AUTOMATIC VIDEO SUMMARIZATION USING HEVC AND CNN FEATURES

by

Obada Issa

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

November 2022

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: Obada Issa

Date: November 2022

Approval Signatures

We, the undersigned, approve the Master's Thesis of Obada Issa

Title: AUTOMATIC VIDEO SUMMARIZATION USING HEVC AND CNN FEATURES

Date of Defense: 29-Nov-2022

Name, Title and Affiliation	Signature
Dr. Tamer Shanableh Associate Professor, Department of Computer Science and Engineering Thesis Advisor	
Dr. Usman Tariq Associate Professor, Department of Electrical Engineering Thesis Committee Member	
Dr. Gerassimos Barlas Professor, Department of Computer Sc. & Eng. Thesis Committee Member	
Dr. Imran Zualkernan Head Department of Computer Science and Engineering	
Dr. Lotfi Romdhane Associate Dean for Graduate Affairs and Research College of Engineering	
Dr. Fadi Aloul Dean College of Engineering	
Dr. Mohamed El-Tarhuni Vice Provost for Research and Graduate Studies Office of Research and Graduate Studies	

Acknowledgements

I would like to thank the American University of Sharjah for providing me with the assistantship program scholarship to pursue my Master’s degree and my deepest thanks and appreciation to my advisor Dr. Tamer Shanableh for providing knowledge, guidance, support, and motivation throughout my research stages. I’m deeply beholden for his great assistance, worthy discussion, and suggestions.

I would also like to thank the professors of the Computer Science and Engineering department at the American University of Sharjah who taught me the master level courses with mighty teaching methods and skills. I really appreciate their dignified advice and motivation.

Abstract

With the incredible surge of the internet and surveillance footage, there is a vast number of digital videos. The need to summarize these videos within databases is very crucial. This is where video summarization comes in handy. Video summarization can be achieved by a number of techniques. This study proposes a novel solution for the detection of key-frames for static video summarization. We preprocessed the well-known video datasets by coding them using the HEVC video coding standard. During coding, 64 proposed features were generated from the coder for each frame. Additionally, we extracted RGB frames from the original raw videos and fed them into pre-trained CNN networks for feature extraction. These include GoogleNet, AlexNet, Inception-ResNet-v2, and VGG16. The modified datasets are made publicly available to the research community. A subset of the proposed HEVC feature set was used to identify duplicate or similar frames and eliminate them from the video. We also propose an elimination solution based on the sum of the absolute differences between a frame and its motion-compensated predecessor. The proposed solutions are compared with existing works based on an SIFT flow algorithm that uses CNN features. Subsequently, an optional dimensionality reduction based on stepwise regression was applied to the feature vectors prior to detecting key-frames. The proposed solution is compared with existing studies that use sparse autoencoders with CNN features for dimensionality reduction. The accuracy of the proposed key-frame detection system was assessed using the Positive Predictive Values, Sensitivity, and F-score metrics. Combining the proposed solution with Multi-CNN features and using a Random Forests classifier, it was shown that the proposed solution achieved an average F-score of 0.98.

Key words: ***Static Video Summarization, Convolution Neural Networks (CNN), Duplicate Frames, Sparse Autoencoders, Random Forests classifier, Video Coding, High Efficiency Video Codec (HEVC), Motion estimation, Motion compensation, Stepwise regression.***

Table of Content

List of Figures

List of Tables

Chapter 1. Introduction

With the incredible surge of the internet and surveillance footage, there is a vast number of digital videos. According to the latest estimates in 2020, over 500 hours of video are uploaded to YouTube every minute, or over 700,000 hours per day. YouTube is not the only digital video host on the internet of course, there are tens and hundreds of other social network platforms with huge amounts of video content. Another source of huge video content is in security or surveillance applications, where there are millions of cameras around the globe recording footage at all times during the day. Hence, the need to summarize these videos within databases is very crucial for easier navigation and retrieval for both general users and database administrators. This is where video summarization comes in handy. Video summarization is the process of generating a meaningful summary of the original video, which in return facilitates video retrieval, anomaly detection [1] and activity monitoring. Some real-life applications of video summarization in the entertainment field are, but not limited to, automatic generation of movies or TV shows trailers or highlights of a sports or musical event. There are also other applications of video summarization in the digital surveillance field like automatic summary generation of the last 24-hours of surveillance footage for time-sensitive security monitoring.

1.1. Overview

Video summarization can be achieved by a number of methods or techniques. These techniques can be categorized into two groups [2]. The first is selecting sections or short cuts of the original video, and the second is selecting key-frames that represent the original video. Key-frame selection techniques require manual human annotation of the videos in order to automate the training of frame selection. This technique is the more common one of the two video summarization techniques mentioned. Therefore, this research focuses on video summarization by automatically selecting key-frames in a video.

The video summarization task is computationally demanding, and newer, more efficient approaches are always needed. If all frames within a video are examined for selection, the summarization process can be slow, and time and computational power are wasted on redundant or similar frames. Additionally, for any set of features, space reduction should be used to accelerate the process and ensure that only meaningful features are considered [3]. This study aims to address these two issues.

Deep Learning has gained tremendous popularity in recent years when it comes to generation tasks in image and video processing. There are many tools that can be used either independently or in combination to achieve the desired results. Most notably, Conventional Neural Networks (CNN) [4] and Random Forests [5].

1.2. Problem Statement

Because of social media and video monitoring, the number of digital videos is growing at an exponential rate. To make information retrieval easier and decrease data storage needs, video summarization is required. However, the video summarization task is computationally taxing. If all frames within a video are to be examined for selection then the summarization process can be extremely slow and precious time and computational power is wasted on redundant or similar frames. Also, for any set of features, space reduction should be used to speed-up the process and ensure that only meaningful features are considered. This research aims at addressing these two issues.

The Deep Learning community happens to fall in the computer and data science field, while and video compression community falls in the electrical engineering field. The separation in these research areas leads the Deep Learning community to often lack sufficient or professional comprehension in video compression. In the Deep Learning world, and with the growth of the High Efficiency Video Codec (HEVC) [6] video standard, HEVC information found in the video bitstream is often ignored and not used to its full potential. This research intends on using low level HEVC features in combination with CNN features to aid in the video summarization task.

1.3. Thesis Objectives

This research's main priority is the examination of low-level HEVC features, and how useful they can be in the video summarization task. The integration of HEVC features can take several forms, including merging bit stream information with CNN feature maps derived from image groups, or constructing a separate channel for deep leaning based on HEVC bit stream information and then fusing the output with classical CNN based results. This research also aims at introducing novel methods for the elimination of similar frames in the preprocessing stage of the methodology. The aforementioned contributions should be massive enough, but the most influential contribution of this research is in the form of incorporating Deep Learning-based video summarization techniques with the use of HEVC features. The planned efforts will also look at the impact of the offered improvements on the video summarizing process' overall correctness and performance.

1.4. Research Contribution

A summary of this research contributions to the body of the literature is in the following points:

- The introduction of the low-level HEVC feature sets suitable for video summarization. HEVC features could also be used in a variety of applications where image or video processing is involved.
- The proposal of novel methods for frame elimination:
  - Frame elimination based on HEVC features.
  - Frame elimination based on motion estimation and compensation.
- These contributions resulted in promising results that outperform state-of-the-art works in the literature.

1.5. Thesis Organization

This paper is structured as follows: Chapter 2 has the literature review, where the reviewed works are grouped based on the main architecture used. Followed by Chapter 3 which has the proposed methodology and explains each component of the system in detail and at length. Then, we have Chapter 4 that presents our experimental results and compares them against previous state-of-the-art works in the literature. Finally, Chapter 5 contains the conclusion summary and future work.

Chapter 2. Literature Review

Video summarization has been researched and studied extensively in the past decade due to the incredible surge of videos online and the challenges that come with sorting and saving these videos in huge databases. This section summarizes the efforts found in the literature that tackle the video summarization task, especially Deep Learning-based methods.

2.1. LSTM-based Video Summarization

Long Short-Term Memory (LSTM) networks were employed by [7] to describe the variable-range temporal dependence among video frames and to create representative and concise video summaries, casting the job as a structured prediction problem. Their approach generates state-of-the-art performance on benchmark datasets by correctly accounting for sequential structure, which is essential for producing insightful video summaries. They also deal with the requirement for a sizable volume of annotated data for sophisticated training methods. The major goal was to make use of supplementary annotated video summarization datasets, despite the fact that they varied widely in terms of visual appearance and content.

In the work by [8], they design a 2-layer network where they have a sliding bidirectional LSTM that acts as a 1D filter that slides through the frame to identify shots boundaries by a threshold and feature extraction. Shallow features are extracted by concatenating color histograms, SIFT and optical flow. Deep features are from the VGG16 pre-trained CNN network. The second layer is also a bidirectional LSTM, which captures the forward and backward temporal dependencies among shots, and predicts which shots are most representative to the video content. They achieve very acceptable results on the following datasets: SumMe, VTW and CoSum.

Recurrent Neural Networks (RNNs) are limited with modelling long-term temporal dependencies due to the restricted memory storage unit, which is disadvantageous for summarizing videos with thousands of frames. Due to this, a stacked memory network called SMN is presented by [9] to model the long dependency among video frames so that redundancy could be minimized in the video summaries produced. SMN consists of two components: an LSTM layer and memory layer, where each LSTM layer is augmented with an external memory layer. In particular, multiple LSTM layers and memory layers are stacked hierarchically to integrate the learned representation from prior layers. By combining the hidden states of the LSTM layers and the read representations of the memory layers, SMN is able to derive accurate video summaries for individual frames. Compared with the existing RNN-based methods, SMN is particularly good at capturing long temporal dependency among frames with few additional training parameters. Experimental results on SumMe and TVSum, demonstrate SMN's ability to outperform state-of-the-art works under various settings.

The authors in [10] achieve Multi-Video Summarization (MVS) by integrating deep neural network based soft computing techniques in a two-tier framework. The first online tier performs target-appearance-based shots segmentation and stores them in a lookup table that is then transmitted to the cloud for further processing. The second-tier extracts deep features from each frame of a sequence in the lookup table and passes them to deep bidirectional LSTM to acquire probabilities of informativeness and generates a summary. Experimental evaluation was done on benchmark dataset VSUMM and industrial surveillance data from YouTube.

To mimic the way a human would select key-shots for a video summary, an attention mechanism is needed. Therefore, [11] proposed a video summarization framework named attentive encoder–decoder networks for video summarization (AVS), where the encoder uses a bidirectional LSTM to encode the contextual information among the input video frames. As for the decoder, two attention-based LSTM networks are exploited by using additive and multiplicative objective functions. The results demonstrate the superiority of the proposed AVS-based approach against state-of-the-art works.

In the research by [12], the goal was to predict a score ranging from 0 to 1 for each shot. The higher the score, the more likely the shot will be selected for the final summary. To this end, a self-attention binary neural tree (SABTNet) model is proposed, including the GoogleNet pre-trained network, shot encoding, branch routing, self-attention, and score prediction modules. The model divides videos into non-overlapping shots by shot segmentation algorithms. Then, Kernel Temporal Segmentation (KTS) is used for shot segmentation. After that, GoogleNet is used for feature extraction for individual frames, and re-encode frame features within the same shot into shot-level features by a bidirectional LSTM. Then, a full binary tree exposes each shot to various evaluation paths for comprehensive scoring. For the tree, each non-leaf node is accompanied by a branch routing module to determine the probability of which path the shot will be sent through for evaluation. Through branch routing, predicted scores are obtained and their proportions based on different root-to-leaf paths.

In the paper by [13], TTH-RNN was proposed for the video summarization task, which aimed to avoid the extremely large feature-to-hidden mapping matrices caused by the high-dimensional video features, and enhance traditional RNNs in long-range temporal dependence exploration. TTH-RNN contains a tensor-train embedding layer and a hierarchical LSTM, where the tensor-train embedding layer is designed to embed video features to a lower dimensional space before being input to the LSTM, and the embedding matrix was factorized to reduce training parameters. The hierarchical LSTM contains two layers. The first layer is a single LSTM to capture the intra-subshot temporal dependence. The second layer is a bidirectional LSTM to encode inter-subshot temporal dependence from the forward and backward directions. Experimental results on four popular datasets demonstrated the superiority of TTH-RNN in the video summarization task.

2.2. CNN-based Video Summarization

The paper by [14] presents a video summarization technique for generating quick previews for online videos. They propose using deep video features that can encode various levels of content semantics, including objects, actions, and scenes, improving the efficiency over standard video summarization techniques. They design a deep network that maps videos and their descriptions to a common semantic space and jointly trained it with associated pairs of videos and descriptions. To generate a video summary, they extract the deep features from each segment of the original video and apply a clustering-based technique on them. They evaluate the summaries on the SumMe dataset. The results demonstrated the advantages of incorporating deep semantic features in a video summarization technique.

The authors in [15] observed two main issues in unsupervised video summarization: (I) Ineffective feature learning due to flat distributions of output importance scores for each frame, and (II) training difficulty when dealing with long-length video inputs. To alleviate the first problem, they propose variance loss. The proposed variance loss allows a network to predict output scores for each frame with high discrepancy which enables effective feature learning and significantly improves model performance. For the second problem, they design a novel two-stream network named Chunk and Stride Network (CSNet) that utilizes local (chunk) and global (stride) temporal view on the video features. CSNet gives better summarization results for long-length videos compared to the existing methods. To deal with dynamic information in videos, they also have an attention function. On two benchmark datasets, they show how well the proposed work compares to the state-of-the-art.

In this paper, [16] formulate video summarization as a sequence labeling problem. Unlike existing approaches that use recurrent models, they propose using fully convolutional sequence models. They establish a connection between semantic segmentation and video summarization, and then adapt popular semantic segmentation networks for video summarization. Extensive experiments and analysis on two benchmark datasets demonstrate the effectiveness of the models.

Video summarization is formulated by [17] as a sequential decision-making process and develop a Deep Summarization Network (DSN). For every video frame, DSN predicts a probability that indicates how likely a frame is selected, and then takes actions based on the probability distributions to select frames that form video summaries. For training, an end-to-end, reinforcement learning-based framework is presented, with a novel reward function that jointly accounts for diversity and representativeness of generated summaries and does not rely on labels or user interactions at all, making the model fully unsupervised. During training, the reward function judges how diverse and representative the generated summaries are, while DSN strives for earning higher rewards by learning to produce more diverse and representative summaries. Extensive experiments on two benchmark datasets show that their method is even superior to most supervised approaches in the literature.

The authors in [18] present a Deep Learning approach to summarizing long soccer videos by leveraging the spatiotemporal learning capability of three-dimensional CNNs (3D-CNN) and an LSTM. The methodology involves a step-by-step development of a Residual Network (ResNet) based 3D-CNN that recognizes soccer actions. Also, manual annotation of 744 soccer clips for training. Finally, training an LSTM network on soccer features extracted by the proposed ResNet based 3D-CNN. A video input is modeled as a sequential concatenation of video segments whose inclusion in a summary video production is based on its validated relevance. To evaluate the proposed summarization system, 10 soccer videos were summarized and subsequently evaluated by 48 participants polled from 8 countries using the Mean Opinion Score (MOS) scale. Collectively, the summarized videos received a 4 of 5 MOS.

The authors in [19] set to solve the imbalanced class distribution problem in video summarization. The framework is a two-stream deep architecture with cost-sensitive learning to handle the class imbalance in feature learning. In the spatial stream, RGB images are used to represent the appearance of video frames, and in the temporal stream, multi-frame motion vectors with Deep Learning framework is used to represent and extract temporal information of the input video. Empirical validations demonstrate that the model achieves performance improvement over the existing and state-of-the-art methods. The model can also automatically preserve connections between consecutive frames. Although the summary is constructed based on the frame level, the final summary is comprised of informative and continuous segments instead of individual separate frames.

In the journal paper by [20], given an input video, a deep summarization network is used to extract deep feature representations from the input video sequence and sequentially models the frame features. It adopts an encoder-decoder CNN structure. The encoder network is a diagnostic view plane detection network pre-trained with ultrasound standard plane detection annotations. The decoder network takes the extracted feature maps of each input frame as input and feeds them into a Bi-directional LSTM to analyze features of both, past and future frames. Following the feature extraction, the reinforcement learning (RL) network interprets the diagnostic video summarization task as a decision-making process, in which a decision is to include a current frame in the summary or not. The RL network accepts latent scores from the Bi-LSTM as input and takes actions at on whether a frame should be selected into the summary set or not by maximizing the expected rewards. The rewards are computed on the quality of the selected frames in terms of their representativeness, diversity, as well as the likelihood of being a standard diagnostic plane. The method is superior to alternative video summarization methods and that it preserves essential information required by clinical diagnostic standards.

Producing a video summary of considerable significance while meeting the demands of Internet of Things (IoT) monitoring systems with limited resources is a difficult challenge. In order to summarize surveillance videos recorded in IoT environments, this paper [21] proposes a novel, computationally efficient method by using a deep CNN architecture with hierarchical weighted fusion. The framework's initial phase creates discriminative rich features for shot segmentation that are taken from deep CNNs. The method then uses aesthetic and entropy factors to preserve the summary's interest and variety in addition to image memorability predictions from a tweaked CNN model. In order to provide an aggregated score for the efficient calculation of the extracted features, a hierarchical weighted fusion approach is third described. The combined score is then used to choose the best key-frames for the final video summary, creating an attention curve.

The authors in [22] develop a method to detect key-frames for static video summarization. Their method is based on feature vectors taken from 4 pre-trained Multi-CNN models. The features are then inputted into a sparse autoencoder, which then outputs a combined representation of the input feature vectors. Finally, the key-frames are chosen based on the combined feature vectors using a Random Forests classifier. The datasets used are VSUMM and OVP and ground-truth scores derived from user summaries. The overall method performs better than all state-of-the-art methods in the study.

This paper by [23] proposes a deep framework called Deep Hierarchical LSTM Networks with Attention for Video Summarization (DHAVS) that has delicate feature extraction, modeling of temporal dependencies, and video summary generation; All in response to the LSTMs' inability to handle longer video sequences. To extract spatio-temporal features, they specifically use 3D CNNs rather than 2D CNNs, and they create an attention-based hierarchical LSTM module to capture the temporal correlations between video frames. A cost-sensitive loss function is created, and they also approach video summarizing as an unbalanced class distribution problem. According to experimental findings, performance on the SumMe and TVSum datasets has improved significantly.

Many of the currently available summarizing techniques solely analyze the visual components of the video input, omitting the impact of audio elements on the resulting summary. Hence, [24, p.] provide a powerful video summarizing method to address these problems, one that extracts key-frames from the raw video input while processing both the visual and audio information. The structural similarity index is used to determine how similar the frames are to one another, and the mel-frequency cepstral coefficient (MFCC) aids in feature extraction from the associated audio signals. A deep CNN model is used to improve the resulting key-frames and produce a list of potential key-frames that ultimately make up the data summary. According to experimental findings, using audio features along with an effective refining method, followed by an optimization function, produces superior summary results than using traditional summarizing methods.

2.3. GAN-based Video Summarization

The key idea of the study by [25] is to develop a deep summarizer network in an unsupervised fashion to reduce the special distance between training videos and their summaries. Such a summarizer can then be applied on a new video for estimating its optimal summarization. For learning, they specify a novel generative adversarial framework (GAN), consisting of the summarizer and discriminator. The summarizer is the autoencoder long short-term memory network (LSTM) aimed at, first, selecting video frames, and then decoding the obtained summarization for reconstructing the input video. The discriminator is another LSTM aimed at distinguishing between the original video and its reconstruction from the summarizer. The summarizer LSTM is cast as an adversary of the discriminator, i.e., trained so as to maximally confuse the discriminator. This learning is also regularized for sparsity. Evaluation on four benchmark datasets, consisting of videos showing diverse events in first-and third-person views, demonstrates competitive performance in comparison to fully supervised state-of-the-art approaches.

Once again, the problem is formulated as a sequence-to-sequence task, where the input sequence is an original video, and the output sequence is its summarization. The architecture put forward by [26] is built on GAN training and takes the advantages of supervised and unsupervised methods for video summarization. The splitting points of summarization segments are generated by an attention-aware Ptr-Net generator. A 3D CNN classifier is used as the discriminator to determine if a fragment is from a produced summary or a ground-truth. The method achieves state-of-the-art results on

SumMe, TVSum, YouTube, and LoL datasets with 1.5% to 5.6% improvements. The framework can also overcome the unbalanced training-test length in the seq2seq problem, and the discriminator is effective in leveraging unpaired summarizations to achieve better performance.

The paper by [27] presents an approach that integrates an attention mechanism to identify the significant parts of the video, and is trained in an unsupervised manner via generative adversarial learning (GAN). Starting from the SUM-GAN model, they develop an improved version of it (called SUM-GAN-sl) that has a significantly reduced number of learned parameters, performs incremental training of the model’s components, and applies a stepwise label-based strategy for updating the adversarial part. Subsequently, an attention mechanism is introduced to SUM-GAN-sl in two ways: i) by integrating an attention layer within the variational autoencoder (VAE) of the architecture (SUM-GAN-VAAE), and ii) by replacing the VAE with a deterministic attention autoencoder (SUM-GAN-AAE). Experimental evaluation on the datasets SumMe and TVSum documents the contribution of the attention autoencoder to faster and more stable training of the model, resulting in a significant performance gain with respect to the original model and demonstrating the competitiveness of the proposed SUM-GAN-AAE against the state of the art.

The proposed approach by [28] employs knowledge distillation for choosing key-frames and GANs for feature extraction. The main component of the given model is adversarial learning, which makes sure that the video's various and representative parts are included in the extracted features. A convolutional recurrent autoencoder serves as the generator, and it gains knowledge about the hidden video representation through reconstruction loss. A discriminator that distinguishes between the original and recreated video samples comes next. A knowledge distillation phase that uses a basic network as a key-frame selection follows the adversarial network. Comprehensive analyses performed on both open-source and private datasets support the usefulness of GANs and the knowledge distillation stage for video summarization. Evaluations show that the suggested methodology creates summaries that are varied, representative, and compact.

2.4. Other Video Summarization Models

Some works opted to base their generated video summaries on user-defined queries or requests. In the paper by [29], they were among the first to incorporate user queries into the summarization process. They achieve this by creating a probabilistic model for query-focused extractive video summarization called Sequential and Hierarchical Determinantal Point Process (SH-DPP). In response to a user query and a video sequence, the algorithm summarizes the video by picking out the most important frames. The option to include a shot in the summary is based on both the shot's significance within the context of the video and its relevance to the user query. They claim that search engines can benefit greatly from query-focused video summarization.

By incorporating representative and interestingness objectives computed from features from a joint vision-language embedding space, the paper by [30] extends a submodular summarization approach to demonstrate that visual representations supervised by freeform language make a good fit for video summarization. Their research also demonstrates that the vision-language embedding may be learned from common vision-language datasets and then transferred to video without the need for training on domain-specific data.

The key concept of the sequence-to-sequence learning model presented by [31] is to complement the discriminative losses with another loss that assesses whether the generated summary still maintains similar information as in the original video. This is done by rewriting the video summarization task as a sequence-to-sequence task. Additionally, they add a second "retrospective encoder" to conventional sequence learning models that incorporates the anticipated summary into a semantic space. The representation found in the semantic space is then contrasted with the original video's embedding in the same location. The idea is that for a video and its related summary, both embeddings should be similar to one another. Therefore, their method increases the distances between mismatched pairs while minimizing the distances between matched pairs, adding a metric learning loss to the discriminative loss. The fact that the measure for learning loss enables learning from videos without automatically created summaries is a significant advantage.

Incomplete selection of video feature points can result in poor video summaries, hence, a compressed domain video summarization generation algorithm is proposed by [32] based on HEVC intra-frame coding. Firstly, the weighted luminance and chrominance mode numbers are counted at the decoding end, respectively. Then, the feature vectors composed of the above two mode numbers are fused and normalized to obtain a mode feature histogram. Secondly, the normalized mode feature histograms are assigned different weight factors, and the two are merged into a new mode feature histogram model to reflect the texture features of the image. Finally, the histogram difference method is used to determine the similarity between two-frame mode feature histograms, and ultimately the histogram classification is used to generate the video summaries. The experimental results show that the accuracy of key-frame extraction on the OVP dataset is 0.82, the error rate is 0.64. And that, the average recall rate achieves 82.0%, and the average F-score is 73.3%. The key-frames extracted by the algorithm better reflect the image texture, and the video summary quality is further improved.

As most existing methods can only extract the static images of videos as the content summarization and ignore the representation of motion information, there is a gap to fill here. To cope with these issues, a novel framework for an efficient video content summarization as well as video motion summarization is proposed by [33]. Initially, Capsules-Net is trained as a spatio-temporal information extractor, and an inter-frames motion curve is generated based on those spatio-temporal features. Subsequently, a transition effects detection method is proposed to automatically segment the video streams into shots. Finally, a self-attention model is introduced to select key-frames sequences inside the shots; thus, key static images are selected as video summarization, and optical flows can be calculated as video motion summarization. The ultimate experimental results demonstrate that the method is competitive on VSUMM, TVSum, SumMe, and RAI datasets about shot segmentation and video summarization, and can also represent a good motion summarization result.

Because they only provide fixed video summary for a particular input video, regardless of the user's actual needs, traditional summarizing methods have a negative effect on the function of video exploration. In this work, [34] describe a technique that takes as its input a text-based query and outputs a video summary of it. They achieve this by posing an end-to-end deep learning-based technique for query-controllable video summarizing to produce a query-dependent video summary and modeling video summarization as a supervised learning issue. A video summary controller, a video summary generator, and a video summary output module make up the suggested technique.

The paper [35] proposed an unsupervised video summarization method by motion-based frame selection and a novel clustering validity index to determine the optimal representatives of the original video. The proposed framework segments shots and selects candidate frames by evaluating their forward and backward motion and can automatically select representatives to highlight all the significant visual properties. Shots are segmented uniformly and the frame with the largest motion is extracted in each segmentation to form the video candidate frame subset. Then Affinity Propagation combined with the validity index is used to automatically select the optimal representatives from the candidate frame subset.

In this paper, [36] offers CLIP-It, a single framework for dealing with both conventional and query-focused video summarization, which are normally addressed separately in the literature. They have a multimodal transformer that learns to score video frames and correlate them with a user-defined written query using a unified language. The model can then be trained without supervision based on real-world data. They significantly exceed baselines and earlier work on the TVSum and SumMe benchmark datasets as well as the query-focused video summarization dataset QFVS.

A multiscale hierarchical attention model for supervised video summarization is introduced in this study by [37]. Their approach takes advantage of the underlying hierarchical structure of video sequences, in contrast to most existing supervised approaches that use bidirectional LSTM networks and learns both the short-range and long-range temporal representations via an intra-block and an inter-block attention. To learn local and global information, they first divide each video sequence into equal-length blocks and use intra-block and inter-block attention. They then combine the representations at the frame, block, and video levels to estimate the importance score at the frame level. After that, they partition shots and calculate shot-level importance scores. Finally, they choose the key shots to create video summaries. Additionally, their approach is expanded into a two-stream architecture that makes use of appearance and motion data. The effectiveness of their solution in comparison to the literature has been validated by experimental results on the SumMe and TVSum datasets.

According to [38], query-based video summarization is flawed in two ways: First, the text query might not be sufficient to capture the user's precise needs. Second, the user is unable to make changes once the summaries are created, even though the user's demands would be delicate and require interactive adjusting. IntentVizor, an interactive video summarizing framework driven by general multimodality inquiries, is proposed a solution to these two issues. The user's needs can be expressed using an input query that includes both text and video samples. They also describe these multi-modality, finer-grained queries as editable user "intents" that can more accurately express the needs of the system. To provide more satisfactory summaries, users can interact, control, and modify these intentions.

Chapter 3. Proposed Methodology

The initial stage of this research is to replicate the work in [22] and experiment with alternative frame deletion and dimensionality reduction methods. After that, the classifier is also replaced with a different model. The purpose of this it to first establish a robust base-model before introducing HEVC features. In the work by [22], they use the SIFT Flow [39] algorithm for the deletion of similar or redundant frames and a sparse autoencoder for reducing the dimension of the feature space. For elimination of similar or redundant frames, two novel methods are proposed based on motion estimation and motion compensation, and HEVC features. To reduce the dimension of the feature space, we use stepwise regression. For classification (or key-frames selection) we use a Random Forests classifier. A novel method is also developed for the extraction of low-level HEVC features. HEVC features are to be tested in a separate channel and then fuse the results with the CNN results, or they are concatenated with CNN features for a final result. A general overview of the system architecture is in Figure 1.

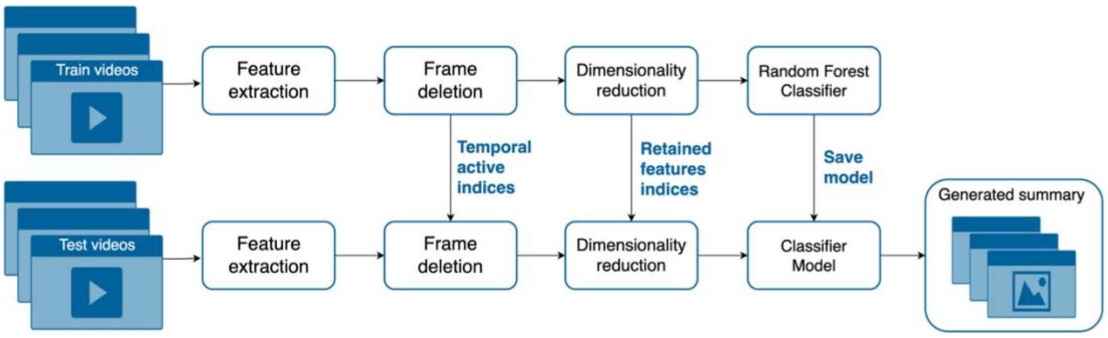

Figure 1: General overview of the system architecture. Feature extraction is done either through CNNs or a custom HEVC decoder. Dimensionality reduction using Stepwise regression. Frame elimination using S.A.D of motion estimation and compensation or HEVC features

3.1. Revision on HEVC video coding

In this section we are going to propose our HEVC feature extraction methods, and therefore we present a revision on HEVC video coding. HEVC is a relatively new video coding standard and the successor to the MPEG-4 (H.264) video coding standard. The ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) joined efforts on developing the HEVC standard [6]. The compression time for HEVC videos is twice as fast when compared to MPEG-4. Improvements in HEVC can be summarized into the following:

- More partitioning options, ranging from large to small partition sizes.
- Prediction modes and transform block sizes have more flexibility.
- Interpolation and de-blocking filters that are more advanced.
- Modes and motion vectors prediction and signaling that are more complex.
- Features that make parallel processing more efficient.

The most noticeable change in HEVC coding is unit partitioning. The macroblock structure found in MPEG-4 is replaced with a coding tree block (CTB) that includes coding units (LCUs). LCUs are then partitioned recursively into sub-coding units (sub-CUs) that can be of size 16x16 to 64x64 pixels. Sub-CUs are then divided to prediction units (PUs) of size 4x4 to 32x32 that contain the prediction information sent to the decoder. PUs are then split into transform units (TUs) of size 4x4 to 32x32 that are transformed using discrete cosine (DCT) or sine (DST) transformation. PUs can be skipped, intra-predicted or inter-predicted. Figure 2 shows the how all the units are partitioned and organized. We use all the partitioning and prediction information possible found in the output bitstream to produce low-level HEVC features.

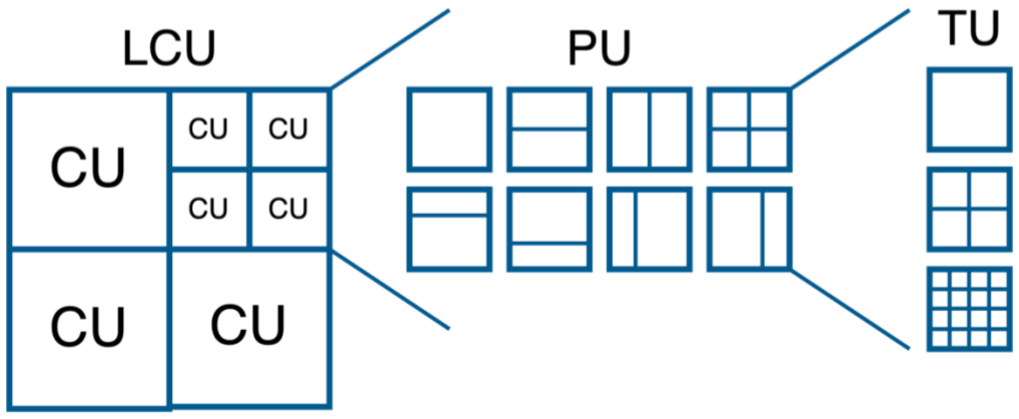

Figure 2: Coding Units partitioning in HEVC

3.2. Data Preprocessing

3.2.1. Proposed HEVC coding and feature extraction

This research presents the use of HEVC information found in the coded video bitstream into Deep Learning-based video summarization. HEVC features have been used in many applications, including encoding speedup and video transcoding [40], data embedding [41], double and triple compression detection [42] and saliency detection [43]. The videos are in MPEG format, which means that they need to be converted to

HEVC. This can be done using existing MPEG to HEVC encoders; However, the trade-off is that they tend to lower the video quality. Thus, a custom re-encoder is used to avoid degrading the video quality. The process is shown in Figure 3.

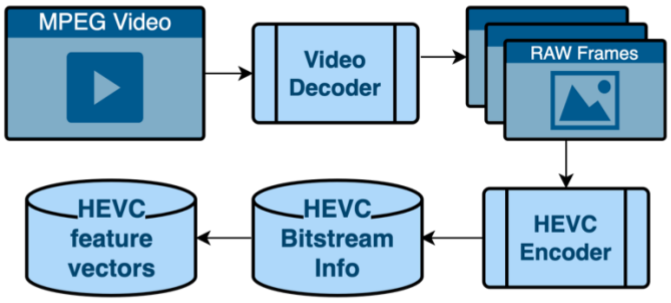

Figure 3: MPEG to HEVC video conversion process

The converted HEVC videos are then fed to a splitting script where the video is split into images (frames) to prepare them for feature extraction of HEVC features on frame-by-frame basis. The generated HEVC features are listed in Table I. MVD is for motion vector difference, SAD is for the sum of absolute differences, and CU is for coding unit.

Table I: Proposed HEVC feature set per frame from a custom HEVC decoder.

	Feature ID	Feature variable
Averaged per frame	1	Number of CU parts
	2	MVD bits per CU
	3	CU bits excluding MVD bits
	4	Percentage of intra CU parts
	5	Percentage of skipped CU parts
	6	Number of CUs with depth 0 (i.e 64x64)
	7	Number of parts with depth 1 (i.e 32x32)
	8	Number of CUs with depth 2 (i.e 16x16)
	9	Number of parts with depth 3 (i.e 8x8)
	10-18	Standard deviation of feature IDs 1-9 per frame
	19	Max CU depth per frame
	20	For CUs with depth > 0, $\log_2(\lvert sum\ of\ MVD \rvert)$?
	21	For CUs with depth = 0, $\log_2(\lvert sum\ of\ MVD \rvert)$?
Averaged per frame	22	Row-wise SAD of the CU prediction error
	23	Column-wise SAD of the CU prediction error
	24	Ratio of gradients (i.e feature 22 divided by feature 23) per CU
	25	Total distortion per CU as computed by the HEVC encoder
	26-29	Standard deviation of feature IDs 22-25 per frame
	30	Per frame: Summation of variance of the x and y components of all MVs
	31-47	Histogram of x-component of all MVs per frame (using 16 pins)
	48-64	Histogram of y-component of all MVs per frame (using 16 pins)

3.2.2. Splitting videos into frames

This process is done because at a later stage, for feature extractions, the CNNs used take 2D vectors as input. Doing so allows us to eliminate certain frames that are deemed to be not useful at the frame elimination stage later in this paper. Splitting videos into frames is carried out with a Python script using the cv2 library. A function takes in the input directory containing all videos and the desired output directory.

3.2.3. Datasets

Experimentation is done over the OVP and VSUMM [44] datasets. The VSUMM dataset contains 50 videos from YouTube, across several genres (news, sports, commercials, tv-shows, etc.) and have a duration of 1-10 mins at 30 fps. The OVP (Open Video Project) dataset has 50 videos from Open Video Project in MPEG-1 format at 30 fps. The videos were distributed among several genres (documentary, educational, ephemeral, historical, and lecture) and have a duration of 1-4 minutes.

As for the ground truths, for every video in both datasets there is a ground truth of the selected frames derived from 5 different users, for a total of 250 user summaries for each of the datasets. For a certain video, the ground truth is a 1-Dimensional vector of ones and zeros. A one means that this frame at that index was selected for the summary by a user. The ground truth for a video has the following form: [0,0,1,0,1,1,0,...] which, for example, indicate that frames 3,5,6 were selected by a user for the final summary.

3.2.4. CNN feature extraction

For feature extraction, we used built-in MATLAB CNN plug-ins for AlexNet [45], IRv2 [46] and VGG16 [47]. These models are trained on millions of images under thousands of categories. GoogleNet features were publicly available for both datasets at [44]. The splitting and feature extraction processes are illustrated in Figure 4.

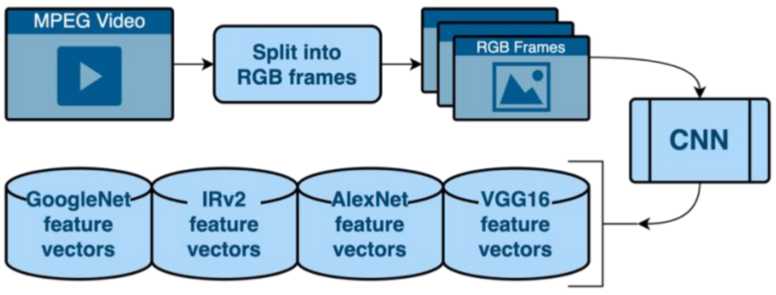

Figure 4: Feature extraction process from pre-trained CNNs

For all of the 4 CNNs, a high-level loop goes through the videos' directories and another lower-level loop goes through the frames. Then, the feature extraction starts, producing a feature vector for every frame. The feature vector values are normalized from 0 to 1. Next, all frame feature vectors belonging to a video are combined to produce a 2D feature vector for the entire video, with each row representing the feature vector of a frame. All the 2D feature vectors are then stored in an HD5F file to be used later. HEVC features extracted in Section 3.2.1 (with feature vector length of 64) are also added to the HD5F file. Every CNN has its own set of features and the produced lengths for each are in Table II and the final HDF5 has the structure illustrated in Figure 5:

Table II: Input and feature vector sizes for networks used for feature extraction

Network	GoogleNet	AlexNet	IRv2	VGG16	HEVC
Input size	224x224	227x227	299x299	224x224	Original
Features	1024	4096	1536	4096	64

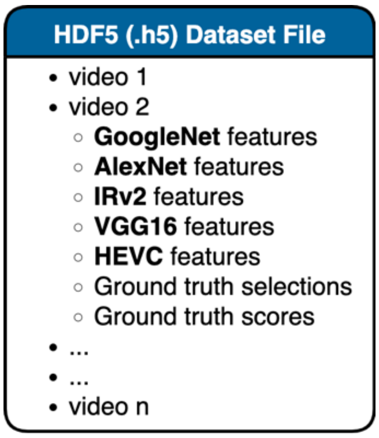

Figure 5: The structure of the HDF5 file with CNN and HEVC features

3.3. Elimination of Similar Frames

Frame elimination is needed to reduce the input size and the amount of data crunched into our models. If all frames within a video are to be examined for selection then the summarization process can be extremely slow and precious time and computational power is wasted on redundant or similar frames. The videos of both the OVP and VSUMM datasets are all at 30 fps, therefore, we decided to take the first and 15th frame of every second. So, if the video is 60 seconds long, we should have 120 frames for it. This criterion is followed in [22]'s paper and we are continuing on it.

3.3.1. Frame elimination using SIFT Flow algorithm

The use of SIFT Flow [39] algorithm was proposed by [22] for eliminating similar frames. The frames are converted to SIFT images where every pixel is replaced with a 128-D SIFT descriptor. Rather than dealing with raw pixels, the flow vectors are calculated by matching SIFT descriptors across subsequent images. On the SIFT image, a Gaussian filter is used, and a four-level pyramid is built. Each level reduces the image's size to half of what it was at the previous level. As a result, the top level of the picture comprises coarse features. Descriptors are matched in order of coarseness to fineness. The loopy belief propagation network, which operates by reducing the energy function, is used to find the ideal flow vector. The magnitude of flow vectors at each pixel point in a frame is added to determine the displacement vector for the entire frame. Local thresholding of the magnitude of displacement vectors between successive frames is used to pick candidate frames from the resulting collection of frames.

3.3.2. Revision on motion estimation and compensation

Consecutive frames might contain the same objects, either still or moving. Motion estimation inspects the movement of objects between a sequence of images to attain vectors that represent the estimated motion in x and y directions. Motion compensation uses the motion vectors obtained to perform data compression. In video coding, motion estimation and compensation are effective methods for removing temporal redundancy brought on by significant relation between successive frames. A simple example of motion estimation and compensation is shown in Figure 6.

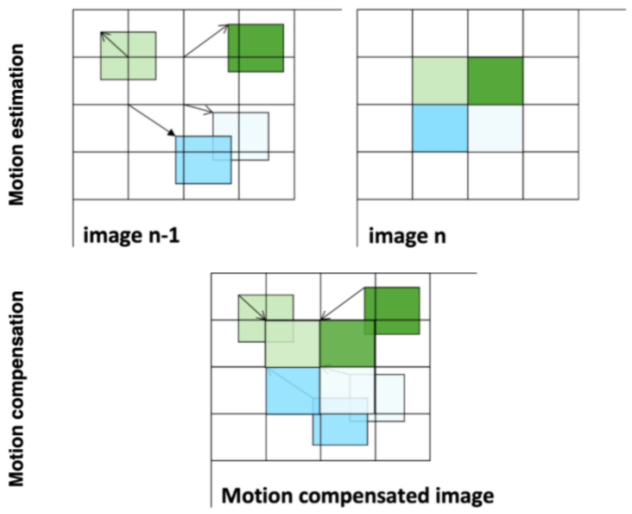

Figure 6: Simple illustration of motion estimation and compensation

3.3.3. Proposed frame elimination using motion estimation and compensation

The minimum sum of absolute differences of the motion estimation and compensation is used for eliminating similar frames. For every 2 consecutive frames, we calculated the motion vectors using optical flow with a noise threshold of 0.009. Then, the X and Y components of the motion vector are divided by 2 and motion compensation is applied. Then, we find the sum of absolute differences between the second frame and the motion compensated frame and add it to a list of sums. Then we compare the sums against a pre-defined threshold that was determined through experimentation, say 0.35, and a sum smaller than the threshold means a similar frame is detected and gets marked for deletion. Figure 7 illustrates the process of the proposed method.

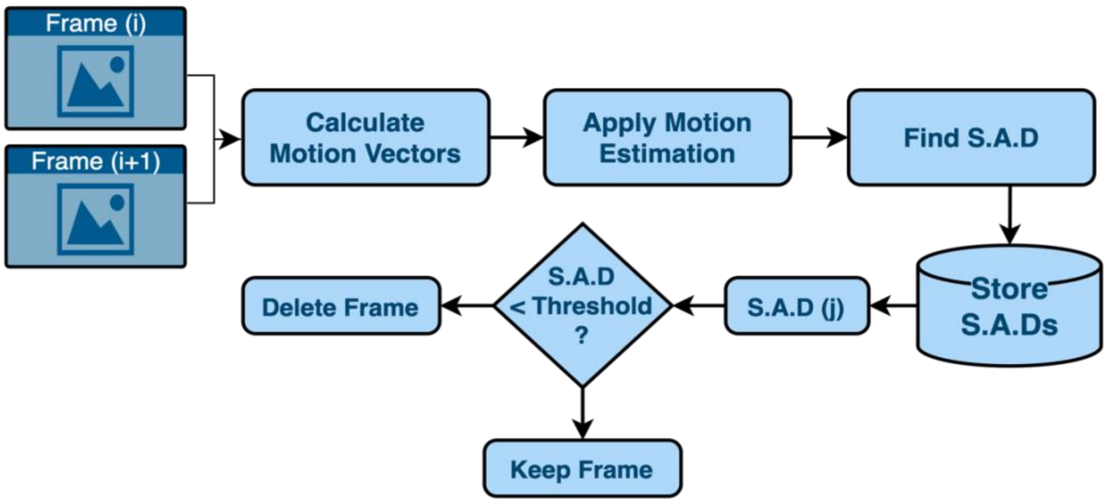

Figure 7: Illustration diagram of the proposed frame elimination method with ME+MC

3.3.4. Proposed frame elimination based on HEVC features

In the proposed list of HEVC feature variables, there exists a sub-set of variables that indicates the low temporal activity of video frames. Such low temporal activity indicates that the current frame is similar to its previous frames, and therefore, can be eliminated. In this solution, we use the sum of the following HEVC feature variables: [1 to 7, 22 to 25, 30], to produce a temporal activity index; The lower the index, the lower the temporal activity. The sub-set feature variables are listed in Table III, along with brief descriptions of each. The removal or elimination of frames that are deemed unnecessary is important to make sure that they do not contribute to the training process. Hence, only distinct frames are fed into the training model.

Table III: List of HEVC feature subset variables used to produce a temporal activity index

Feature ID	Feature variable (averaged over a frame)
1	Number of CU parts
2	MVD bits per CU
3	CU bits excluding MVD bits
5	Percentage of skipped CU parts
6	Number of CUs with depth 0 (i.e 64x64)
7	Number of parts with depth 1 (i.e 32x32)
22	Row-wise SAD of the CU prediction error
23	Column-wise SAD of the CU prediction error
24	Ratio of gradients (i.e feature 22 divided by feature 23) per CU
25	Total distortion per CU as computed by the HEVC encoder
30	Summation of variance of the x and y components of all MVs

In the implementation process, the video dataset was split into training and test data, as explained in the experimental results section. For each split of the dataset, we used video frames with ground truth zero in the training data to determine the average value of the features listed in Table II. Video frames with ground truth zero are those that are not part of the video summary. The averages were then summed to compute the temporal activity index. To vary the value of the mentioned index, the sum of the standard deviations of the feature variables can be added, as illustrated in Equation 1.

$$TAI = \sum_{i=1}^{I} \bar{f_i} + c \sum_{i=1}^{I} \sigma_{f_i} \tag{1}$$

where *TAI* is the temporal activity index, *fi* is feature *i* from Table III, and *i* ranges from 1 to 11, which is the total number of features used. Constant *c* ranges from 0 to 1 and can be used to vary the value of the temporal activity index. In this work, it is set to 0.35 using empirical testing. Consequently, a video frame from the test data was eliminated according to the Boolean condition presented in Equation 2.

$$Eliminate\ frame_j = \begin{cases} False, & if\ \sum_{i=1}^{I} \bar{f}_{i,j} > TAI \\ True, & if\ \sum_{i=1}^{I} \bar{f}_{i,j} \leq TAI \end{cases} \tag{2}$$

where the frame at index *j* belongs to the test video set and *fi,j* is feature *i* from Table III of the test frame at index *j*.

3.4. Feature Space Dimensionality Reduction

We employ CNN features from popular pre-trained CNN models, namely, GoogleNet, Inception-ResNet-V2 (IRv2), AlexNet and VGG16 as done previously by [22]. When taking all four CNN features the total length of the feature vector for every frame is 10,752 features, or 10,816 with HEVC features.

3.4.1. Sparse Autoencoder

A Sparse Autoencoder (SAE) was used by [22] to reduce the dimension of the feature space to 500 features. The SAE architecture is the following: An input layer takes in the entirety of the 10,752-feature vector for a given frame with a node representing each feature, a latent layer with the reduced feature space and an output layer with a feature vector of length 500. An SAE adds a sparsity penalty. Every feature is compared against a weight decay penalty, which is one thousand the value of the sparsity penalty. This means that if the autoencoder deems the feature is not as useful as other, it does not make it into the reduced space. Figure 8 shows a simple representation of an SAE.

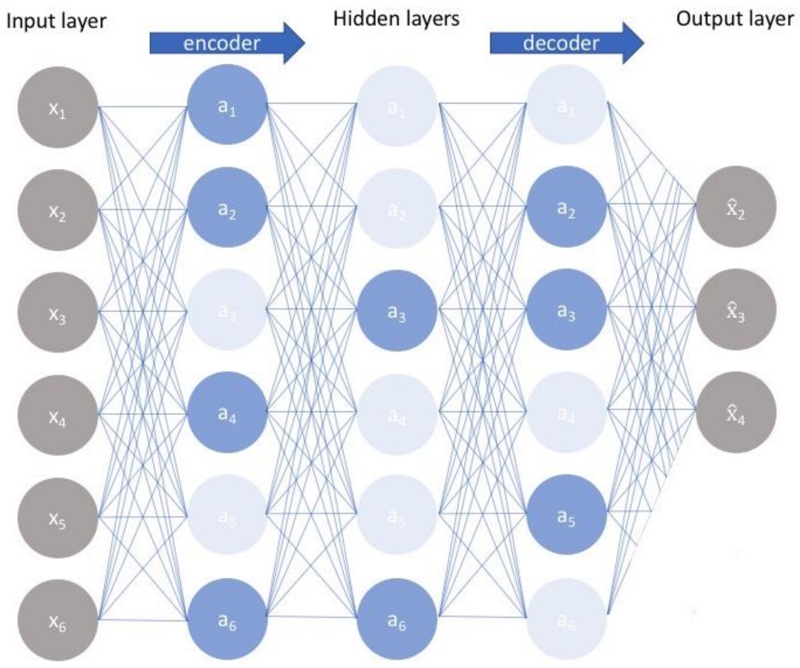

Figure 8: Simple representation of sparse autoencoders

3.4.2. Stepwise Regression

Dimensionality reduction methods are usually forward, starting with one feature and adding up features to reach the optimal model; Or backward, starting with all features and dropping one feature at a time to reach the optimal model. Stepwise regression (SW) is a variable selection method that can be used for dimensionality reduction. SW combines forward and backwards methods, because at each iteration a feature could be dropped or added. Figure 9 describes the setup used with SW.

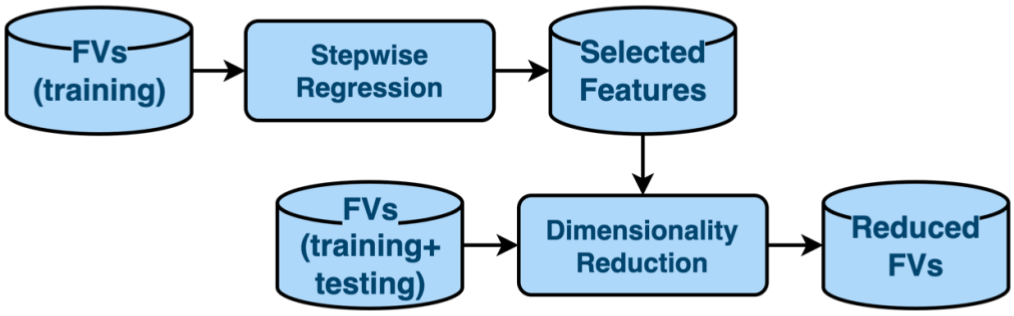

Figure 9: Process of dimensionality reduction using Stepwise regression

For a set of features $x_1, x_2, \ldots, x_k$. F_{in} is the F-random feature for the feature to be added to the model, while F_{out} is for the feature to be dropped from the model.

The following are the step for stepwise regression:

1- Form 1-itemsets from all features to produce a 1-feature model(s):

$$h(x) = \theta_0 + \theta_1 x_1 \tag{3}$$

where $h(x)$ is the hypothesis that the added features are needed for the classification task. x_1 is one of the features that gives the *highest* F-score. f_1 is the statistic of x_1 and given by the following formula:

$$f_1 = \frac{SS_R(\theta_2|\theta_1\theta_0)}{MS_E(x_2, x_1)} \tag{4}$$

where SS_R is the regression sum square error and MS_E is the mean square error.

2- From the equation obtained above for the 1-itemset feature, we examine the rest of the $k-1$ features that can, when combined with $h(x)$, produce higher hypothesis than $h(x)$ by itself. We add x_2 if its f_2 is greater than F_{in} and get the following:

$$f_2 = \frac{SS_R(\theta_1|\theta_2\theta_0)}{MS_E(x_1, x_2)} \tag{5}$$

$$h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \tag{6}$$

After adding x_2, we check if x_1 needs to be removed by comparing f_1 to the new F_{out}. If f_1 is lesser, x_1 gets dropped

3- The remaining $k-2$ features are examined to obtain x_3 and get the following hypothesis:

$$h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \tag{7}$$

The algorithm continues until there are no features to add or drop. Figure 9 briefly explains the feature vectors reduction using stepwise regression process.

3.5. Classification

3.5.1. Random Forests

An approach to supervised learning is Random Forests. An ensemble of decision trees, which are usually trained using the "bagging" method, are combined to form a "forest." The fundamental tenet of the bagging approach is that the output is improved by combining several learning models. Classification and regression problems, which make up the majority of modern Machine Learning systems, may be solved using Random Forests. Random Forests increases the model's variability when creating the decision trees. Instead of looking for the best feature when splitting a node, it seeks for the best feature from a random selection of features. There is a lot of diversity as a consequence, which results in a better model. By employing random thresholds for every feature, Random Forests add more randomness to the creation of trees. Figure 10 shows a representation of a Random Forests with 2 trees.

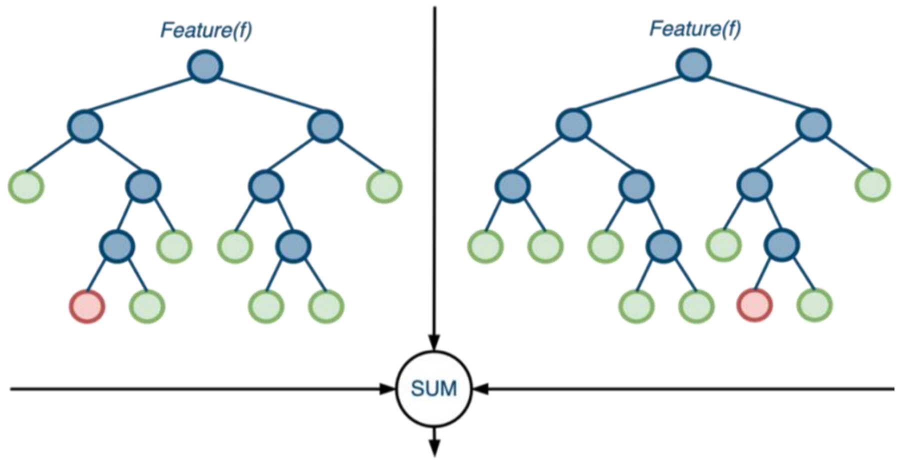

Figure 10: Simple view of Random Forests classifier

3.5.2. Classification setup

In our implementation, we use a threshold of 0.9 to retain the features with a value above the threshold. If none of the features are above the threshold, then use all of them, else, use the retained ones only. We specify that the forest generates 128 trees for training over the selected features. We then save the predicted labels and calculate a few performance measures to help in quantifying the obtained results.

Chapter 4. Experimental Results

4.1. Evaluation Criteria

The used quantitative performance metrics are the following:

4.1.1. Positive predictive values (PPV)

The percentage of true positive predictions over all positive predictions. Where P_T is the true positive predictions, and P is all positive predictions.

$$PPV = {}^{P_T}/_{P} \tag{8}$$

4.1.2. Sensitivity (S)

The percentage of true positive predictions over the users ground truth (at the same frame indices). Where P_T is the true positive predictions, and P_u is the user selected frames and their indices.

$$S = {}^{P_T}/_{P_u} \tag{9}$$

4.1.3. F-measure

The F-measure (or F-score) is the harmonic mean of the precision and recall scores. It provides a combined view when either of these scores are not enough to describe the imbalanced classification problem like the frame selection problem we have where most frames are not selected and only a few are selected.

$$F - score = {}^{2 \times PPV \times S}/_{PPV + S} \tag{10}$$

4.2. Experimental Results

After describing our data preprocessing, relevant components, and setup, we proceeded with experimenting our proposed methods. The proposed dimensionality reduction method using SW was compared with the SAE as in [22]. In addition, the proposed elimination of similar frame methods using ME+MC and low-level HEVC features, were compared against the SIFT flow algorithm used in [22]. Finally, the proposed HEVC feature set was tested against the features from well-known CNN models.

The trial runs reported have the following setup: Each run consists of 5 non-overlapping folds. In other words, the data (videos) are split in an 80%-20% fashion for training and testing, respectively. For single CNN runs, the feature vector lengths are mentioned in Section-III. For Multi-CNN runs, the feature vector length is 10,752, which is a combination of the 4 CNNs (GoogleNet: 1024 features, AlexNet: 4096, IRv2: 1536 features, VGG16: 4096 features). For Multi-CNN & HEVC runs, the feature vector length is 10,816, which is a combination of the 4 CNNs and the HEVC feature set (GoogleNet: 1024 features, AlexNet: 4096, IRv2: 1536 features, VGG16: 4096 features, HEVC: 64). The results reported for each trial run are the average results for of the 5 testing folds. The metrics reported are PPV, Sensitivity and F-score. Confusion matrices are shown and discussed in subsection 4.2.7. Further in-depth discussion on the elapsed run times and performance for the models can be found later in subsection 4.3. The experiments were conducted on a PC provided by the American University of Sharjah with Intel i7 (7th gen) CPU, 16 GB of RAM and NVIDIA GTX 1070 GPU.

4.2.1. OVP dataset – Single-CNN results

Table IV has the experimental results for the OVP dataset using the Random Forests classifier with features derived from GoogleNet, AlexNet, IRv2, and VGG16, along with the results from using the proposed HEVC feature set.

In Table IV, the accuracy of the summarization with different methods is presented. The best summary results are obtained when using the proposed HEVC feature set with SW and HEVC-based frame elimination. For features generated from the pertained CNNs, we experimented with dimensionality reduction using the existing work of [22], which is based on SAEs, and using the proposed SW. Higher detection accuracies have been reported for the latter. The results also indicate that eliminating replicated frames based on HEVC features results in a higher detection accuracy compared to the use of the SIFT-based algorithm. The results for features with IRv2 and VGG16 features peaked with the use of the proposed ME+MC frame elimination solution.

Table IV: Experimental results on the OVP dataset - Single-CNN

Feature Sets	Feature Reduction	Frame Elimination	Metrics			Elapsed times (seconds)
			PPV	S	Fs	
HEVC (Proposed)	None	SIFT	0.59	0.71	0.83	*44.6*
		ME+MC	0.71	0.85	0.92	*41.3*
		HEVC	**0.87**	**0.86**	**0.93**	*41.8*

GoogleNet	SAE [22]	SIFT [22]	0.50	0.93	0.61	*803.2*
	SW	SIFT	0.55	0.91	0.86	*112.4*
		ME+MC	0.63	0.95	0.87	*46.8*
		HEVC	**0.59**	**0.96**	**0.88**	*94.0*
AlexNet	SAE [22]	SIFT [22]	0.70	0.89	0.78	*2653.3*
	SW	SIFT	0.54	0.93	0.86	*314.1*
		ME+MC	0.62	0.95	0.87	*93.9*
		HEVC	**0.59**	**0.97**	**0.88**	*262.6*
IRv2	SAE [22]	SIFT [22]	0.75	0.84	0.79	*1096.5*
	SW	SIFT	0.54	0.93	0.86	*129.3*
		ME+MC	**0.63**	**0.96**	**0.88**	*55.4*
		HEVC	0.59	0.94	0.87	*108.1*
VGG16	SAE [22]	SIFT [22]	0.68	0.61	0.64	*2615.8*
	SW	SIFT	0.54	0.71	0.85	*219.2*
		ME+MC	**0.62**	**0.74**	**0.87**	*79.5*
		HEVC	0.59	0.76	0.87	*183.3*

4.2.2. OVP dataset – Multi-CNN results

In Table V, we present the detection results for the OVP dataset after combining all feature sets. In one set of experiments, all CNN features were combined (Multi-CNN) and compared when using SAE against SW along with SIFT versus ME+MC and HEVC based elimination, and in the other set, we combined the proposed HEVC features with all CNN features (Multi-CNN and HEVC).

Table V: Experimental results on the OVP dataset - Multi-CNN

Feature Sets	Feature Reduction	Frame Elimination	Metrics			Elapsed times (seconds)
			PPV	S	Fs	
Multi-CNN	SAE [22]	SIFT [22]	0.78	0.86	0.82	*6343.3*
	SW	SIFT	0.55	0.94	0.87	*1690.3*
		ME+MC	0.62	0.96	0.90	*1055.6*
		HEVC	**0.59**	**0.97**	**0.93**	*1229.3*
Multi-CNN & HEVC (proposed)	SW	SIFT	0.54	0.93	0.96	*1543.6*
		ME+MC	0.65	0.95	0.97	*812.7*
		HEVC	**0.83**	**0.96**	**0.98**	*836.8*

The Multi-CNN model performed better than all previous models with single CNN features by up to a 6% increase in performance across all performance metrics. This shows how the SW can retain the best features from multiple CNNs and attain satisfactory results. The detection accuracy resulting from Multi-CNN surpasses the use of individual CNNs. More noticeably, the use of Multi-CNN and HEVC resulted in accuracy of 0.98 F-score.

4.2.3. OVP dataset versus existing work

Figure 11 shows how our best model on the OVP dataset, with Multi-CNN and HEVC features with HEVC-based frame elimination exceeds state-of-the-art [22], VISCOM [48], VRHDPS [49] and VSUMM [50] across all performance metrics.

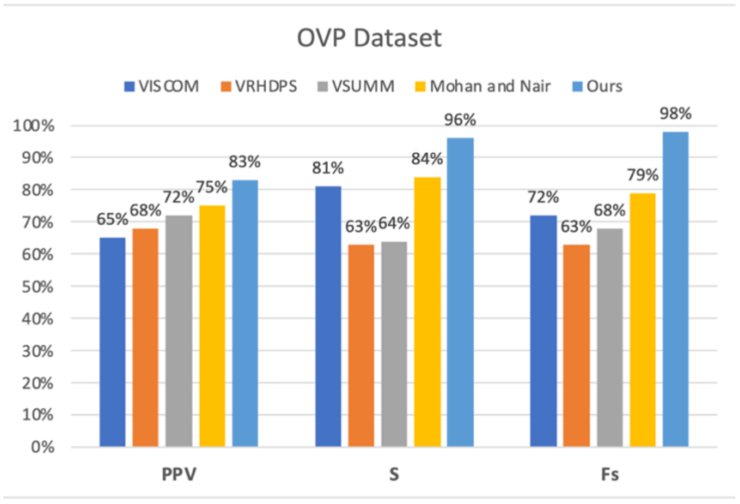

Figure 11: Performance metrics of our top performing model (Multi-CNN & HEVC and HEVC-based frame elimination) on the OVP dataset compared with existing works in the literature

4.2.4. VSUMM dataset – Single-CNN results

Similar to Table IV, Table VI reports the results for the VSUMM dataset. Similar to the conclusions drawn from the results in Table V, when using the proposed HEVC feature set, our solution surpasses the CNN models when combined with SW and the HEVC feature-based frame elimination method. Additionally, the proposed SW method for dimensionality reduction achieved up to a 17% increase in performance compared to SAE across all performance metrics. Further improvement can be observed when using our proposed elimination of similar frame methods using ME+MC and HEVC features with up to 5% and 12% performance improvement, respectively, when compared to SIFT Flow across all performance metrics.

Table VI: Experimental results on the VSUMM dataset - Single-CNN

Feature Sets	Feature Reduction	Frame Elimination	Metrics			Elapsed times (seconds)
			PPV	S	Fs	
HEVC (proposed)	None	SIFT	0.54	0.52	0.51	*58.1*
		ME+MC	0.71	0.82	0.75	*53.7*
		HEVC	**0.88**	**0.86**	**0.86**	*54.4*
GoogleNet	SAE [22]	SIFT [22]	0.61	0.81	0.69	*1044.2*
	SW	SIFT	0.56	0.82	0.76	*147.7*
		ME+MC	0.70	0.82	0.76	*60.8*
		HEVC	**0.68**	**0.84**	**0.78**	*122.2*
AlexNet	SAE [22]	SIFT [22]	0.66	0.86	0.74	*3449.3*
	SW	SIFT	0.71	0.80	0.76	*370.8*
		ME+MC	0.71	0.81	0.76	*122.1*
		HEVC	**0.69**	**0.83**	**0.78**	*341.3*
IRv2	SAE [22]	SIFT [22]	0.71	0.78	0.75	*1425.5*
	SW	SIFT	0.69	0.82	0.75	*176.5*
		ME+MC	0.68	0.82	0.75	*72.1*
		HEVC	**0.71**	**0.83**	**0.77**	*140.5*
VGG16	SAE [22]	SIFT [22]	0.67	0.73	0.70	*3400.5*
	SW	SIFT	0.69	0.83	0.75	*249.3*
		ME+MC	0.69	0.83	0.76	*103.4*
		HEVC	**0.68**	**0.85**	**0.77**	*238.3*

4.2.5. VSUMM dataset – Multi-CNN results

Table VII shows the performance of our proposed models when using Multi-CNN features only and when using Multi-CNN features with HEVC features on the VSUMM dataset under the same run conditions as the previous runs. Again, the detection accuracy resulting from Multi-CNN surpasses the use of individual CNNs, as reported in Table VII. More noticeably, the use of Multi-CNN and HEVC resulted in an outstanding detection accuracy score, as indicated by the 0.98 F-score. These results are similar to the findings in Table V and provide further proof of the effectiveness of the proposed methods against works in the literature. It is important to look at the run times for each of models as our models show significant gains in faster run times while still preceding larger and heavier models.

Table VII: Experimental results on the VSUMM dataset - Multi-CNN

Feature Sets	Feature Reduction	Frame Elimination	Metrics			Elapsed times (seconds)
			PPV	S	Fs	
Multi-CNN	SAE [22]	SIFT [22]	0.80	0.84	0.83	*8246.3*
	SW	SIFT [22]	0.78	0.88	0.91	*2197.0*
		ME+MC	0.80	0.89	0.93	*1372.3*
		HEVC	**0.82**	**0.92**	**0.96**	*1598.1*
Multi-CNN & HEVC (proposed)	SW	SIFT [22]	0.80	0.91	0.94	*2006.7*
		ME+MC	0.81	0.92	0.96	*1056.5*
		HEVC	**0.85**	**0.94**	**0.98**	*1087.9*

The Multi-CNN model performed better than all previous models with single CNN features by up to a 7% increase in performance across all metrics. When combining the HEVC features with the Multi-CNN features, the models showed up to a 5% score increase in performance across all metrics. Our best performing model (Multi-CNN & HEVC with HEVC-based elimination) is compared with existing works in the following section.

4.2.6. VSUMM dataset versus existing work

Figure 12 shows our best solution on the VSUMM dataset, which uses Multi-CNN and HEVC features and HEVC-based frame elimination against the state-of-the-art [22], VISCOM [48], VRHDPS [49] and VSUMM [50]. Similar to the results reported for the OVP dataset, our proposed solution for the VSUMM dataset surpasses existing work.

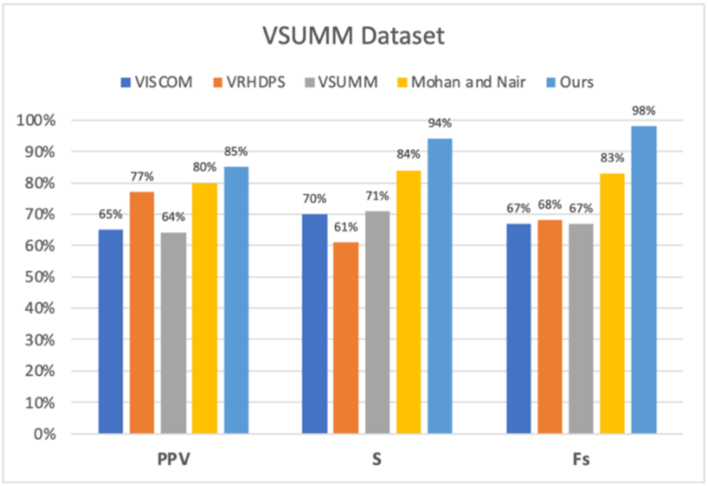

Figure 12: Performance metrics of our top performing model (Multi-CNN & HEVC and HEVC-based frame elimination) on the VSUMM dataset compared with existing works in the literature

4.2.7. Confusion matrices

Figure 13 has the confusion matrices for our best performing models on both the OVP and VSUMM datasets. The best performing model on both datasets is Multi-CNN & HEVC with SW for feature space reduction and HEVC-based frame elimination (i.e Our best models with the highest scores in Tables V and VII). The OVP confusion matrix (on the left) translates to a sensitivity score of 0.96 and the VSUMM confusion matrix (on the right) translates to a sensitivity score of 0.94. Both models have an F-score of 0.98, exceeding existing works in the literature.

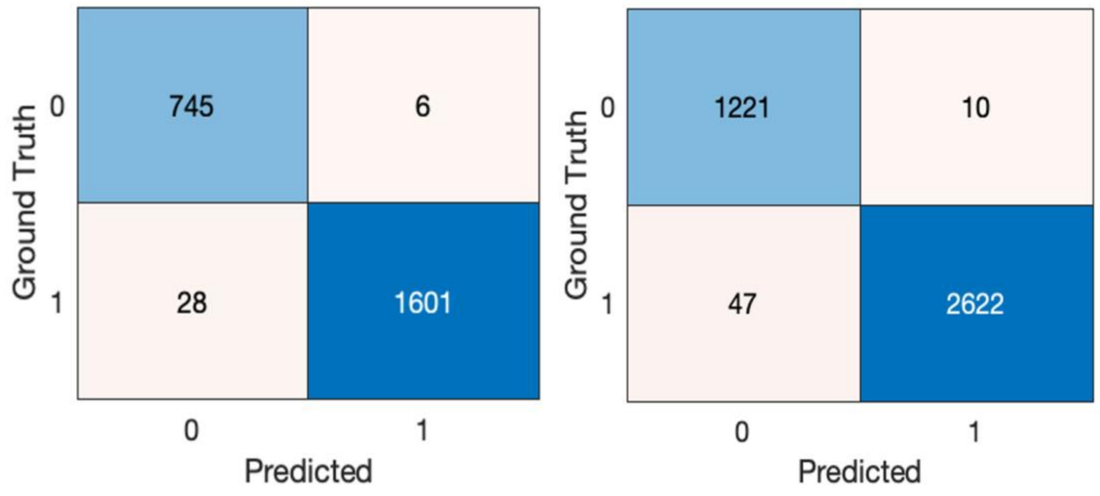

Figure 13: Confusion matrices of models "Multi-CNN & HEVC" with SW feature space reduction and HEVC-based frame elimination. OVP (left), VSUMM (right).

4.3. Discussion of Results

4.3.1. Proposed work versus existing work

Our results show an overall improvement over previous works in the literature. The advantages and limitations of the proposed solution in comparison with existing work are summarized as follows. The proposed 64 HEVC feature variables are precise and concise compared to CNN-generated features with significantly higher dimensionality. In comparison to the reviewed work, VISCOM described video frames using novel color co-occurrence matrices [48]. VSUMM extracted video frames attributes based on color histograms and line profiles [50]. VRHDPS used the SIFT descriptors [49] and [22] used a novel combination of CNN features. In this work it was shown that the HEVC feature set contains rich video descriptors based on recursive splitting of coding units. These descriptors provide rich information about the spatio-temporal video content and therefore provide an excellent choice for the task at hand.

Moreover, in terms of dimensionality resolution, our solution used SW, which is significantly faster than the use of SAE, and yet retains enough features that are the best representative features for training and classification. The use of SAE was used successfully for directionality reduction as reported in [22].

The proposed HEVC-based frame elimination avoids the high complexity of optical flow based in SIFT-descriptors which was used for frame elimination as reported in [22]. Black frames and shot boundaries were eliminated in VRHDPS [49] as they were deemed useless. In VISCOM [48], monotonic frames are eliminated based on normalized summations of squared distances between frames.

In this work, combining the proposed solutions together resulted in an average F-score of 0.93 and 0.86. Further combination of the proposed solution with multi-CNN resulted in an outstanding F-score of 0.98 for both the OVP and VSUMM datasets. However, a drawback of combining the proposed solution with Multi-CNNs is that it can be computationally intensive on some computer systems. This limitation can be overcome by simply relying on the HEVC feature set alone, as it proved to identify key-frames more accurately in comparison to existing work. On the other hand, it was reported in VRHDPS [49] that their solution does not require any iteration in the clustering process, rendering it an efficient algorithm.

4.3.2. Elapsed run times

Performance metrics are one side of the coin, while the time it takes each model to finish running is the other side. All our proposed models are consistently at least 5x and up to 15x faster than the existing work in [22]. The slower times found in [22]'s models are mainly due to the use of SAE for reducing the dimensionality of the feature space. SAE is very high in complexity, and the bigger the feature set is, the slower it gets to reduce all the features being fed. The use of optical flow SIFT-descriptors for frame elimination also slows their setups even further, similarly, due to the high complexity of having to construct a 128-imentional descriptor for every pixel in the image or frame. For instance, in Table IV, when the feature space is moderately small, like with GoogleNet features on OVP dataset, [22]'s model with SAE feature reduction and

SIFT-based frame elimination scores 0.61 with a run time of 803.2 seconds. Our comparable model on GoogleNet features with SW and ME+MC or HEVC score 0.87 and 0.88 with run times of 46.8 and 94.0 seconds, respectively. This trend continues across all comparable models in Tables IV, V, VI, and VII.

While in our model it seems like using ME+MC for frame elimination is considerably faster than using HEVC-based frame elimination, the later always scores higher and the bigger the feature space is the smaller the gap gets in run times between these two models. For instance, in Table VI, using IRv2 features on the VSUMM dataset with SW for feature reduction we get scores of 0.75 and 0.77 and run times of 72.1 and 140.5 seconds, respectively, with ME+MC frame elimination and HEVC-based elimination. The gap between the two narrows down as the feature space gets bigger. For instance, in Table V, using Multi-CNN & HEVC features on the OVP dataset with SW for feature reduction we get scores of 0.97 and 0.98 with run times of 812.7 and 836.8 seconds, respectively, with ME+MC frame elimination and HEVC-based elimination.

Chapter 5. Conclusion

5.1. Summary

With the surge of the Internet and surveillance footage, the need for video summarization is crucial. Video summarization can facilitate video retrieval, anomaly detection and activity monitoring. This research focused on the key-frame detection technique for its wide use in the literature. The proposed methodology used CNNs and Random Forests. We proposed a feature set extracted from HEVC-coded videos. Eliminating duplicate or similar video frames was performed based on a subset of the proposed HEVC features, it was also performed based on the sum of absolute differences resulting from motion estimation and motion compensation. The dimensionality reduction of the feature vectors was based on stepwise regression. Using Random Forests classification, it is shown that by combining the proposed solution with Multi-CNN features, an average PPV, Sensitivity and F-score of 0.83, 0.96 and 0.98 are reported for the OVP dataset and an average of 0.85, 0.94 and 0.98 are reported for the VSUMM dataset, respectively.

5.2. Future work

In the future, this research can be expanded to include more datasets and experiment with different methods for frame elimination. A number of classifiers can also be explored, namely 1-Dimensional CNNs and LSTMs. 1D-CNNs and LSTMs have been long used in the literature with time-series based applications, in this case, videos which are timed sequences of frames. LSTM networks are known for their ability to take temporal dependencies into account when dealing with video frames. 1D-CNNs have also recently emerged as a distilled version of traditional CNNs, which are 2-Dimensional. 1D-CNNs can usually be used with time-series applications as well, just like LSTM. Further studies and experimentation can be done with these two networks in hopes to further optimize and achieve better results.

References

[1] M. Basavarajaiah and P. Sharma, “Survey of Compressed Domain Video Summarization Techniques,” *ACM Comput. Surv.*, vol. 52, no. 6, pp. 1–29, Nov. 2020, doi: 10.1145/3355398.

[2] T. Subba, B. Roy, and A. Pradhan, “A Study On ‘VIDEO SUMMARIZATION’ Tanuja Subba1, Bijoyeta Roy2, Ashis Pradhan,” *Int. J. Adv. Res. Comput. Commun. Eng.*, Jun. 2016, doi: 10.17148/IJARCCE.2016.56164.

[3] L. Van Der Maaten, E. Postma, J. Van den Herik, and others, “Dimensionality reduction: a comparative,” *J Mach Learn Res*, vol. 10, no. 66–71, p. 13, 2009.

[4] C. Szegedy *et al.*, “Going Deeper with Convolutions,” *ArXiv14094842 Cs*, Sep. 2014, Accessed: Nov. 28, 2021. [Online]. Available: http://arxiv.org/abs/1409.4842

[5] L. Breiman, “Random Forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[6] V. Sze, M. Budagavi, and G. J. Sullivan, Eds., *High Efficiency Video Coding (HEVC): Algorithms and Architectures*. Cham: Springer International Publishing, 2014. doi: 10.1007/978-3-319-06895-4.

[7] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, “Video Summarization with Long Short-Term Memory,” in *Computer Vision – ECCV 2016*, vol. 9911, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 766–782. doi: 10.1007/978-3-319-46478-7_47.

[8] B. Zhao, X. Li, and X. Lu, “HSA-RNN: Hierarchical Structure-Adaptive RNN for Video Summarization,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, Jun. 2018, pp. 7405–7414. doi: 10.1109/CVPR.2018.00773.

[9] J. Wang, W. Wang, Z. Wang, L. Wang, D. Feng, and T. Tan, “Stacked Memory Network for Video Summarization,” in *Proceedings of the 27th ACM International Conference on Multimedia*, Nice France, Oct. 2019, pp. 836–844. doi: 10.1145/3343031.3350992.

[10] T. Hussain, K. Muhammad, A. Ullah, Z. Cao, S. W. Baik, and V. H. C. de Albuquerque, “Cloud-Assisted Multiview Video Summarization Using CNN and Bidirectional LSTM,” *IEEE Trans. Ind. Inform.*, vol. 16, no. 1, pp. 77–86, Jan. 2020, doi: 10.1109/TII.2019.2929228.

[11] Z. Ji, K. Xiong, Y. Pang, and X. Li, “Video Summarization With Attention-Based Encoder–Decoder Networks,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1709–1717, Jun. 2020, doi: 10.1109/TCSVT.2019.2904996.

[12] H. Fu and H. Wang, “Self-attention binary neural tree for video summarization,” *Pattern Recognit. Lett.*, vol. 143, pp. 19–26, Mar. 2021, doi: 10.1016/j.patrec.2020.12.016.

[13] B. Zhao, X. Li, and X. Lu, “TTH-RNN: Tensor-Train Hierarchical Recurrent Neural Network for Video Summarization,” *IEEE Trans. Ind. Electron.*, vol. 68, no. 4, pp. 3629–3637, Apr. 2021, doi: 10.1109/TIE.2020.2979573.

[14] M. Otani, Y. Nakashima, E. Rahtu, J. Heikkilä, and N. Yokoya, “Video Summarization Using Deep Semantic Features,” in *Computer Vision – ACCV 2016*, vol. 10115, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 361–377. doi: 10.1007/978-3-319-54193-8_23.

[15] Y. Jung, D. Cho, D. Kim, S. Woo, and I. S. Kweon, “Discriminative Feature Learning for Unsupervised Video Summarization,” *ArXiv181109791 Cs*, Nov. 2018, Accessed: Nov. 15, 2021. [Online]. Available: http://arxiv.org/abs/1811.09791

[16] M. Rochan, L. Ye, and Y. Wang, “Video Summarization Using Fully Convolutional Sequence Networks,” *ArXiv180510538 Cs*, Aug. 2018, Accessed: Nov. 15, 2021. [Online]. Available: http://arxiv.org/abs/1805.10538

[17] K. Zhou, Y. Qiao, and T. Xiang, “Deep Reinforcement Learning for Unsupervised Video Summarization with Diversity-Representativeness Reward,” *ArXiv180100054 Cs*, Feb. 2018, Accessed: Nov. 15, 2021. [Online]. Available: http://arxiv.org/abs/1801.00054

[18] R. Agyeman, R. Muhammad, and G. S. Choi, “Soccer Video Summarization Using Deep Learning,” in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, San Jose, CA, USA, Mar. 2019, pp. 270–273. doi: 10.1109/MIPR.2019.00055.

[19] S. Zhong, J. Wu, and J. Jiang, “Video summarization via spatio-temporal deep architecture,” *Neurocomputing*, vol. 332, pp. 224–235, Mar. 2019, doi: 10.1016/j.neucom.2018.12.040.

[20] T. Liu, Q. Meng, A. Vlontzos, J. Tan, D. Rueckert, and B. Kainz, “Ultrasound Video Summarization Using Deep Reinforcement Learning,” in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, vol. 12263, A. L. Martel, P. Abolmaesumi, D. Stoyanov, D. Mateus, M. A. Zuluaga, S. K. Zhou, D. Racoceanu, and L. Joskowicz, Eds. Cham: Springer International Publishing, 2020, pp. 483–492. doi: 10.1007/978-3-030-59716-0_46.

[21] K. Muhammad, T. Hussain, M. Tanveer, G. Sannino, and V. H. C. de Albuquerque, “Cost-Effective Video Summarization Using Deep CNN With Hierarchical Weighted Fusion for IoT Surveillance Networks,” *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4455–4463, May 2020, doi: 10.1109/JIOT.2019.2950469.

[22] M. S. Nair and J. Mohan, “Static video summarization using multi-CNN with sparse autoencoder and random forest classifier,” *Signal Image Video Process.*, vol. 15, no. 4, pp. 735–742, Jun. 2021, doi: 10.1007/s11760-020-01791-4.

[23] J. Lin, S. Zhong, and A. Fares, “Deep hierarchical LSTM networks with attention for video summarization,” *Comput. Electr. Eng.*, vol. 97, p. 107618, Jan. 2022, doi: 10.1016/j.compeleceng.2021.107618.

[24] M. Rhevanth, R. Ahmed, V. Shah, and B. R. Mohan, “Deep Learning Framework Based on Audio–Visual Features for Video Summarization,” in *Advanced Machine Intelligence and Signal Processing*, vol. 858, D. Gupta, K. Sambyo, M. Prasad, and S. Agarwal, Eds. Singapore: Springer Nature Singapore, 2022, pp. 229–243. doi: 10.1007/978-981-19-0840-8_17.

[25] B. Mahasseni, M. Lam, and S. Todorovic, “Unsupervised Video Summarization with Adversarial LSTM Networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, Jul. 2017, pp. 2982–2991. doi: 10.1109/CVPR.2017.318.

[26] T.-J. Fu, S.-H. Tai, and H.-T. Chen, “Attentive and Adversarial Learning for Video Summarization,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa Village, HI, USA, Jan. 2019, pp. 1579–1587. doi: 10.1109/WACV.2019.00173.

[27] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris, and I. Patras, “Unsupervised Video Summarization via Attention-Driven Adversarial Learning,” in *MultiMedia Modeling*, vol. 11961, Y. M. Ro, W.-H. Cheng, J. Kim, W.-T. Chu, P. Cui, J.-W. Choi, M.-C. Hu, and W. De Neve, Eds. Cham: Springer International Publishing, 2020, pp. 492–504. doi: 10.1007/978-3-030-37731-1_40.

[28] M. U. Sreeja and B. C. Kovoor, “A multi-stage deep adversarial network for video summarization with knowledge distillation,” *J. Ambient Intell. Humaniz. Comput.*, Jan. 2022, doi: 10.1007/s12652-021-03641-8.

[29] A. Sharghi, B. Gong, and M. Shah, “Query-Focused Extractive Video Summarization,” in *Computer Vision – ECCV 2016*, vol. 9912, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 3–19. doi: 10.1007/978-3-319-46484-8_1.

[30] B. A. Plummer, M. Brown, and S. Lazebnik, “Enhancing Video Summarization via Vision-Language Embedding,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, Jul. 2017, pp. 1052–1060. doi: 10.1109/CVPR.2017.118.

[31] K. Zhang, K. Grauman, and F. Sha, “Retrospective Encoders for Video Summarization,” in *Computer Vision – ECCV 2018*, vol. 11212, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 391–408. doi: 10.1007/978-3-030-01237-3_24.

[32] F. Wang, F. Liu, S. Zhu, L. Fu, Z. Liu, and Q. Wang, “HEVC intra frame based compressed domain video summarization,” in *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing - AIIPCC ’19*, Sanya, China, 2019, pp. 1–7. doi: 10.1145/3371425.3371450.

[33] C. Huang and H. Wang, “A Novel Key-Frames Selection Framework for Comprehensive Video Summarization,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 2, pp. 577–589, Feb. 2020, doi: 10.1109/TCSVT.2019.2890899.

[34] J.-H. Huang and M. Worring, “Query-controllable Video Summarization,” in *Proceedings of the 2020 International Conference on Multimedia Retrieval*, Dublin Ireland, Jun. 2020, pp. 242–250. doi: 10.1145/3372278.3390695.

[35] Y. Zhao, Y. Guo, R. Sun, Z. Liu, and D. Guo, “Unsupervised video summarization via clustering validity index,” *Multimed. Tools Appl.*, vol. 79, no. 45–46, pp. 33417–33430, Dec. 2020, doi: 10.1007/s11042-019-7582-8.

[36] M. Narasimhan, A. Rohrbach, and T. Darrell, “CLIP-It! Language-Guided Video Summarization.” arXiv, Dec. 07, 2021. Accessed: Oct. 05, 2022. [Online]. Available: http://arxiv.org/abs/2107.00650

[37] W. Zhu, J. Lu, Y. Han, and J. Zhou, “Learning multiscale hierarchical attention for video summarization,” *Pattern Recognit.*, vol. 122, p. 108312, Feb. 2022, doi: 10.1016/j.patcog.2021.108312.

[38] G. Wu, J. Lin, and C. T. Silva, “IntentVizor: Towards Generic Query Guided Interactive Video Summarization,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, USA, Jun. 2022, pp. 10493–10502. doi: 10.1109/CVPR52688.2022.01025.

[39] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman, “SIFT Flow: Dense Correspondence across Different Scenes,” in *Computer Vision – ECCV 2008*, vol.

5304, D. Forsyth, P. Torr, and A. Zisserman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 28–42. doi: 10.1007/978-3-540-88690-7_3.

[40] M. Hassan and T. Shanableh, “Predicting split decisions of coding units in HEVC video compression using machine learning techniques,” *Multimed. Tools Appl.*, vol. 78, no. 23, pp. 32735–32754, Dec. 2019, doi: 10.1007/s11042-018-6882-8.

[41] T. Shanableh, “Altering split decisions of coding units for message embedding in HEVC,” *Multimed. Tools Appl.*, vol. 77, no. 7, pp. 8939–8953, Apr. 2018, doi: 10.1007/s11042-017-4787-6.

[42] S. Youssef and T. Shanableh, “Detecting Double and Triple Compression in HEVC Videos Using the Same Bit Rate,” *SN Comput. Sci.*, vol. 2, no. 5, p. 406, Sep. 2021, doi: 10.1007/s42979-021-00800-8.

[43] T. Shanableh, “Saliency detection in MPEG and HEVC video using intra-frame and inter-frame distances,” *Signal Image Video Process.*, vol. 10, no. 4, pp. 703–709, Apr. 2016, doi: 10.1007/s11760-015-0798-9.

[44] S. E. F. de Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo, “VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method,” *Pattern Recognit. Lett.*, vol. 32, no. 1, pp. 56–68, Jan. 2011, doi: 10.1016/j.patrec.2010.08.004.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[46] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” *ArXiv160207261 Cs*, Aug. 2016, Accessed: Nov. 28, 2021. [Online]. Available: http://arxiv.org/abs/1602.07261

[47] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *ArXiv14091556 Cs*, Apr. 2015, Accessed: Nov. 10, 2021. [Online]. Available: http://arxiv.org/abs/1409.1556

[48] M. V. Mussel Cirne and H. Pedrini, “VISCOM: A robust video summarization approach using color co-occurrence matrices,” *Multimed. Tools Appl.*, vol. 77, no. 1, pp. 857–875, Jan. 2018, doi: 10.1007/s11042-016-4300-7.

[49] J. Wu, S. Zhong, J. Jiang, and Y. Yang, “A novel clustering method for static video summarization,” *Multimed. Tools Appl.*, vol. 76, no. 7, pp. 9625–9641, Apr. 2017, doi: 10.1007/s11042-016-3569-x.

[50] S. E. F. de Avila, A. da_Luz Jr., A. de A. Araújo, and M. Cord, “VSUMM: An Approach for Automatic Video Summarization and Quantitative Evaluation,” in *2008 XXI Brazilian Symposium on Computer Graphics and Image Processing*, Campo Grande, Brazil, Oct. 2008, pp. 103–110. doi: 10.1109/SIBGRAPI.2008.31.

Appendix

This thesis work was published as a journal article in IEEE Access in July of 2022. The following is the citation to the published journal article:

O. Issa and T. Shanableh, "CNN and HEVC Video Coding Features for Static Video Summarization," in *IEEE Access*, vol. 10, pp. 72080-72091, 2022, doi: 10.1109/ACCESS.2022.3188638.

Vita

Obada Issa was born in 1998, Kuwait, where he received his primary and secondary education. He received his B.Sc. degree in Computer Science from the American University of Sharjah in 2020.

In January 2021, he joined the Computer Engineering M.Sc. program in the American University of Sharjah as a graduate research and teaching assistant. His research interests are in Data Science , Data Analytics, Machine Learning, Deep Learning, Image processing, and Video processing.