24 feature vectors are randomly deleted out of the 100 frames so 1848 feature vectors will be deleted resulting in the required 5852 training samples per class. Finally, when using the frame-level feature set for double compression detection, a total of 11704 (5852x2) training samples and 10000 (50x100x2) testing samples from both classes are used for the re-compression detection. For triple compression detection, a total of 17556 (5852x3) training samples and 15000 (50x100x3) testing samples from the three classes are used for the re-compression detection. The extracted frame-level features are then used to train and test a Random Forest and a bi-LSTM network and the results of each are reported and compared to existing work

4.2.3 GoP-level feature set. The Group of Pictures or GoP-level feature set is calculated from the frame-level feature set previously discussed, such that the features of frames belonging to each GoP are combined and used to calculate the final GoP-level features. It is important to mention that throughout the experiments, the GoP size was set to 15 (each GoP contains 15 frames) and thus each sequence of 100 frames shall contain 7 GoPs or 7 feature vectors. Table 10 shows the equations used to calculate them. For each class, a total of 889 (127x7) feature vectors are obtained which are divided to training and testing such that the training set consists of 539 (77x7) samples and the testing set consists of 350 (50x7) samples per class. No down sampling is applied on the training set when using GoP-level features since much fewer samples are used in which is already a more challenging case. Finally, when using the GoP-level feature set for double compression detection, a total of 1078 (539x2) training samples and 700 (350x2) testing samples from both classes are used for the re-compression detection. For triple compression detection, a total of 1617 (539x3) training samples and 1050 (350x3) testing samples from the three classes are used for the re-compression detection. The extracted GoP-level features are then used to train and test a Random Forest and a bi-LSTM network and the results of each are reported and compared to existing work.

Table 8: Equations used to calculate the GoP-level features.

Feature Name	Equation
Mean of CU quantization parameter	$\mu_{q} = {}^{1}/_{G} \sum_j ({}^{1}/_{N} \sum_i Q_i(j))$ (45) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$Q_i(j)$ is the quantization parameter of the ith CU in the current frame j and G is the number of frames per GoP*
Standard deviation of CU quantization parameter	$\sigma_q = \sqrt{E\left[\left(Q_i(j) - \mu_q\right)^2\right]}$ (46) *where E denotes the expected value.*
Mean of *number of sub-CU partitions*	$\mu_{cuparts} = {}^{1}/_{G} \sum_j ({}^{1}/_{N} \sum_i A_i(j))$ (47) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$A_i(j)$ is the number of sub-CUs of the ith CU in the current frame j and G is the number of frames per GoP*
Standard deviation of *number of sub-CU partitions*	$\sigma_{cuparts} = \sqrt{E\left[\left(A_i(j) - \mu_{cuparts}\right)^2\right]}$ (48) *where E denotes the expected value.*
Mean of ratio of MVD bits	$\mu_{mvd} = {}^{1}/_{G} \sum_j ({}^{1}/_{N} \sum_i M_i(j))$ (49) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$M_i(j)$ is the ratio of MVD bits of the ith CU in the current frame j and G is the number of frames per GoP*

Standard deviation of ratio of MVD bits	$\sigma_{mvd} = \sqrt{E[(M_i(j) - \mu_{mvd})^2]}$ (50) *where E denotes the expected value.*
Mean of number of CU bits	$\mu_{cubits} = {}^1/_G \sum_j ({}^1/_N \sum_i B_i(j))$ (51) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$B_i(j)$ is the number of CU bits of the ith CU in the current frame j and G is the number of frames per GoP*
Standard deviation number of CU bits	$\sigma_{cubits} = \sqrt{E[(B_i(j) - \mu_{cubits})^2]}$ (52) *where E denotes the expected value.*
Mean of percentage of intra partitions in a CU	$\mu_{cuintra} = {}^1/_G \sum_j ({}^1/_N \sum_i I_i(j))$ (53) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$I_i(j)$ is the percentage of intra partitions of the ith CU in frame j and G is the number of frames per GoP*
Standard deviation of percentage of intra partitions in a CU	$\sigma_{cuintra} = \sqrt{E[(I_i(j) - \mu_{cuintra})^2]}$ (54) *where E denotes the expected value.*
Mean of percentage of skipped partitions in a CU	$\mu_{cuskipped} = {}^1/_G \sum_j ({}^1/_N \sum_i S_i(j))$ (55) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$S_i(j)$ is the percentage of skipped partitions of the ith CU in frame j and G is the number of frames per GoP*
Standard deviation of percentage of skipped partitions in a CU	$\sigma_{cuskipped} = \sqrt{E\left[\left(S_i(j) - \mu_{cuskipped}\right)^2\right]}$ (56) *where E denotes the expected value.*
Mean of percentage of inter partitions in a CU	$\mu_{cuinter} = {}^1/_G \sum_j ({}^1/_N \sum_i P_i(j))$ (57) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)*

	$P_i(j)$ is the percentage of inter partitions of the ith CU in frame j and G is the number of frames per GoP
Standard deviation of percentage of inter partitions in a CU	$\sigma_{cuinter} = \sqrt{E[(P_i(j) - \mu_{cuinter})^2]}$ (58) *where E denotes the expected value.*
Mean of prediction residual energy of CU	$\mu_{\epsilon} = {}^1/_G \sum_j ({}^1/_N \sum_i R_i(j))$ (59) *where N is the total number of CUs per frame in the video sequence and j is the index of frame in the current GoP (each GoP contains 15 frames)* *$R_i(j)$ is the sum of absolute residual values of the ith CU in frame j and G is the number of frames per GoP*
Standard deviation of prediction residual energy of CU	$\sigma_{\epsilon} = \sqrt{E[(R_i(j) - \mu_{\epsilon})^2]}$ (60) *where E denotes the expected value.*
Mean of estimated PSNR	$\mu_{PSNR} = {}^1/_G \sum_j P(j))$ (61) *where j is the index of frame j in the current GoP (each GoP contains 15 frames)* *P(j) is the estimated PSNR of frame j* *G is the number of frames per GoP*
Standard deviation of estimated PSNR	$\sigma_{PSNR} = \sqrt{E[(P(j) - \mu_{PSNR})^2]}$ (62) *where E denotes the expected value.*

4.2.4. Sequence-level classification using random forest (RF). The RF classifier is trained and tested twice, once using the frame-level features and once using the GoP-level features. The obtained classification accuracies are on a frame-level and GoP-level but since we are interested in sequence-level accuracies, majority voting is then used. In sequence-level majority voting, each sequence is classified as single, double or triple compression based on the majority predicted label of its frames in case of frame-level features and the majority predicted label of its GoPs in case of GoP-level features. Finally, sequence-level accuracies are obtained using the two feature sets for double and triple compression for each of the 4 bitrates and these are the accuracies being reported throughout the experiments when using the Random Forest classifier.

4.2.5. Sequence-level classification using bi-direction short-term memory (bi-LSTM) network.

(bi-LSTM) network. The bidirectional Longest Short-Term Memory (bi-LSTM) deep learning network is used for the classification of re-compression in HEVC coded videos. Frame-level and GoP-level features are extracted from the video sequences and used as input to the bi-LSTM network, directly producing the sequence level accuracies for re-compression detection. The diagram in Figure 4 illustrates the network architecture. The Sequence Input layer involves inputting the video sequences to the network. A sequence folding layer is applied for the splitting of the video sequences into independent frames. Features are then extracted on a frame-level and GoP-level after which sequence unfolding is implemented to restore the sequence structure (feature vectors corresponding to a single sequence are combined together to represent one train/test sample). The bi-LSTM layers are then used to classify the resulting sequences producing sequence-level accuracies.

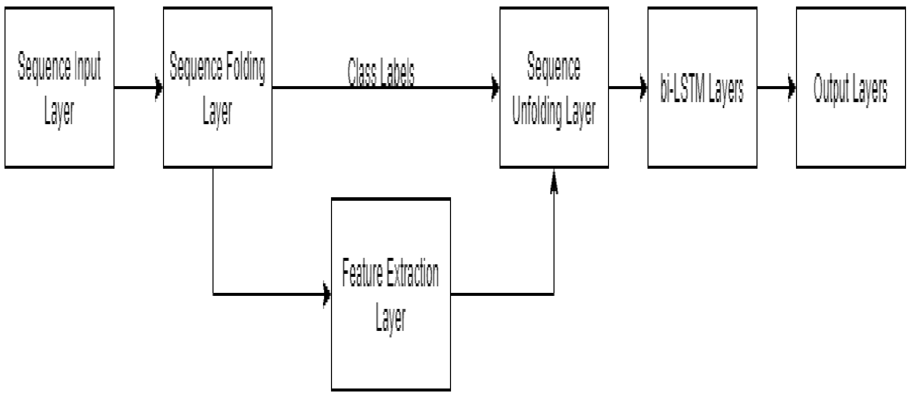

Figure 11: bi-LSTM Network Architecture.

Chapter 5. Experimental Results

This section discusses the results obtained from the proposed compression detection techniques. The experimental results of double and triple compression detection for MPEG2 and HEVC using a fixed quantization parameter value are discussed in Section 5.1. Results for double and triple compression detection in HEVC using the same re-compression bitrate are discussed in Sections 5.2 and 5.3 respectively. The double compression detection results have been compared to three of the existing solutions in literature. The performance measures used in all experiments are accuracy, precision and recall. These measures are based on the confusion matrix generated from each experiment, which has the format shown in Table 9 and Table 10 below.

Table 9: Confusion Matrix Template for Two-Class Prediction.

	Predicted	
Actual	Yes	No
Yes	True Positive (TP)	False Negative (FN)
No	False Positive (FP)	True Negative (TN)

Table 10: Confusion Matrix Template for Three-Class Prediction.

	Predicted Class		
Actual	Class 1	Class 2	Class 3
Class 1	True Negative (TN)	True Negative (TN)	False Positive (FP)
Class 2	True Negative (TN)	True Negative (TN)	False Positive (FP)
Class 3	False Negative (FN)	False Negative (FN)	True Positive (TP)

From the confusion matrix, we can say that accuracy or classification rate refers to the percentage of correctly classified instances and is calculated using Equation 63.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100\% \qquad (63)$$

Precision is a measure of exactness or quality. It is the percentage of test cases that are classified as X and have a true label of X. It is also referred to as the exactness of a classifier and is calculated using the below equation.

$$Precision = \frac{TP}{TP + FP} \tag{64}$$

Finally, recall is a measure of completeness or quantity. It is defined as the percentage of test cases that have a true label of X and are actually labelled as X and is calculated using the equation below.

$$Recall = \frac{TP}{TP + FN} \tag{65}$$

5.1. MPEG2 and HEVC Double and Triple Compression Detection Using Same Quantization Parameter

In this set of experiments, the results obtained for MPEG2 re-compression detection using the same re-compression QP are compared against those obtained for HEVC. The same quantization parameters, data set and experimental setup used in MPEG2 are replicated and used in HEVC to be able to achieve a fair comparison for the two coding schemes. HEVC re-compression detection is tested three times, once using the same features used in MPEG2 (MPEG2-specific features) and once using HEVC-specific features such as those related to HEVC CUs and finally once using both creating a full expanded feature set. The aim behind testing using MPEG2-specific features, HEVC-specific features and the full feature set is to identify the contribution of each of the feature sets on the classification results. The double compression detection results of the MPEG2 experiments and the HEVC experiments using the three feature sets are reported and compared with each other. The same experiments previously conducted for MPEG2 and HEVC are tested again but for triple compression detection and the results are reported. HEVC is found to have equal performance as MPEG2 for double compression detection. However, for triple compression detection, it is found to perform significantly better than MPEG2.

To summarize the results, HEVC and MPEG2 comparison shows that higher results are achieved for re-compression detection in HEVC for all experiments. HEVC produced higher results with a 100% classification rate being obtained for all double compression detection experiments when using the MPEG2-specific features as compared to MPEG2. When testing HEVC videos with the expanded feature set, same results are obtained as above as shown in Table 13. However, experimenting with only HEVC-specific features resulted into very low accuracies with a classification rate of 57% for double compression and 39% for triple compression as shown in Table 12. This shows that the features used in MPEG2 are sufficient to successfully classify re-compression and that the HEVC-specific features have no contribution to the classification rate. In triple compression detection, significantly higher results are achieved in HEVC where MPEG2 produced 64% accuracy while HEVC produced 99% accuracy. Table 9 presents a detailed comparison between the best performing experiments conducted and the results obtained for each. The corresponding confusion matrices, precision and recall obtained from the comparison of Table 11 have been reported in Table 14.

Table 11: A comparison between MPEG2 and HEVC double and triple compression detection.

	MPEG2			**HEVC(MPEG2-specific features)**		
	QP	**Classifier**	**Classification Rate (%)**	**QP**	**Classifier**	**Classification Rate (%)**
Double Compression Detection	22	KNN	**99**	36	KNN RF	**100**
Triple Compression Detection	7	KNN	**64**	12	RF	**99**

Table 12: HEVC re-compression detection using HEVC-specific features.

	HEVC (HEVC-specific features)		
	QP	**Classifier**	**Classification Rate (%)**
Double Compression Detection	QP=36	RF	**57**
Triple Compression Detection	QP=12	RF	**39**

Table 13: HEVC re-compression detection using full feature

	HEVC (Full feature set)		
	QP	**Classifier**	**Classification Rate (%)**
Double Compression Detection	QP=36	RF	100
Triple Compression Detection	QP=12	RF	99

Table 14: Confusion matrices, precision and recall for MPEG2 and HEVC re-compression detection.

	MPEG2						**HEVC (MPEG2-specific features)**					
	Confusion Matrix				**Precision**	**Recall**	**Confusion Matrix**				**Precision**	**Recall**
Double Compression Detection		C1	C2		0.97	1		C1	C2		1	1
	C1	94.12	5.88				C1	100	100			
	C2	11.76	88.24				C2	100	100			
Triple Compression Detection		C1	C2	C3	0.89	0.94		C1	C2	C3	0.97	1
	C1	47.1	41.2	11.8			C1	33.3	1.3	0		
	C2	35.3	52.94	11.76			C2	0	30.32	0.7		
	C3	2.94	5.88	91.8			C3	0	0	34.2		

5.2. HEVC Double Compression Detection using Same Bitrate with Comparison to Existing Work

In this set of experiments, the results obtained for HEVC double compression detection using the same recompression bitrate are reported and compared against the results obtained in existing literature. The results are reported for four different bitrates (800,1000,1200,1400) kbps. For each of the four bitrates, experiments are conducted for features extracted on a GoP-level and frame-level and for each of the feature sets, results are obtained when using a Random Forest classifier and a bi-LSTM deep learning network. The proposed solution is found to perform significantly better as compared to the existing methods found in literature. Frame-level features using Random Forest classifier and bi-LSTM network produced good but lower results compared to GoP-level features. The highest results are obtained when using GoP-level features and bi-LSTM network where a 100% classification rate is obtained for all the different bitrates. These results are significantly higher than those reported in Jiang's, Xu's and Liang's methods in [11, 13, 19]. Table 15 provides a detailed comparison between the results obtained by our proposed solution and the results found in existing literature. In Figure 12, a comparison is done for the average accuracies obtained across each of the proposed and existing methods for the conducted experiments, showing that the highest average accuracy is obtained using the proposed GoP-level feature set and bi-LSTM network. In Table 16, the confusion matrices for the lowest bitrate (800 kbps) have been reported for each of the four proposed methods as this bitrate is the most challenging amongst all. This is because the higher the compression, the higher the information loss, making recompression detection much more challenging. The corresponding precision and recall have also been reported.

Table 15: HEVC double compression detection using same bitrate with comparison to existing work.

	FRAME-LEVEL FEATURES		GOP-LEVEL FEATURES		EXISTING WORK		
B1,B2	Proposed Solution (RF)	Proposed Solution (bi-LSTM)	Proposed Solution (RF)	Proposed Solution (LSTM)	Jiang Method [19]	Xu Method [11]	Liang Method [13]
800,800	94%	98%	98%	100%	93%	92.5%	85%
1000,1000	96%	96%	97%	100%	95%	93%	87%
1200,1200	96%	96%	97%	100%	95.5%	92.5%	86.25%
1400,1400	96%	96%	98%	100%	96%	92%	86.25%
Average Accuracy	**95.5%**	**96.5%**	**97.5%**	**100%**	**94.88%**	**92.5%**	**86.13%**

Table 16: Confusion matrices, precision and recall for HEVC double compression detection.

		Confusion Matrix			Precision	Recall
Frame-level Features	**Proposed Solution (RF)**		C1	C2	0.81	0.95
		C1	95.32	4.68		
		C2	22.38	77.62		
	Proposed Solution (bi-LSTM)		C1	C2	0.96	1
		C1	48.00	0.00		
		C2	2.00	50.00		
GoP-level Features	**Proposed Solution (RF)**		C1	C2	0.85	0.98
		C1	98.29	1.71		
		C2	17.14	82.86		
	Proposed Solution (bi-LSTM)		C1	C2	1	1
		C1	100.00	0.00		
		C2	0.00	100		

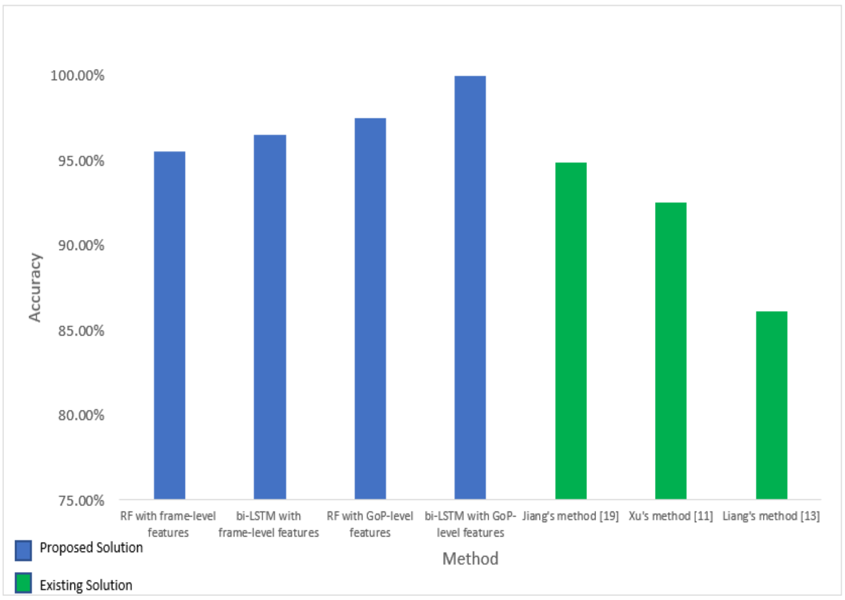

Figure 12: Average accuracies obtained for HEVC double compression detection.

5.3. HEVC Triple Compression Detection using Same Bitrate

The same experiments conducted for double compression detection are implemented again but for triple compression detection with promising results being obtained for each of the four bitrates. To the best of our knowledge, no existing literature was found for such application and thus the proposed solution for HEVC triple compression detection is considered novel in the field of video forgery detection. For triple compression detection, experiments are again conducted for frame-level and GoP-level features using the Random Forest classifier and bi-LSTM network for each of the four bitrates. Highest results were achieved when using GoP-level features and bi-LSTM network for classification with results ranging from 98% to 98.7% for the different bitrates under experiment. Table 17 presents a detailed comparison of the results obtained for each of the four bitrates using the different feature sets and classifiers. Table 18 shows the confusion matrices, precision and recall obtained for the most challenging case with the lowest bitrate of 800 kbps. Figure 13 then shows a comparison for the average accuracies obtained for each method across the four different bitrates, indicating that the highest average accuracy for triple compression detection is also obtained when using GoP-level features and bi-LSTM deep learning network.

Table 17: HEVC triple compression detection using same bitrate.

	FRAME-LEVEL FEATURES		GOP-LEVEL FEATURES	
B1,B2,B3	Proposed Solution (RF)	Proposed Solution (LSTM)	Proposed Solution (RF)	Proposed Solution (bi-LSTM)
800,800,800	82.67%	94%	76.67%	98%
1000,1000,1000	79.33%	92%	84.00%	98.7%
1200,1200,1200	75.33%	92.67%	86.67%	98.7%
1400,1400,1400	80%	94%	82.67%	98.7%
Average Accuracy	**79.33%**	**93.17%**	**82.5%**	**98.55%**

Table 18: Confusion matrices, precision and recall for HEVC triple compression detection.

		Confusion Matrix				**Precision**	**Recall**
Frame-level Features	**Proposed Solution (RF)**		C1	C2	C2	0.67	0.51
		C1	95.84	3.50	0.66		
		C2	21.68	53.46	24.86		
		C3	8.18	40.36	51.46		
	Proposed Solution (bi-LSTM)		C1	C2	C2	1	0.85
		C1	33.30	0.00	0.00		
		C2	0.00	27.30	0.00		
		C3	0.00	6.00	33.30		
GoP-level Features	**Proposed Solution (RF)**		C1	C2	C2	0.74	0.56
		C1	98.29	1.43	0.29		
		C2	19.43	61.43	19.14		
		C3	6.29	37.43	56.29		
	Proposed Solution (bi-LSTM)		C1	C2	C2	1	0.98
		C1	33.30	1.30	0.00		
		C2	0.00	31.30	0.00		
		C3	0.00	0.70	33.3		

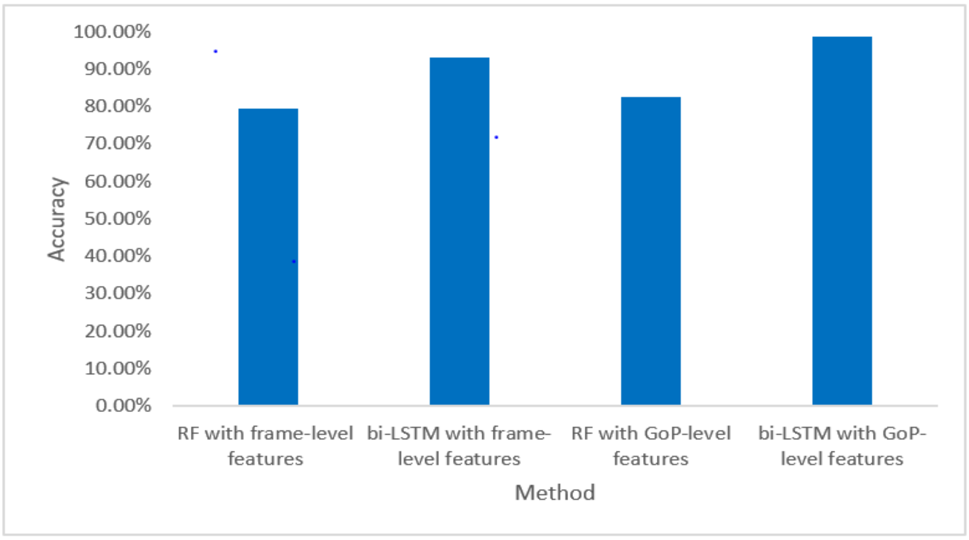

Figure 13: The average accuracies obtained in HEVC triple compression detection.

5.4. Summary of Results

This section briefly summarizes the best results obtained for each of the three sets of experiments conducted.

1. MPEG2 and HEVC Double and Triple Compression Detection using Same Quantization Parameter:

In this set of experiments, highest results for re-compression detection were obtained in HEVC coded videos when using the MPEG2-specific features discussed in the Feature Extraction and Classification section. HEVC achieved slightly higher results in double compression detection and significantly higher results in triple compression detection. In double compression detection, a 100% classification rate was obtained using both KNN and RF classifiers and a 99% classification rate was obtained using RF classifier for triple compression detection.

2. HEVC Double Compression Detection using Same Bitrate with Comparison to Existing Work:

In this set of experiments, results have been obtained for double compression detection in HEVC coded videos using the same re-compression bitrate for four different bitrates, two different feature sets and two different classifiers. Highest results are obtained when using GoP-level features and the bi-LSTM deep learning network with a classification rate of 100% being achieved for the four different bitrates. The results obtained are proven to be significantly higher than the three existing solution being compared against.

3. HEVC Triple Compression Detection using Same Bitrate

In this set of experiments, results have been obtained for triple compression detection in HEVC coded videos using the same re-compression bitrate. Experiments were conducted using frame-level and GoP-level features using both RF and bi-LSTM network with the highest results being obtained using the bi-LSTM network and GoP-level features as was the case for double compression detection. A 98% classification accuracy is achieved for the lowest and most challenging bitrate of 800 kbps and a 98.7% accuracy is achieved for the remaining three bitrates. Despite the promising results obtained, the classification rate of triple compression detection is still lower than that of double compression detection in all the experiments conducted due to having three classes, making the classification more challenging. Using three classes for triple compression detection instead of two will result in lower classification rates as distinguishing between the second and third compression becomes much more difficult compared to distinguishing between single (original) and triple compression. The results obtained have not been compared to existing solutions as to the extent of our knowledge, no existing literature was found to tackle the problem of triple compression detection in HEVC coded videos.

Chapter 6. Conclusion and Future Work

Digital video forensics is the process of identifying manipulation and forgery in videos to evaluate their trustworthiness and authenticity for use in court cases or similar investigations. Such field has gained much popularity in the past years due to the widespread of video capturing devices and mobile phones as well as the ease of use of video editing software such as Adobe, CyberLink, Wondershare, Apple Final Cut and many others. This gives the normal user very strong capability of editing and manipulating a video to hide information or alter its content without the need for any professional knowledge. The existence of several social media platforms also makes the development of more efficient video forensic techniques extremely crucial due to the widespread of videos on such platforms and thus, the significant need for identifying their authenticity before presenting them to the public. For such reasons, we propose a system towards the problem of digital forgery detection by focusing on compression-based video forensics. Compression-based forensics is one of the very important forensic techniques as the detection of video re-compression will indicate the existence of manipulation or forgery in the video.

In this work, we developed a system that improves on the existing solutions of double compression detection in MPEG2 videos that have the same recompression quantization parameter. We also proposed a new system for the detection of multiple compressions with a main focus on a maximum of three compressions. The same experiments for double and triple compression were also conducted for HEVC coded videos and a comparison is provided for the two coding schemes when the recompression QP is kept the same. In both codecs, two machine learning algorithms were used, KNN and RF, to evaluate and assess the performance. The highest results have been obtained for HEVC coded videos with a 100% classification rate for double compression and a 99% accuracy for triple compression detection.

In the proposed solution, we also developed a system that detects double and triple compression in HEVC coded videos that have the same recompression bitrate with comparison to some of the existing solutions. Two feature sets have been used, GoP-level and frame-level features, and four different bitrates have been tested. Along with the random forest classifier, we also introduced the use of the bi-LSTM deep learning network for the classification of recompression. The results obtained for double compression detection have been compared to three of the existing solutions in literature. Highest results were achieved when using the GoP-level features where a 97.5% average accuracy is achieved using the RF classifier and a 100% average accuracy is achieved using the bi-LSTM network for double compression detection, clearly outperforming the existing solutions. The same experiments were conducted for triple compression detection where to the best of our knowledge, no existing solutions have been implemented to tackle this problem. The highest average accuracy of 98.55% was achieved for triple compression detection when also using the GoP-level features with the bi-LSTM network. From the results obtained, we can clearly state the effectiveness of the proposed solution for both double and triple compression detection in HEVC coded videos when having the same recompression bitrate.

As for future work, more focus is to be put on the detection of double and triple compression with different coding parameters as well as the detection of a higher number of compressions. In addition, we would like to further investigate the problem of double and multiple compression with some underlying forgery, such as frame duplication, insertion or cropping, from which we shall identify the exact type of forgery being hidden within the re-compressions.

References

[1] M. C. Stamm, M. Wu, and K. J. R. Liu, "Information Forensics: An Overview of the First Decade," *IEEE Access,* vol. 1, pp. 167-200, 2013.

[2] P. He, X. Jiang, T. Sun, and S. Wang, "Double compression detection based on local motion vector field analysis in static-background videos," *Journal of Visual Communication and Image Representation,* vol. 35, pp. 55-66, 2016.

[3] J. Abbasi Aghamaleki and A. Behrad, "Inter-frame video forgery detection and localization using intrinsic effects of double compression on quantization errors of video coding," *Signal Processing: Image Communication,* vol. 47, pp. 289-302, 2016.

[4] P. He, X. Jiang, T. Sun, and S. Wang, "Detection of double compression in MPEG-4 videos based on block artifact measurement," *Neurocomputing,* vol. 228, pp. 84-96, 2017.

[5] X. Jiang, W. Wang, T. Sun, Y. Q. Shi, and S. Wang, "Detection of Double Compression in MPEG-4 Videos Based on Markov Statistics," *IEEE Signal Processing Letters,* vol. 20, no. 5, pp. 447-450, 2013.

[6] T. Sun, W. Wang and X. Jiang, "Exposing video forgeries by detecting MPEG double compression," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, 2012, pp. 1389-1392.

[7] Y. Su and J. Xu, "Detection of Double-Compression in MPEG-2 Videos," *2010 2nd International Workshop on Intelligent Systems and Applications*, Wuhan, 2010, pp. 1-4.

[8] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double MPEG compression," in 2012 *Proceedings of the 8th workshop on Multimedia and security*, Geneva, 2006.

[9] H. Ravi, A. V. Subramanyam, G. Gupta, and B. A. Kumar, "Compression noise based video forgery detection," in *2014 IEEE International Conference on Image Processing (ICIP),* 27-30 Oct. 2014, pp. 5352-5356.

[10] Z. Huang, F. Huang, and J. Huang, "Detection of double compression with the same bit rate in MPEG-2 videos," in *2014 IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP)*, 9-13 July 2014 2014, pp. 306-309.

[11] Q. Xu, T. Sun, X. Jiang, and Y. Dong, "HEVC Double Compression Detection Based on SN-PUPM Feature," in *Digital Forensics and Watermarking*, Cham, C. Kraetzer, Y.-Q. Shi, J. Dittmann, and H. J. Kim, Eds., 2017// 2017: Springer International Publishing, pp. 3-17.

[12] R.-S. Jia, Z.-H. Li, Z.-Z. Zhang, and D.-D. Li, "Double HEVC Compression Detection with the Same QPs Based on the PU Numbers," *ITM Web Conf.*, vol. 7, pp. 02010, 2016.

[13] X. Liang, Z. Li, Y. Yang, Z. Zhang, and Y. Zhang, "Detection of Double Compression for HEVC Videos With Fake Bitrate," *IEEE Access*, vol. 6, pp. 53243-53253, 2018.

[14] M. Huang, R. Wang, J. Xu, D. Xu, and Q. Li, "Detection of Double Compression for HEVC Videos Based on the Co-occurrence Matrix of DCT Coefficients," in *Digital-Forensics and Watermarking*, Cham, Y.-Q. Shi, H. J. Kim, F. Pérez-González, and I. Echizen, Eds., 2016// 2016: Springer International Publishing, pp. 61-71.

[15] Q. a. W. R. a. X. D. Li, "Detection of Double Compression in HEVC Videos Based on TU Size and Quantized DCT Coefficients," *IET Information Security*, vol. 13, pp.1049, 2018.

[16] L. Yu, Y. Yang, Z. Li, Z. Zhang, and G. Cao, "HEVC double compression detection under different bitrates based on TU partition type," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 1, p. 67, 2019.

[17] M. C. Stamm, W. S. Lin, and K. J. R. Liu, "Temporal Forensics and Anti-Forensics for Motion Compensated Video," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1315-1329, 2012.

[18] D. Vazquez-Padin, M. Fontani, T. Bianchi, P. Comesana, A. Piva, and M. Barni, "Detection of video double encoding with GOP size estimation," in *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2-5 Dec. 2012, pp. 151-156.

[19] X. Jiang, P. He, T. Sun, and R. Wang, "Detection of Double Compressed HEVC Videos Using GOP-Based PU Type Statistics," *IEEE Access*, vol. 7, pp. 95364-95375, 2019.

[20] R. Korada, P. Gupta, T. Raghu, and K. Suman, "Algorithmic Optimizations for Software-only MPEG-2 Encoding," *IEEE Transactions on Consumer Electronics,* vol. 50, no. 1, pp. 366-375, 2004.

[21] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1649-1668, 2012.

[22] A. A. Elrowayati, M. F. L. Abdullah, A. A. Manaf, and A. S. Alfagi, "Tampering detection of double-compression with the same quantization parameter in HEVC video streams," in *2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, 24-26 Nov. 2017, pp. 174-179.

[23] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN Model-Based Approach in Classification," Berlin, Heidelberg, 2003: Springer Berlin Heidelberg, pp. 986-996.

[24] C. Antonio, S. Jamie, and K. Ender, *Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning*. now, 2012, p. 1.

[25] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001.

[26] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning,* vol. 63, no. 1, pp. 3-42, 2006.

[27] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone, "Classification and Regression Trees," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* vol. 1, no. 1, pp. 14-23, 1983.

[28] M. Hassan and T. Shanableh, "Predicting split decisions of coding units in HEVC video compression using machine learning techniques," *Multimedia Tools and Applications,* vol. 78, no. 23, pp. 32735-32754, 2019.

[29] A. Mohan and D. Gaitonde, "A Deep Learning based Approach to Reduced Order Modeling for Turbulent Flow Control using LSTM Neural Networks", *arXiv: Computational Physics,* 2018.

[30] T. Shanableh, "Detection of frame deletion for digital video forensics," *Digit. Investig.,* vol. 10, no. 4, pp. 350-360, 2013.

[31] T. Shanableh, "No-Reference PSNR Identification of MPEG Video Using Spectral Regression and Reduced Model Polynomial Networks," *IEEE Signal Processing Letters,* vol. 17, no. 8, pp. 735-738, 2010.

Vita

Seba Youssef was born in 1994, in Jeddah, Saudi Arabia. She received her primary education in Saudi Arabia and her secondary education in Cairo, Egypt. She received her B.Sc. degree in Computer Engineering from the American University of Cairo in 2017 as a cum laude. During her studies, she joined multiple extracurricular activities and internships.

In September 2018, Seba joined the Computer Engineering Master's program at the American University of Sharjah as a graduate teaching assistant. Her research interests are in data science, machine learning, and image and video processing.