Deep Learning Methods for Codecs

Zahra Montajabi

A Thesis

in

The Department

of

Concordia Institute for Information Systems Engineering (CIISE)

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Quality System Engineering) at

Concordia University

Montréal, Québec, Canada

December 2022

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Zahra Montajabi**

Entitled: **Deep Learning Methods for Codecs**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Quality System Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair and Examiner
Dr. Abdessamad Ben Hamza

_____ Examiner
Dr. Arash Mohammadi

_____ Supervisor
Dr. Nizar Bouguila

Approved by _____
Dr. Abdessamad Ben Hamza, Chair
Department of Concordia Institute for Information Systems Engineering (CIISE)

_____ 2022

_____
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Deep Learning Methods for Codecs

Zahra Montajabi

Due to the recent advent of high-resolution mobile and camera devices, it is necessary to develop an efficient solution for saving the new video content instead of traditional compression methods. Recently, video compression received enormous attention among computer vision problems in media technologies. Using state-of-the-art video compression methods, videos can be transmitted in a better quality requiring less bandwidth and memory. The advent of neural network-based video compression methods remarkably promoted video coding performance.

In this thesis, two different video compression methods are proposed. The details of the models architectures and evaluation methods are elaborated, and the results are reported numerically and visually. In the first method, Recurrent Neural Network (RNN) and long short-term memory (LSTM) units are used to keep the valuable information and eliminate unnecessary ones to iteratively reduce the quality loss of reconstructed videos and therefore, encode the videos with less quality loss. In the second method, an Invertible Neural Network (INN) is utilized to reduce the information loss problem. Unlike the classic auto-encoders which lose some information during encoding, INN can preserve more information and therefore, reconstruct videos with more clear details. The proposed methods are evaluated using the peak signal-to-noise ratio (PSNR), video multimethod assessment fusion (VMAF), and structural similarity index measure (SSIM) quality metrics. The proposed methods are applied to two different public video compression datasets, the Ultra Video Group (UVG) dataset and the YouTube UGC video compression dataset, and the results show that our methods outperform existing standard video encoding schemes such as H.264 and H.265.

In the third part of this thesis, a deep learning method is used to find the semantic regions of interest (SRoI) which is one of the most challenging problems in computer vision and image

processing. Finding the semantic regions of interest can be used in different image processing tasks such as image and video compression, enhancement, and reformatting. By knowing the semantic region of interest within images, we can improve the visual quality of images by compressing the more important parts with higher quality and the less important parts, such as the background, with a lower quality. This operation can be achieved without changing the overall compression ratio and the Peak Signal-to-noise Ratio (PSNR) quality metric. Finding the SRoI can make the processes of image enhancement and color correction more accurate by focusing only on the important parts. Moreover, for the image reformatting process, the important parts of the image may be lost. But by using the SRoI, we can reformat the image in a better way by keeping the most important regions in the frame. For these purposes, a method is proposed using OpenAI's CLIP model to find SRoI by performing a semantic search for objects in the image which are detected by an object detection model called Generic RoI Extractor (GRoIE). The results of the proposed method are reported.

Acknowledgments

First and foremost, I am extremely grateful to my supervisor, Prof. Nizar Bouguila for his invaluable advice, continuous support, and patience in this Master's program. It was my honor to be her student and I will be forever thankful to him for providing me with this great opportunity. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life.

I would like to acknowledge the thesis committee members, Dr. Abdessamad Ben Hamza and Dr. Arash Mohammadi for their encouragement and wonderful and insightful comments that helped me considerably improve my research work.

I was able to complete this thesis thanks to Mitacs and Avid company for their financial support. In addition, I would like to thank my internship supervisor, Rob Gonsalves, and the rest of the Avid team who gave me excellent advice and assistance. I had a terrific time working with them and look forward to continuing professional relationships.

I had the pleasure of having Vahid by my side while working on my thesis, and I would like to express my gratitude to him for his advice and support. I would not be able to complete this research without his help.

Also, I want to thank all my friends; especially, Behzad, Leila, Farhad, Masi, and Sina for helping me in the process of arrival and settle-down in Montreal and always supporting me like a family.

A special thanks to my family and my aunt, I would not have been able to finish my education without their unconditional support and encouragement throughout my studies.

Last but not least, I must express my special gratitude to my husband, Saman, for his continued

support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Backgound

Nowadays, many web activities and web-based applications such as real-time communications and live streaming include a large amount of video content. Video contents contribute to about 80% of the Internet traffic Networking (2016). The use of camera has increased in the recent decade and there are more images and videos with a higher resolution and new formats than in previous decades. So it is necessary to store them efficiently. Image/video compression is a computing technology to convert image/video into smaller size binary code to make the process of storage and transmission easier. A good compression method has two features; first, the number of bits required to store the data should be lower, and second, the loss between original images and reconstructed one should be lower (higher quality) D. Liu, Li, Lin, Li, and Wu (2021). Many lossy compression methods are used as a solution, in which instead of storing the raw RGB data, a lossy image that has few visual changes is stored. Fig. 1.1 shows the basic compression scheme.

There are different types of redundancies within images/videos such as spatial redundancy, visual redundancy and statistical redundancy which are very important for coding. Traditional coding methods used entropy coding to decrease statistical redundancies Ma et al. (2019). Huffman coding Huffman (1952), Golomb coding Golomb (1966), and arithmetic coding Witten, Neal, and Cleary (1987) are some examples. Also, to decrease spatial frequencies, transform coding was proposed in 1960 such as Fourier transform Andrews and Pratt (1968) and Hadamard transform Pratt, Kane, and

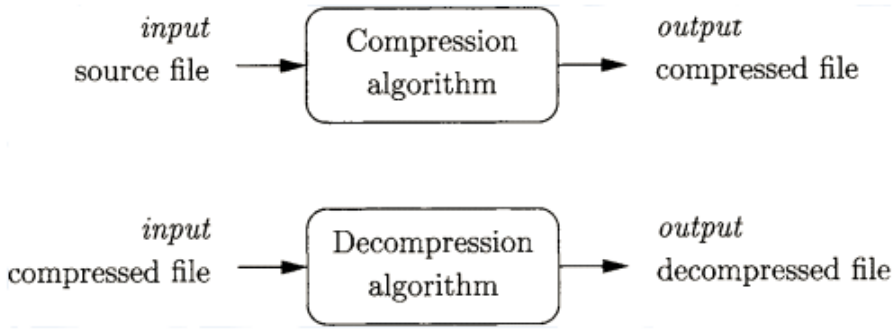


Figure 1.1: Block diagram of the basic compression scheme. Pu (2006)

Andrews (1969). To reduce spatial redundancy and visual redundancy, the quantization techniques are proposed Ma et al. (2019).

As high-quality video content such as 4k videos become more prevalent, more sophisticated video compression methods are required to save the communication bandwidth/storage space while offering high-quality video encoding with less loss. Moreover, the higher quality of encoded videos enhances computer vision schemes such as object tracking and action recognition.

Among different recent methods developed for compression, deep learning-based compression methods gained more interest and attention. As a result, learned autoencoder-based video compression is developed as a powerful competitor to traditional methods. An autoencoder is an artificial neural network used for unsupervised learning and its goal is learning a representation (encoding) of a set of data, and reducing the size of that set Mentzer, Toderici, Tschannen, and Agustsson (2020). Neural networks, specifically convolution neural networks (CNN) were very successful in the recent image/video processing tasks. CNN consists of convolutional layers and sometimes may contain fully connected layers. The parameters of the layers are trained for different tasks like classification and prediction or feature extraction to transform the image/video into feature space with compressed representation which is useful for image/video coding. The architecture of a simple neural network is shown in Fig. 1.2.

Some of the earliest use of neural networks for image/video coding goes back to the 1990s; in those researches, the networks were very shallow and the compression was not efficient Dony and Haykin (1995), J. Jiang (1999). Later, the computing platforms were improved and more data was

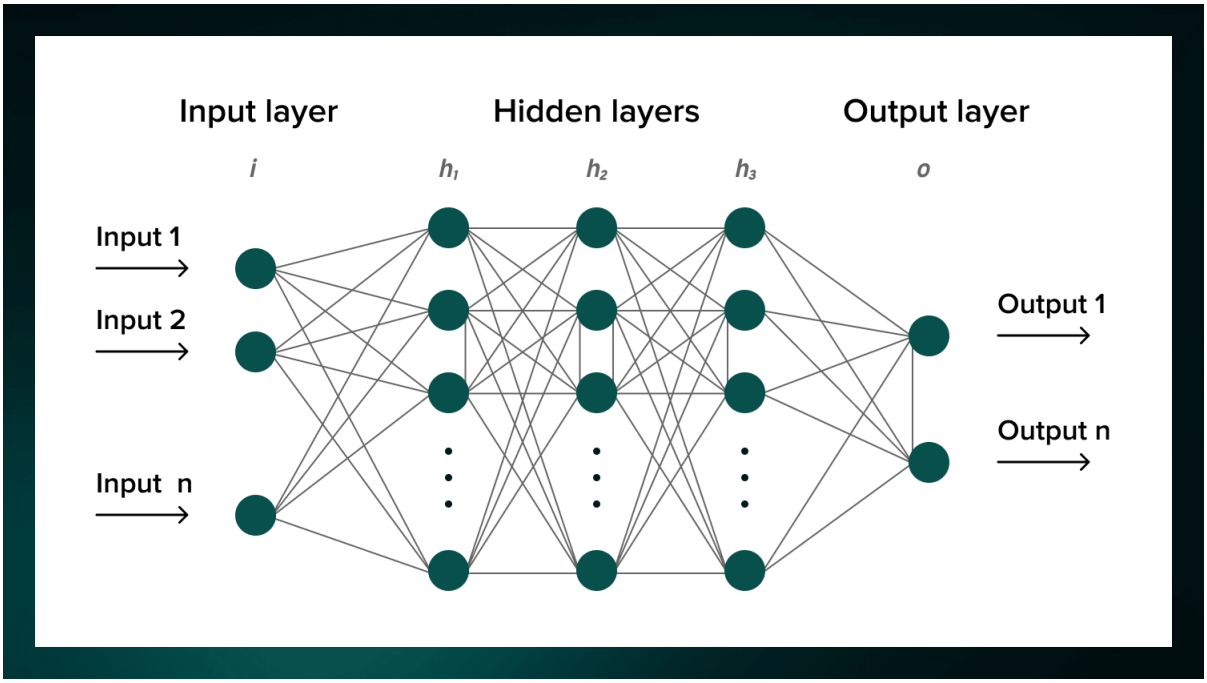


Figure 1.2: Architecture of a simple neural network. Gavrilova (2021)

available, so it was possible to design and train deeper models with even more than 1000 layers He, Zhang, Ren, and Sun (2016).

Deep learning reduces the need of handcrafted representations and can be used especially for processing natively unstructured data like visual signal D. Liu et al. (2021). In traditional compression methods, they map the input to latent feature representation linearly. In contrast, deep neural network-based methods are capable of using highly non-linear transformations and end-to-end training on a large scale to optimize compression. Some of the recent deep learning-based models are reviewed comprehensively in D. Liu et al. (2021).

1.1.1 Types of Compression Techniques

There are two types of coding, lossy and lossless. The compression ratios are higher in lossy methods than the lossless compression techniques Hussain, Al-Fayadh, and Radi (2018). In lossless compression, the image/video is reconstructed from the bits perfectly without losing data while in lossy coding some parts of the data cannot be reconstructed and lead to some negligible artifacts in the reconstructed image/video. JPEG is one example of lossy compression and PNG is an example

of a lossless compression format. Some of the common methods are elaborated in the following.

Lossy Compression

- Transform Coding:

  Transformation of the image from one domain to another, and then encoding the transformed data based on interpixel correlation. Some of the transform coding approaches are Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) Edwards (1991), Karhunen–Loeve Transform (KLT) Dony et al. (2001), and K-Means algorithm Krishna, Ramakrishnan, and Thathachar (1997).

- Chroma Subsampling:

  It uses the fact that human eyes are more sensitive to brightness than colors and can recognize changes in brightness more than colors. So it is possible to reduce some color information without significant artifacts. The color information (chroma) can be sampled at a lower rate than brightness information (luma) Chung, Hsu, and Huang (2017). Some of the chroma subsampling approaches which are based on a three-component ratio are $4:2:2$ Lin, Chung, and Fang (2014), $4:1:1$, and $4:2:0$.

- Fractal Compression:

  This approach is based on the fact that each image (usually natural images) has identical and repetitive parts and patterns Khaitan and Agarwal (2019). Some of the fractal coding approaches are Fast Sparse Fractal Image Compression (FSFIC), and Optimization-Based Fractal Compression Techniques Menassel, Nini, and Mekhaznia (2018).

Lossless Compression

The goal of lossless compression is to encode all the data and reconstruct the output the same as the original. It is usually useful for medical imaging, technical drawing, and satellite images Elakkiya and Thivya (2021).

- Run-Length Coding:

This approach is appropriate for monochrome images that have large areas of the same color. It works based on searching the image for runs of pixels with the same color value and encoding the length. Some of the run-length coding approaches are Run-length coding (RLC) Apostolico, Landau, and Skiena (1997), Improved Versions of RLC Babu, Eswaran, and Kumar (2016), and RLC for Data and Speech Compression Arif and Anand (2012), Amin, Qureshi, Junaid, Habib, and Anjum (2011).

- Entropy Coding:

  In this approach, the minimum number of bits is dedicated to representing the quantized data. Some of the entropy coding methods are Huffman coding and Arithmetic coding. There are also some other variants of entropy coding proposed which are proposed later such as Kabir and Mondal (2017), Mentzer, Agustsson, Tschannen, Timofte, and Van Gool (2019).

- Dictionary-Based Compression:

  In this type of coding, short codewords and preferably fixed-length replaces the variable-length codes Sharma and Gupta (2017) and they are kept in a dictionary. They are also useful for text compression. One of the dictionary-based compression methods is Lempel–Ziv algorithm which includes LZ77, and LZ78 Jain and Lakhtaria (2016), Guo, Lu, Allebach, and Bouman (2017).

- Predictive Coding:

  This approach is based on the fact that in each image, pixels can be predicted from their neighbors with good accuracy since they are correlated. Their redundancies can be reduced and therefore, it is possible to create one dimensional data instead Shukla, Alwani, and Tiwari (2010). Some of the predictive coding methods are Median Edge Detector (MED) Martucci (1990), Gradient Adjusted Predictor (GAP) Rahman and Mohamed (2022), Gradient Edge Detector (GED) Avramović and Reljin (2010), Adaptive Linear Prediction Coding (ALPC) Motta, Storer, and Carpentieri (2000).

- Chain Codes:

This approach is useful in monochrome or binary images and for segmentation. Each connected component, (or blob) in the image is separately encoded. Some of the chain codes methods are Freeman Chain Code of Eight Directions (FCCE) Kothuri, Annapurna, and Lukka (2013), Vertex Chain Code (VCC) Y. K. Liu, Wei, Jie Wang, and Žalik (2007), Three Orthogonal Symbol Chain Code (3OT) Sanchez-Cruz and Rodriguez-Dagnino (2005), and Unsigned Manhattan Chain Code (UMCC) Žalik, Mongus, Liu, and Lukač (2016).

1.2 Related Works

In this section, related works for each of the traditional methods and deep learning-based methods are elaborated.

1.2.1 Hand-crafted Compression

Existing traditional compression methods like JPEG Wallace (1992), JPEG 2000 Skodras, Christopoulos, and Ebrahimi (2001), BPG Bellard (2014), WebP *An image format for the Web* (n.d.), HEVC Sullivan, Ohm, Han, and Wiegand (2012), and VVC Ohm and Sullivan (2018) are not optimized; because each module is optimized independently. In these methods, they map the input to latent feature representation linearly. These modules include intra-prediction, wavelet transform or discrete cosine transform Ahmed, Natarajan, and Rao (1974), quantization, and entropy coders such as content adaptive binary arithmetic coder (CABAC) or Huffman coder. In these methods, each module is designed in a way to have multiple modes and the best mode is determined by rate-distortion optimization. In VVC, even more modes, transform types, and larger coding units are designed. Also, some hybrid methods are designed, such as Z. Cheng, Sun, Takeuchi, and Katto (2018b), that have the advantage of both conventional compression algorithms and the latest learned super-resolution approaches.

1.2.2 Learned Compression

Using autoencoder architecture is very widespread in the recent deep learning-based compression methods Vincent, Larochelle, Bengio, and Manzagol (2008). To make the end-to-end training

possible, some studies Theis, Shi, Cunningham, and Huszár (2017), Ballé, Laparra, and Simoncelli (2017), Agustsson et al. (2017) worked on using non-differential quantization and rate estimation. Later works improved their methods by designing the network layers to obtain more efficient latent representation. For example in some papers Rippel and Bourdev (2017), Santurkar, Budden, and Shavit (2018), Agustsson, Tschannen, Mentzer, Timofte, and Gool (2019) to obtain better quality at a low bit rate, generative models which learn the distribution by adversarial training are used. Some approaches Toderici et al. (2016), Toderici et al. (2017), Johnston et al. (2018) used recurrent neural networks to compress the residual information recursively. To improve models, one paper used deep residual units Z. Cheng, Sun, Takeuchi, and Katto (2019a) and another paper used energy compaction Z. Cheng, Sun, Takeuchi, and Katto (2019b). Also, some methods used principle component analysis Abdi and Williams (2010) for de-correlating different channels Z. Cheng, Sun, Takeuchi, and Katto (2018a). Content weighted strategy is used in another approach M. Li, Zuo, Gu, Zhao, and Zhang (2018). The recent methods use adaptive context model which can obtain a tradeoff between reconstruction errors and entropy (required bits) and make the process more optimized such as Lee, Cho, and Beack (2019), Minnen, Ballé, and Toderici (2018), Ballé, Minnen, Singh, Hwang, and Johnston (2018), Mentzer, Agustsson, Tschannen, Timofte, and Van Gool (2018). These methods generate promising results compared with previous learned compression methods.

1.3 Thesis Overview

The organization of this thesis is as follows:

- In Chapter 1, the background knowledge regarding compression and deep learning methods and the related works are introduced.
- In Chapter 2, a video compression method using a Recurrent Neural Network is proposed and the experimental results are presented.
- In Chapter 3, a video compression method using an Invertible Neural Network is proposed and the experimental results are presented.

- In Chapter 4, a method to find the semantic regions of interest in images using deep learning methods is proposed and tested for three different image processing applications.
- In Chapter 5, our research works and contributions are provided that generalize the significance of the proposed methods in this thesis.

Chapter 2

Recurrent Neural Network-Based Video Compression

In this chapter, a video compression method is presented based on Recurrent Neural Network (RNN). The method includes an encoder, a middle module, and a decoder. Binarizer is utilized in the middle module to achieve better quantization performance. In encoder and decoder modules, long short-term memory (LSTM) units are used to keep the valuable information and eliminate unnecessary ones to iteratively reduce the quality loss of reconstructed video. This method reduces the complexity of neural network-based compression schemes and encodes the videos with less quality loss. The proposed method is evaluated using peak signal-to-noise ratio (PSNR), video multimethod assessment fusion (VMAF), and structural similarity index measure (SSIM) quality metrics. The proposed method is applied to two different public video compression datasets and the results show that the method outperforms existing standard video encoding schemes such as H.264 and H.265.

2.1 Introduction

Many of deep learning-based compression methods used convolutional neural networks (CNN) and auto-encoders such as Ballé et al. (2018), Ballé et al. (2017), Theis et al. (2017). Some of these deep learning based models are reviewed comprehensively in D. Liu et al. (2021). There are several

CNN-based compression methods Murn, Blasi, Smeaton, and Mrak (2021), Murn, Blasi, Smeaton, O'Connor, and Mrak (2020) which achieved a higher compression ratio with a less quality loss, however, these methods developed very complex networks with deeper layers which increase the computational cost. For example, in Chen et al. (2017), a convolutional auto-encoder is used; which also used Huffman coding to encode the quantized feature maps. Another method Y. Li et al. (2018) used CNN-based block up-sampling scheme for intra frame coding. Also, in F. Jiang et al. (2018), CNN is used to learn and preserve the structural information (downsampling) prior to encoding and then reconstruct the decoded image with high quality (upsampling). Compared to recent CNN methods which have lots of layers for improving the performance, RNN-based methods are less complex and they are less common among recent deep learning-based compression approaches. RNNs can improve the future predictions iteratively. An example of RNN-based compression methods is the method proposed in Toderici et al. (2017) .

Previous works have shown that using generalized divisive normalization (GDN) is more helpful to use in compression tasks, instead of other nonlinear activation functions such as ReLU, due to its higher consistency with human visual perception Ballé et al. (2017). For example, Ballé (2018) proved that GDN can be applied instead of other nonlinearity transforms to increase the approximation capacity of the transforms. Also, GDN is used in the autoencoder structures of image compression Lee et al. (2019) and Klopp, Wang, Chien, and Chen (2018); it helped to decrease redundancies between channels and apply a different normalization to each channel at the same spatial location.

Some parts of images contain more important information. Image/video scenes can be grouped into perceptually significant and perceptually insignificant areas. These visually insignificant pixels are compressed more and users cannot notice compression impairments if they don't compare details of the picture with the original image pixel by pixel. So, it is possible to use fewer bits. Therefore, many compression methods have texture analyzer which sort the areas based on significance; and encode the perceptually significant regions as the main bit stream payload and perceptually insignificant areas as the side information. Also, in the inverse process, they have a texture synthesizer which synthesize texture to restore the pixels Ding et al. (2021). We used this type of analysis/synthesis method in our video compression model.

Our method, inspired by Ballé et al. (2018), Toderici et al. (2017), Islam, Dang, Lee, and Moon (2021), Toderici et al. (2016), incorporates an encoder, a middle module, and a decoder. These modules benefit from convolution layers, GDN, long short-term memory cells, and recurrent neural networks. In the method, first, feature extraction and analysis, down sampling, and quantization are done and then the decoder performs synthesis and up-sampling and the video is iteratively reconstructed through LSTM units. To the best of our knowledge, this is the first video coding/decoding method which benefits from a GDN layer to manage different bit rates, offers an enhanced performance by using binarizer Toderici et al. (2017) for quantization in the middle module, and also using LSTM cells to keep the necessary information and forget useless ones to improve the output iteratively.

To validate the performance of the proposed method, we tested our method on Ultra Video Group (UVG) dataset Mercat, Viitanen, and Vanne (2020) and YouTube UGC video compression dataset Y. Wang, Inguva, and Adsumilli (2019) and reported the results by using PSNR, VMAF and SSIM quality metrics under the similar setting.

The chapter is organized as follows. The video coding method is described in detail in Section 2.2. Experimental results are presented in Section 2.3, followed by concluding remarks in Section 2.4.

2.2 Approach

The method architecture consists of three parts as illustrated in Fig. 2.1: an encoder block, a middle layer, and a decoder block. Encoder block performs texture analysis in the beginning, using a convolution and a GDN layer and then encode the frames. Decoder performs synthesis inversely using convolution and iGDN layer and then decoding the frames. In the first step, video frames are sent into the encoder for feature extraction and generating sequences. In the middle module, the input frames are converted into binary codes that can be stored or transmitted to the decoder. Middle module uses the binary code generator method Toderici et al. (2017) for quantization and finally, the video will be reconstructed through decoder.

Let $V = \{f_1, f_2, ..., f_T\}$ be the video sequences and f_t be the video frame at time t and $\hat{f}_t$ be

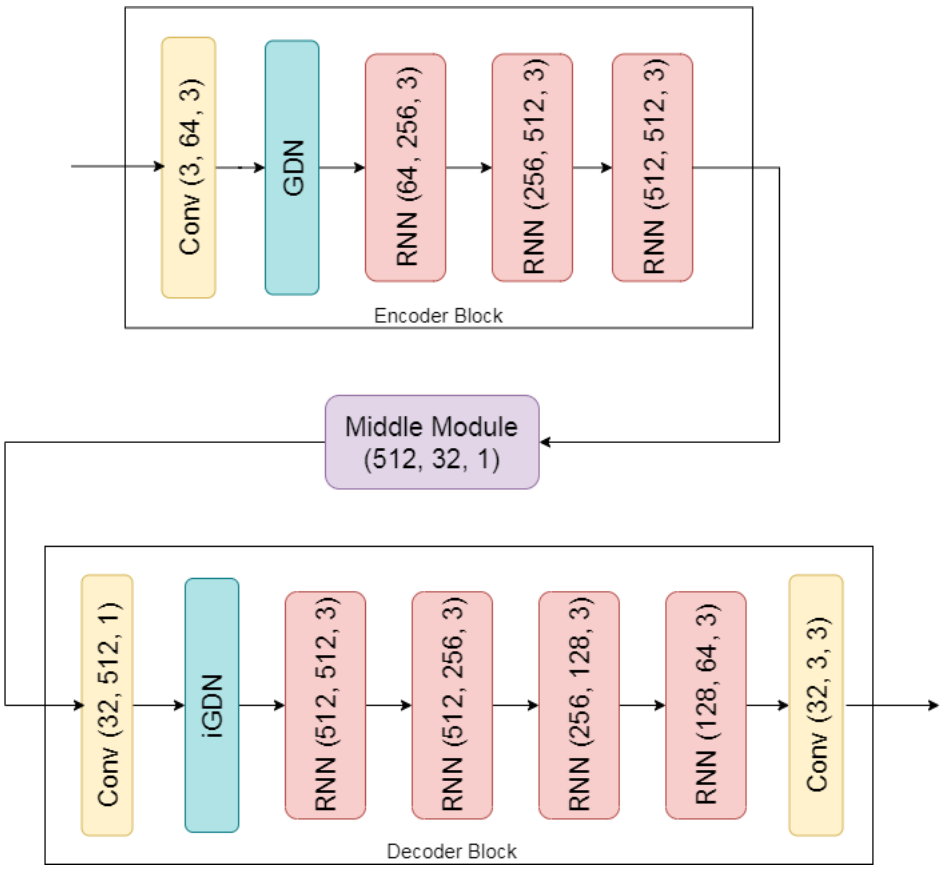


Figure 2.1: Structure of our video compressor

the reconstructed frame. The residual between the original frame and the decoded one is obtained by $r_t = f_t - \hat{f}_t$.

If we have n iterations, the loss at each iteration of the method can be calculated as:

$$L_1 = \beta \sum_n ||r_{t,n}|| \tag{1}$$

In the following, the details of the three blocks are introduced.

2.2.1 Encoder Block

The encoder block consists of a convolution layer with the input channel of size 3, output channel of size 64, kernel size of 3, and stride of 2. The encoder block has a generalized divisive normalization layer Ballé, Laparra, and Simoncelli (2016) which acts as a nonlinear transformation

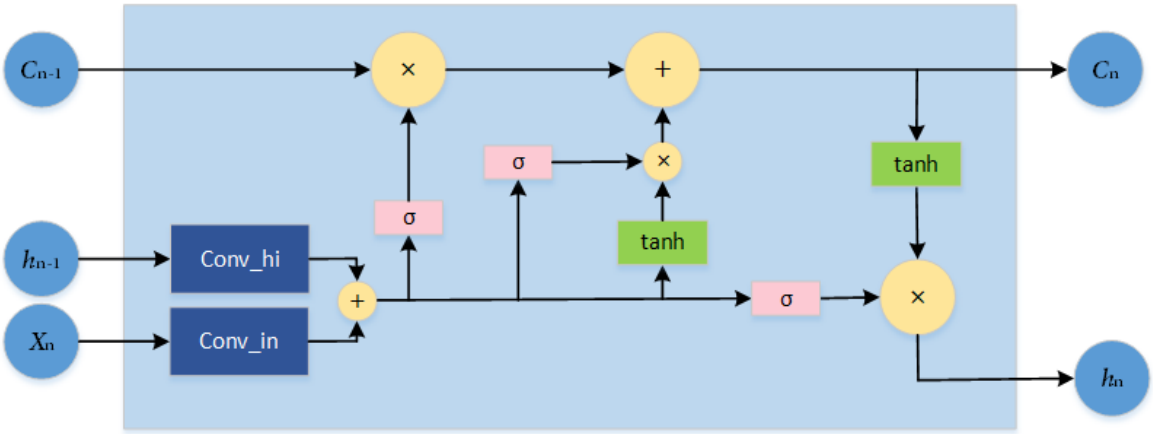


Figure 2.2: Structure of LSTM unit

between convolution layers. After 2D convolution, spatial location (i, j) of k^{th} input channel in the n^{th} step (iteration) is calculated by:

$$w_{t,k}^{n}(i,j) = \sum_{m} (h_{t,k,m}^{n} * f_{t,k,m}^{n})(i,j) \tag{2}$$

After convolution, GDN layer works based on 3 in which bias and scale parameters of normalization are $\beta_{t,k}^{n}$ and $\gamma_{t,k}^{n}$.

$$s_{t,k}^{(n)}(i) = \frac{w_{t,k}^{n}(i)}{(\beta_{t,k}^{n} + \sum_{j} \gamma_{t,k,j}^{n}(w_{t,k}^{n}(j))^{2})^{0.5}} \tag{3}$$

Next part of the encoder block consists of three long short-term memory (LSTM) Hochreiter and Schmidhuber (1997) units which are a type of RNN. Feature extraction occurs iteratively in LSTM units, which remember the states and reconstruct the image in the decoder block. In each iteration, the estimated output of each LSTM unit is passed to the next unit's hidden layer. Fig. 2.2 shows the structure of the LSTM unit where x_n is the input feature vector of n^{th} iteration. c_{n-1} is the memory (cell) state, and h_{n-1} is the hidden state. Moreover, there are two convolution layers for the hidden vector and the input vector which are called Conv_hi and Conv_in respectively. The kernel size of Conv_in is 3 and the stride is 2. The kernel size of Conv_hi is 1 and the stride is 1. The outputs of LSTM cells are calculated according to the structure as follows:

$$c_n = f_n \odot c_{n-1} + i_n \odot \tilde{c_n} \tag{4}$$

$$h_n = o_n \odot tanh(c_n) \tag{5}$$

where, the output of forgete gate (f_n), input gate (i_n) and output gate (o_n) are:

$$f_n = \sigma(W_f \cdot conv\_in(x_n) + U_f \cdot conv\_hi(h_{n-1}) + b_f) \tag{6}$$

$$i_n = \sigma(W_i \cdot conv\_in(x_n) + U_i \cdot conv\_hi(h_{n-1}) + b_i) \tag{7}$$

$$\tilde{c}_n = tanh(W_c \cdot conv\_in(x_n) + U_c \cdot conv\_hi(h_{n-1})] + b_c) \tag{8}$$

$$o_n = \sigma(W_o \cdot conv\_in(x_n) + U_o \cdot conv\_hi(h_{n-1})] + b_o) \tag{9}$$

2.2.2 Middle Module

According to quantization method which is elaborated in Toderici et al. (2017) and Toderici et al. (2016), the quantization noise is applied for training at the middle module. The module includes a linear convolution layer with an activation function. This module produces binary codes between -1 and 1 and can compress each input frame with dimensions of height $\times$ width $\times$ 3 to the size of (height/16) $\times$ (width/16) $\times$ 32 which means 1/8 reduction for bit per pixel (bpp). In the middle module, the input channel size is 512, the output channel size is 32 and the kernel size is 1. For activation function, tanh is used. Using this approach, the compression ratio of the n_{th} iteration will be $n/192$ which enhances the performance of method.

2.2.3 Decoder Block

In order to reconstruct the video, the inverse of the functions incorporated in the encoder, must be implemented in the decoder. Therefore, the first step in the decoding process is using a CNN layer with input channel size of 32, output channel size of 512, and kernel size of 1. The next step is applying the inverse of GDN layer using the following equation:

$$\hat{w}^n_{t,k}(i) = \hat{s}^n_{t,k}(i).(\hat{\beta}^n_{t,k} + \sum_j \hat{\gamma}^n_{t,k,j}(\hat{s}^n_{t,k}(j))^2)^{0.5} \tag{10}$$

Similar to encoder layers, the kernel size and stride of the other layers is 3 and 1, respectively. Finally, the video is reconstructed through four LSTM units and the final output is generated after a deconvolution.

2.3 Experiments

Experiments are performed on a Tesla V100-SXM2-16GB GPU. The network is trained for 12 epochs using a batch size of 16, learning rate of 0.0005, and with Adam optimizer Kingma and Ba (2015) on Pytorch framework.

2.3.1 Dataset

To train our video compression model, vimeo-90k dataset is used Xue, Chen, Wu, Wei, and Freeman (2019) which is a large dataset developed for different kinds of video processing tasks. The dataset includes 89800 clips with different contents in different categories. The resolution of the videos is 256 x 256 pixels. 5000 video clips from different categories are used to train our model.

To test the method and report the results, Ultra Video Group (UVG) dataset Mercat et al. (2020) is used which includes 16 versatile 4K (3840 × 2160) video sequences. Each video is 50 or 120 frames per second available in 4:2:0 YUV format. To test the videos, 1920 x 1080 resolution of the dataset is used, and they are resized by padding followed by cropping to be dividable by 32.

Another dataset which is used to test and evaluate the method is YouTube UGC Dataset Y. Wang

et al. (2019). There are 1500 video clips with a length of 20 seconds in different categories such as gaming, sports, animation, and lecture. Each clip is available in 4:2:0 YUV format and resolutions of 360P, 480P, 720P, and 1080P. To test the method, we selected 40 videos in different categories with 720P resolution.

2.3.2 Evaluation Method

To measure the performance of the proposed framework, peak signal-to-noise ratio (PSNR), video multimethod assessment fusion (VMAF), and structural similarity index measure (SSIM) quality metrics are used. A higher PSNR value shows a higher image quality Horé and Ziou (2010); and SSIM Dosselmann and Yang (2005) measures the similarity between two images and is better correlated with the human perception of distortion. Compared to PSNR and SSIM, VMAF is a subjective measure of the human eye perception and so it is a pivotel measure in real-world applications Rassool (2017).

2.3.3 Results

Average PSNR, VMAF, and SSIM of the proposed model on UVG dataset are 42.8, 80.15, and 0.98, respectively, which outperforms H.264 Wiegand, Sullivan, Bjontegaard, and Luthra (2003) and H.265 Sullivan et al. (2012) as can be seen in Table 2.1.

Table 2.1: Performance comparison of the methods applied to UVG dataset. The proposed method outperforms the others.

Quality Metric	PSNR	VMAF	SSIM
Proposed	**42.8**	**80.15**	**0.98**
H.264	38.8	68.3	0.925
H.265	39.2	67.9	0.94

Fig. 2.3 shows samples of the outputs of our framework and H.264 and H.265. In the first example of the dog 2.3a, the eye drops is more blur in H.264 and H.265. In the second example 2.3b, the details of the trees (like the green tree in the up-right of the zoomed image) and the sea are sharper and less quantized using our method than other methods. In the third example 2.3c which is

woman's face, the details of the face such as eyebrow and eyelash are more clear and sharper using our method than H.264 or H.265. The average bit per pixel (BPP) for all test sets is 0.32.

It is shown in Table 2.2 that our method outperforms H.264 and H.265 in terms of PSNR and SSIM on YouTube UGC video compression dataset. It has a better VMAF than H.265 and approximately similar VMAF with H.264.

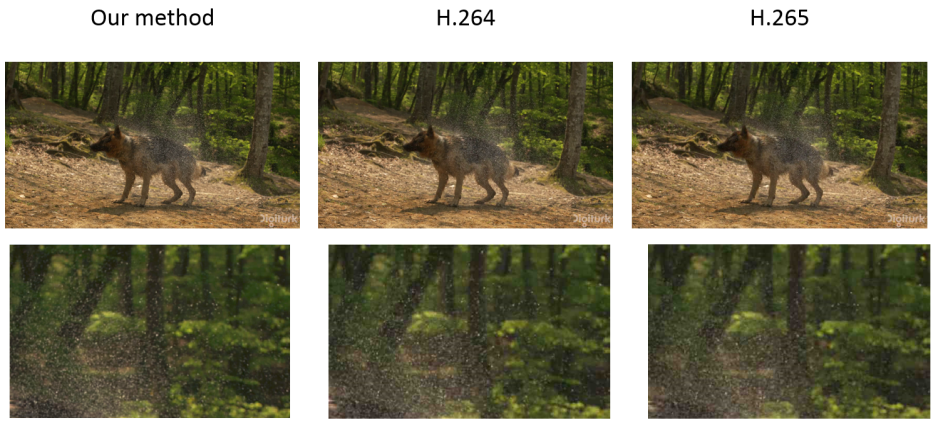
Table 2.2: Performance comparison of the methods applied to YouTube UGC dataset. The proposed method outperforms others in terms of PSNR and SSIM and has a competitive result with H.264 in terms of VMAF.

Quality Metric	PSNR	VMAF	SSIM
Proposed	**29.7**	**79.13**	**0.9**
H.264	27.8	79.26	0.85
H.265	27.9	78.5	0.86

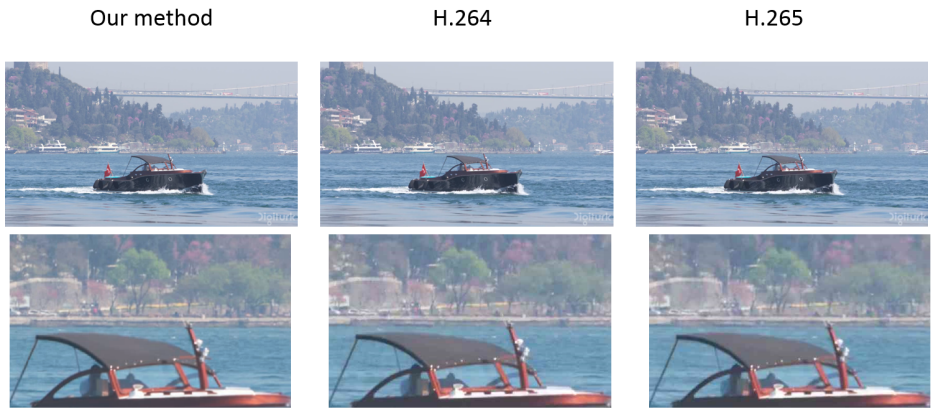
Some sample outputs of our framework and H.264 and H.265 are demonstrated in Fig. 2.4; in the first example 2.4a, the lips of woman's face are more blurry in H.264 and H.265, while using our mehtod they are less quantized and more clear. Also in the second example 2.4b, the letter (NBR) on the microphone of reporter is more quantized in H.264 and more blurry in H.265 while it is more clear and sharper using our method. Therefore, our method outperforms H.264 and H.265 based on both numerical and visual comparisons.

2.4 Conclusion

A video compression method based on RNN has been introduced. The method takes advantage of CNN, generalized divisive normalization method, and RNN layers and LSTM cells in encoder and decoder parts. Unlike the recent CNN-based methods that is very complex since they are deeper, the proposed RNN-based method has less complexity. The performance of the proposed architecture has been further improved by using binarizer for quantization and using LSTM cells in encoder/decoder parts for reducing useless information. The results of the method applied to standard datasets show a better quality in terms of PSNR, VMAF and SSIM as compared to the recognized methods such as H.264 and H.265.



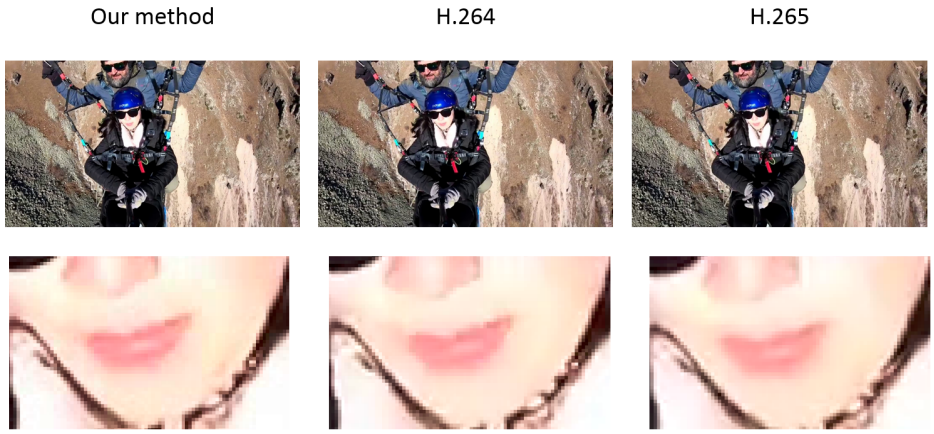
(a)



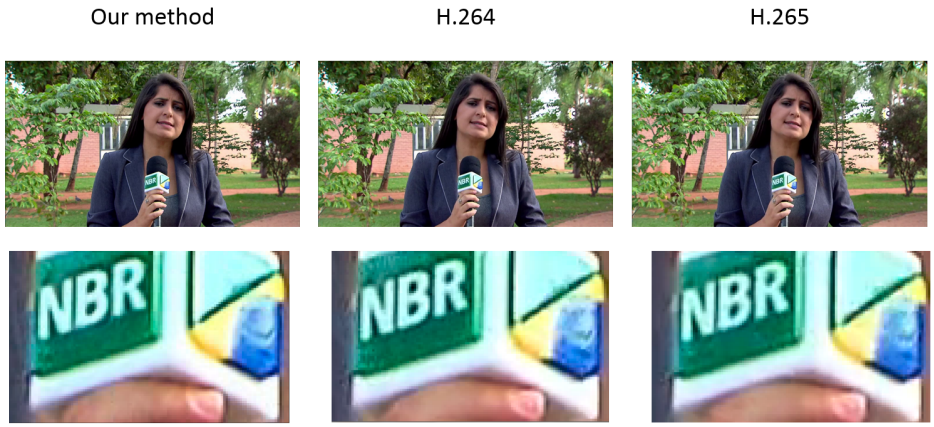
(b)

(c)

Figure 2.3: Samples of the proposed method, H.264, and H.265 outputs. The data used here are from the UVG dataset. (a) The eye drops is more blur and more quantized in H.264 and H.265. (b) The details of the trees (like the green tree in the up-right of the zoomed image) and the sea are sharper and less quantized using our method. (c) The details of the face such as eyebrow and eyelash are more clear and sharper using our method.



(a)



(b)

Figure 2.4: Samples of the proposed method, H.264, and H.265 outputs. The data used here are from the YouTube UGC video compression dataset. (a) The lips of woman's face are more blurry in H.264 and H.265, while using our mehtod they are less quantized and more clear. (b) The letter (NBR) on the microphone of reporter is more quantized in H.264 and more blurry in H.265 while it is more clear and sharper using our method.

Chapter 3

Invertible Neural Network-Based Video Compression

In this chapter, a video compression method is proposed using Invertible Neural Network (INN) to reduce the information loss problem. Unlike the classic auto-encoders which lose some information during encoding, INN can preserve more information and therefore, reconstruct videos with more clear details. Moreover, they don't increase the complexity of the network compared to traditional auto-encoders. The proposed method is evaluated on a public dataset and the experimental results show that the proposed method outperforms existing standard video encoding schemes such as H.264 and H.265 in terms of peak signal-to-noise ratio (PSNR), video multimethod assessment fusion (VMAF), and structural similarity index measure (SSIM).

3.1 Introduction

Among different recent methods developed for compression, deep learning-based compression methods gained more interest and attention. There are several traditional compression methods like JPEG Wallace (1992) and JPEG 2000 Skodras et al. (2001), which are not optimized; because each module is optimized independently. In these methods, they map the input to latent feature representation linearly. In contrast, deep neural network-based methods are capable of using highly non-linear transformations and end-to-end training on a large scale to optimize compression. Some

of the recent deep learning-based models are reviewed comprehensively in D. Liu et al. (2021).

Recently, INNs got more popular than previous auto-encoder frameworks and they concentrate on learning the forward process, using additional latent output variables to capture the information that would otherwise be lost, in contrast to classical neural networks that attempt to solve the ambiguous inverse problem directly Kingma and Dhariwal (2018), Dinh, Sohl-Dickstein, and Bengio (2016), Ardizzone et al. (2018). Although autoencoders are very capable of choosing the significant information for reconstruction, some amount of information is completely lost; while using INN for encoding and decoding helps to preserve the information. Specific features of INN are: it has a bijective mapping between input and output and its inverse exists; it is possible to compute both forward and inverse mapping efficiently; and there is a tractable Jacobian of both mappings that makes it possible to calculate posterior probabilities explicitly. The work in Ardizzone et al. (2018) demonstrated theoretically and practically that INNs are an effective analysis tool by using both synthetic data and real-world issues from astronomy and medicine categories. Another work on image generation Ardizzone, Lüth, Kruse, Rother, and Köthe (2019) used conditional INN architecture which performs better than variational autoencoders (VAEs) and generative adversarial networks (GANs). In Lugmayr, Danelljan, Gool, and Timofte (2020), INN is employed to more effectively address the ill-posed issue of super-resolution compared to GAN-based frameworks. Similarly, for image rescaling in Xiao et al. (2020), an invertible bijective transformation is used to reduce the ill-posed nature of image upscaling.

In each image or frame of a video, there are some parts that are less important, and by compressing them more than other parts and using fewer bits for them, it is not easy to notice the difference with the original one unless by pixel by pixel comparison. Therefore, each image/video frame can be grouped into perceptually significant and perceptually insignificant areas. Sorting the areas based on significance is called texture analyzer which is used in many compression methods, and the inverse process is called texture synthesizer, which restores the pixels Ding et al. (2021). This type of analysis/synthesis method is used in our video compression model. The proposed method, inspired by Minnen et al. (2018), Kingma and Dhariwal (2018), Xie, Cheng, and Chen (2021), Shi et al. (2016), Z. Cheng, Sun, Takeuchi, and Katto (2020) incorporates four different modules: feature enhancement which is used to improve the nonlinear representation, INN which helps to reduce

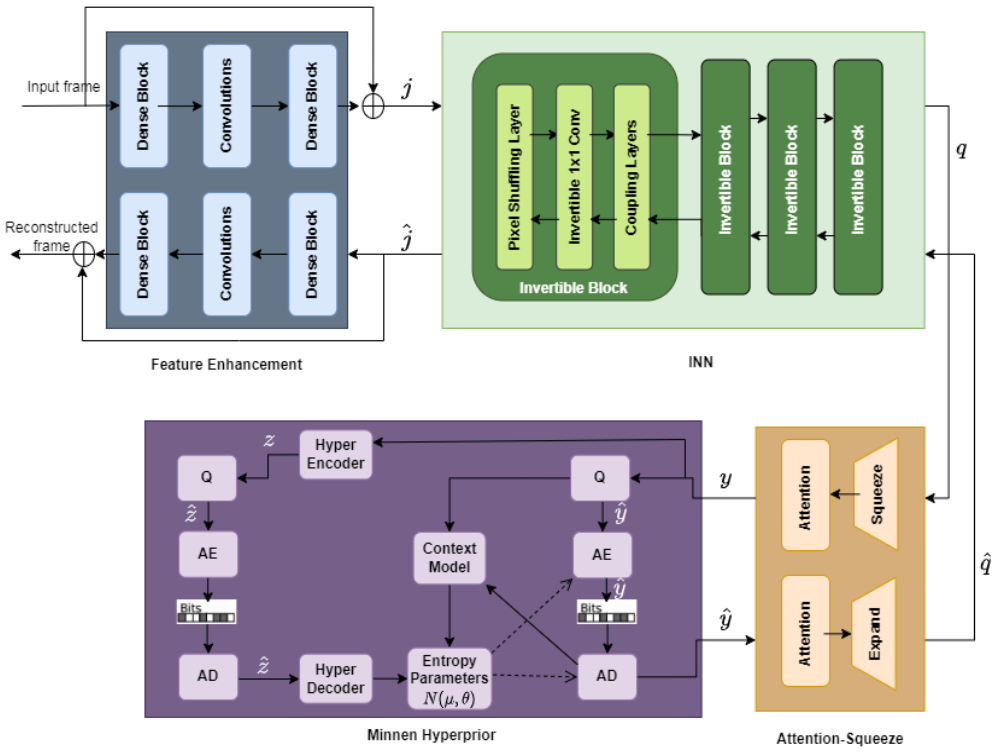


Figure 3.1: Structure of our video compressor

information loss, attention squeeze which is used to stabilize the training process instead of using unstable sampling technique Y. Wang, Xiao, Liu, Zheng, and Liu (2020), and hyperprior which performs analysis/synthesis and entropy coding.

To validate the performance of our method, we tested the model on the YouTube UGC video compression dataset Y. Wang et al. (2019) and reported the results by using PSNR, VMAF, and SSIM quality metrics under a similar setting. The results are reported and compared numerically and visually. The visual results demonstrate that under the same BPP, reconstructed video frames using our method has more clear details.

The paper is organized as follows. The video coding method is described in detail in Section 3.2. Experimental results are presented in Section 3.3, followed by concluding remarks in Section 3.4.

3.2 Approach

The proposed method consists of four modules: feature enhancement, INN, squeeze module, and main module. The architecture is shown in Fig. 3.1. Let $V = \{f_1, f_2, ..., f_T\}$ be the video sequences and f_t be the video frame f at time t. The input frame f has dimensions of (3, H, W). The first step, feature enhancement, adds non-linearity to each input frame and turns it to j with the same dimension of (3, H, W). The second step, INN, turns j to q with dimensions of (3×4^4, $\frac{H}{2^4}$, $\frac{W}{2^4}$) in the forward pass. Third step, attention-squeeze module, leads q to turn into y with dimensions of ($\frac{3\times4^4}{\alpha}$, $\frac{H}{2^4}$, $\frac{W}{2^4}$) in which α is the compression ratio. For the rest of the model, the hyperprior of Minnen et al. (2018) paper is used in which Minnen presented an autoregressive context model with a mean and scale hyperprior; In the hyperprior, using a mean and scale gaussian distribution, the quantized latent features $\hat{y}$ is parameterized with an analysis transform hyper encoder and a synthesis transform hyper decoder. The hyper encoder consists of three convolution layers with Leaky Relu activation between them. Analysis hyper encoder takes y to generate side information z and synthesis hyper decoder takes quantized side information $\hat{z}$. Asymmetric numeral system (ANS) Duda (2009) is used for entropy coding. Each frame loss is calculated based on the following equation Minnen et al. (2018):

$$\begin{aligned} L = R(\hat{y_t}) + R(\hat{z_t}) + \lambda \cdot D(f_t, \hat{f}_t) = \\ E[-\log_2 p_{\hat{y}}(\hat{y_t})] + E[-\log_2 p_{\hat{z}}(\hat{z_t})] + \lambda \cdot D(f_t, \hat{f}_t) \end{aligned} \tag{11}$$

In which rate R is the entropy of quantized latent features, λ is the Lagrange multiplier and its different values correspond to different bit rates, D is the distortion term which can represent mean squared error (MSE) for MSE optimization, or $1 - MS\text{-}SSIM$ for MS-SSIM optimization Z. Wang, Simoncelli, and Bovik (2003), and here it represents mean squared error.

The inverse for the decoding process is as follows: squeeze module copies $\hat{y}$ for alpha times and reshapes it to $\hat{q}$. Then $\hat{q}$ is passed to the INN inversely and turned to $\hat{j}$ and finally, after the inverse of feature enhancement, the reconstructed video with frames $\hat{f}$ is generated. The detail of each module is elaborated in this section:

3.2.1 Feature Enhancement

Feature enhancement module is used to improve the nonlinear representation of the network because INNS are not often capable of nonlinear representation Dinh, Krueger, and Bengio (2015). This module includes a Dense block Huang, Liu, and Weinberger (2017) with input channel size of 3 and output channel size of 64, three convolution layers, and another Dense block with input channel size of 64 and output channel size of 3. Convolution layers have input channel size of 64, output channel size of 64, stride 1, and kernel sizes of 1, 3, and 1.

3.2.2 INN

There are two invertible layers in the INN module which are the down-sampling layer and the coupling layer. A down-sampling layer includes a pixel shuffling layer Shi et al. (2016) and an invertible 1x1 convolution layer Kingma and Dhariwal (2018). Four invertible blocks are utilized for down-sampling and up-sampling similar to the method proposed in Minnen et al. (2018). Each of them consists of one down-sampling layer and three coupling layers. Using these four blocks, the input is down-sampled by 16 times (the down-sampling factor in each pixel shuffling layer is 2). Affine coupling layer Dinh et al. (2016) is defined as following equations:

$$j^{i+1}_{t,1:d} = j^{i}_{t,1:d} \odot exp(\sigma_c(g_2(j^{i}_{t,d+1:D}))) + h_2(j^{i}_{t,d+1:D}) \tag{12}$$

$$j^{i+1}_{t,d+1:D} = j^{i}_{t,d+1:D} \odot exp(\sigma_c(g_1(j^{i+1}_{t,1:d}))) + h_1(j^{i+1}_{t,1:d}) \tag{13}$$

In the above equations, $j^{i}_{t,1:D}$ is the D dimensional input at time frame t to the i_{th} coupling layer which is divided into two parts with dimensions d and D-d. The functions h_1, h_2, g_1, g_1 are Convolutions with stride one with activation functions. Each of them consists of a convolution with kernel size 3, leaky relu, convolution with kernel size 1, leaky relu, and another convolution layer with kernel size 3. Also, $\odot$, exp, and σ_c show the Hadamard product, exponential, and center sigmoid functions respectively.

The inverse process is similar;

$$\hat{j}^{i}_{t,d+1:D} = (\hat{j}^{i+1}_{t,d+1:D} - h_1(\hat{j}^{i+1}_{t,1:d}) \odot exp(-\sigma_c(g_1(\hat{j}^{i+1}_{t,1:d})))) \tag{14}$$

$$\hat{j}^{i}_{t,1:d} = (\hat{j}^{i+1}_{t,1:d} - h_2(\hat{j}^{i}_{t,d+1:D})) \odot exp(-\sigma_c(g_2(\hat{j}^{i}_{t,d+1:D}))) \tag{15}$$

3.2.3 Attention-Squeeze Module

INN does not change the size of input while many of the pixels are useless for compression. So, the channel dimension of the INN's output is reduced through the Squeeze layer. If the output tensor of INN has the size of (D, H, W), it is reshaped into (r, $\frac{D}{r}$, H, W) in which r is the compression ratio. Then, it takes the average on the first dimension to turn the tensor into size ($\frac{D}{r}$, H, W) and passes it to the attention module; Attention module helps the model to pay more attention to challenging parts and reduce the bits of simple parts Z. Cheng et al. (2020). The inverse module in the decompression phase has an attention module and then it copies the quantized tensor r times and reshapes it to the size of (D, H, W).

3.3 Experiments

Experiments are performed on a Tesla V100-SXM2-16GB GPU. The network is trained for 550 epochs using a batch size of 16, learning rates of 0.0001 for the first 450 epochs and 0.00001 for the rest, with Adam optimizer Kingma and Ba (2015) on Pytorch framework.

3.3.1 Dataset

To train our video compression model, the Vimeo-90k dataset is used Xue et al. (2019) which is a large dataset developed for different kinds of video processing tasks. The dataset includes 89800 clips with different contents in different categories. 5000 video clips from different categories are used to train our model. The resolution of the videos is cropped to 256 x 256 pixels.

To test the method and report the results, YouTube UGC Dataset Y. Wang et al. (2019) is used.

There are 1500 video clips with the length of 20 seconds in different categories such as gaming, sports, animation, and lecture. Each clip is available in 4:2:0 YUV format and resolutions of 360P, 480P, 720P, and 1080P. To test the method, we selected 40 videos in different categories with 720P resolution. Also, another public dataset of Ultra Video Group (UVG) Mercat et al. (2020) is used which includes 16 versatile 4K (3840×2160) video sequences. Each video is 50 or 120 frames per second available in 4:2:0 YUV format. To test the videos, the 1920 x 1080 resolution of the dataset is used.

3.3.2 Evaluation Method

To measure the performance of the proposed framework, peak-signal-to-noise ratio (PSNR), video multimethod assessment fusion (VMAF), and structural similarity index measure (SSIM) quality metrics are used. A higher PSNR value shows a higher image quality Horé and Ziou (2010), and SSIM Dosselmann and Yang (2005) measures the similarity between two images and is better correlated with the human perception of distortion. Compared to PSNR and SSIM, VMAF is a subjective measure of the human eye perception and so it is a pivotal measure in real-world applications Rassool (2017).

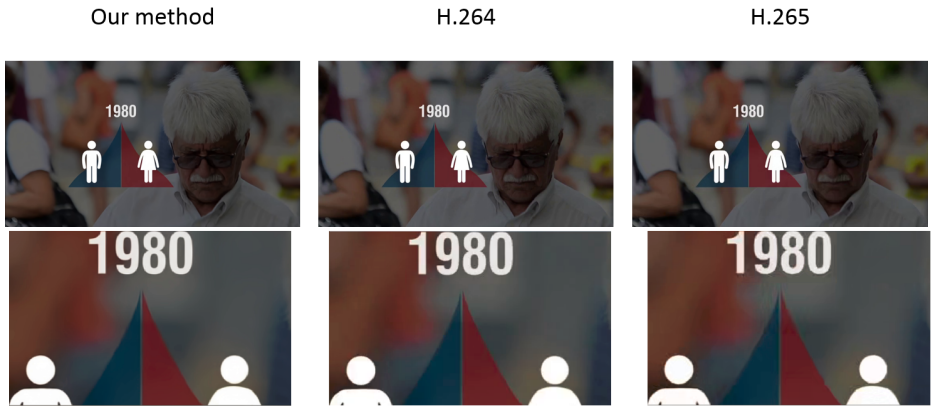
3.3.3 Results

The average PSNR, VMAF, and SSIM of the proposed model on the UGC dataset are 45.5, 98.9, and 0.982, respectively, which outperforms H.264 Wiegand et al. (2003) and H.265 Sullivan et al. (2012) as can be seen in Table 3.1. The average bit per pixel (BPP) for all test sets in the UGC dataset is 0.6.

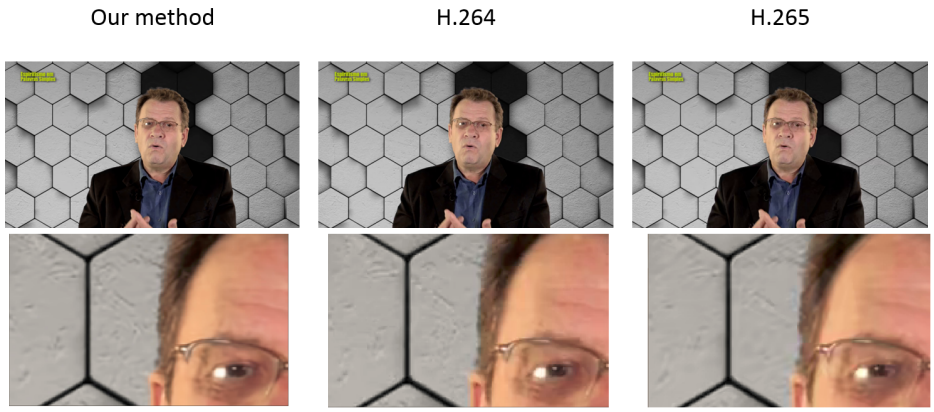
Fig. 3.2 shows samples of the outputs of our framework and H.264 and H.265 on the UGC dataset. In the first example of the lecturer 3.2a, the text on the laptop is sharper and more clear using our method. In the second example 3.2b, the edge of the shapes is more quantized in H.264 and more blurry in H.265. Also, in the third example of a television clip 3.2c, the face of the person is less quantized, sharper, and more clear using our method.

The average PSNR, VMAF, and SSIM of the proposed model on the UVG dataset are 43.9,97.4, and 0.971, respectively, which outperforms H.264 and H.265 as can be seen in Table 3.2. The

(a)



(b)



(c)

Figure 3.2: Samples of the proposed method, H.264, and H.265 outputs. The data used here are from the UGC dataset. (a) The text on the laptop is sharper and more clear using our method. (b) The edge of the shapes is more quantized in H.264 and more blurry in H.265. (c) The face of the person is less quantized, sharper, and more clear using our method.

(a)

(b)

Figure 3.3: Samples of the proposed method, H.264, and H.265 outputs. The data used here are from the UVG dataset. (a) The output is sharper and more clear using our method than H.264 and H.265. (b) The numbers are more quantized and more blurry in H.264 and H.265 than in our method.

Table 3.1: Performance comparison of the methods applied to the YouTube UGC dataset. The proposed method outperforms others in terms of PSNR, VMAF, and SSIM.

Quality Metric	PSNR	VMAF	SSIM
Proposed	**45.5**	**98.9**	**0.982**
H.264	43.9	97.5	0.975
H.265	40.6	93.4	0.96

Table 3.2: Performance comparison of the methods applied to the UVG dataset. The proposed method outperforms others in terms of PSNR, VMAF, and SSIM.

Quality Metric	PSNR	VMAF	SSIM
Proposed	**43.9**	**97.4**	**0.971**
H.264	43.1	94.5	0.96
H.265	41.5	91.5	0.955

average bit per pixel (BPP) for all test samples in the UVG dataset is 0.4.

Fig. 3.3 shows samples of the outputs of our framework and H.264 and H.265 on the UVG dataset. In the first example of the honey bee among flowers 3.3a, the output of H.264 is more quantized and the honey bee in the output of H.265 is more blurry while using our method it is more clear and sharper. In the second example 3.3b, the numbers are more quantized and more blurry in H.264 and H.265 than in our method.

3.4 Conclusion

A video compression method based on INN has been introduced. First, a feature enhancement method is used for enhancing the nonlinear representation. Then, INN is used to decrease the information loss problem. Compared with traditional auto-encoders which lose information in the encoding process, INN can persevere the information and leads to reconstructed videos with more clear details without making the network more complex. To solve the problem of unstable training in INNs, attention-squeeze module is used which makes the feature dimension adjustment stable and tractable.

The results of the method are reported and compared numerically and visually. Evaluations of the proposed method on two standard public datasets show better quality in terms of PSNR, VMAF, and SSIM as compared to the recognized methods such as H.264 and H.265. The visual comparison of the output of the proposed method shows that it has more clear details than the outputs of H.264 and H.265.

Chapter 4

Using ML to Find the Semantic Region of Interest

One of the most challenging problems in computer vision and image processing is the detection of the semantic regions of interest (SRoI). In this chapter, we propose a method using OpenAI's CLIP model to find SRoI by performing semantic search for objects in the image which are detected by an object detection model called Generic RoI Extractor (GRoIE). Finding the semantic regions of interest can be used in different image processing tasks such as image and video compression, enhancement, and reformatting. By knowing the semantic region of interest within images, we can improve the visual quality of images by compressing the more important parts with higher quality and the less important parts, such as the background, with a lower quality. This operation can be achieved without changing the overall compression ratio and the Peak Signal-to-noise Ratio (PSNR) quality metric. Finding the SRoI can make the processes of image enhancement and color correction more accurate by focusing only on the important parts. Moreover, for the image reformatting process, the important parts of the image may be lost. But by using the SRoI, we can reformat the image in a better way by keeping the most important regions in the frame.

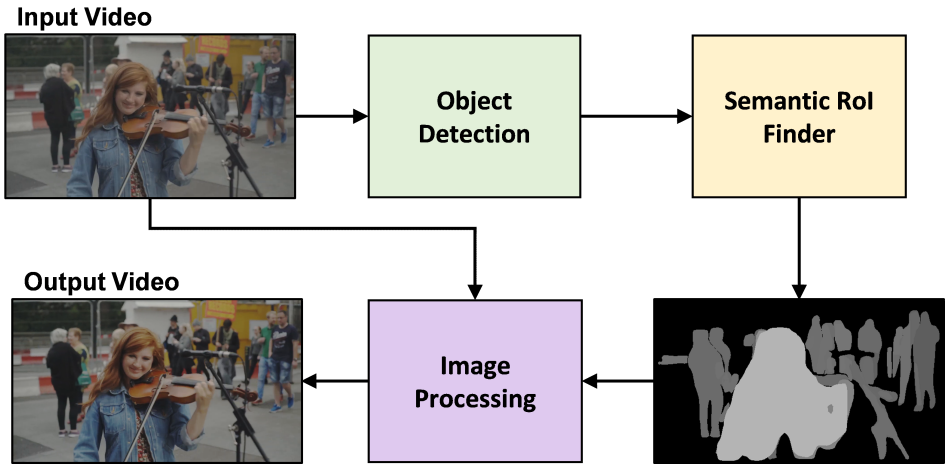


Figure 4.1: Overview of Using Semantic RoI for Image Processing

4.1 Introduction

A recent technique in image processing is finding the Semantic Regions of Interest (SRoI) to help improve the apparent quality of images and videos for various operations. The SRoI contain the objects in the image that are the most important to the overall scene. It can be a person, an object, or a part of a nature scene. Tactics using computer vision techniques to locate regions of interest have existed, such as services provided by Microsoft Azure to generate smart cropped thumbnails Farley, Buck, Sharkey, Christiani, and Kennedy (2022). In computer vision academic publications, saliency maps have been generated using bottom-up visual attention based computational models to identify regions of interest in images to extract thumbnails Amrutha, Shylaja, Natarajan, and Murthy (2009).

A novel method to find semantically interesting regions of images is proposed in this paper using OpenAI's CLIP Radford et al. (2021) model, which is explained in the following section. An overview of the system is depicted in Fig. 4.1. As shown in Fig. 4.1, the input is passed to an object detection module; then CLIP is applied to find the SRoI by running a semantic search among objects and generate a saliency map; finally, the output is generated based on the saliency map and image processing application. To the best of our knowledge, there is no prior work on using OpenAI's CLIP model to perform semantic search just on images, instead of image and text pairs, and generate a saliency map.

The advantages of this method are as follows:

- The CLIP model is trained for different categories of concepts, so the importance map is not biased toward specific objects (e.g., people).
- No additional training is needed.

The second section of this chapter is dedicated to the applications of our method to:

- Image/video compression,
- Image/video enhancement, and
- Image/video reformatting

There are several papers on methods for finding RoI. One work proposed a convolutional neural network (CNN) to find RoI and compress the images according to the found regions Prakash, Moran, Garber, DiLillo, and Storer (2017). In that method, a feature map is generated by summing the top features learned separately for each set of object categories. However, this model does not work well for all types of images and is biased to the classes which were learned. There are many object detection models like Fast R-CNN Girshick (2015), Mask R-CNN He, Gkioxari, Dollár, and Girshick (2020), and Cascade R-CNN Cai and Vasconcelos (2021), which can detect a great variety of objects very well, but they cannot provide the semantic region of interest.

The images of this chapter which are used to show the results are gathered from public internet.

4.2 Approach

In Fig. 4.2, the architecture of our method is shown in detail. In the first module dedicated to object detection, GRoIE Rossi, Karimi, and Prati (2021) model is used. The next module is the CLIP model, which provides object embeddings and image embedding using an image encoder. Then it applies cosine similarity Salton and Buckley (1988) to compare the similarity of the objects to the whole image and to generate an importance map. In the last step, this saliency map can be used for different image processing applications such as image/video compression, enhancement, and reformatting.

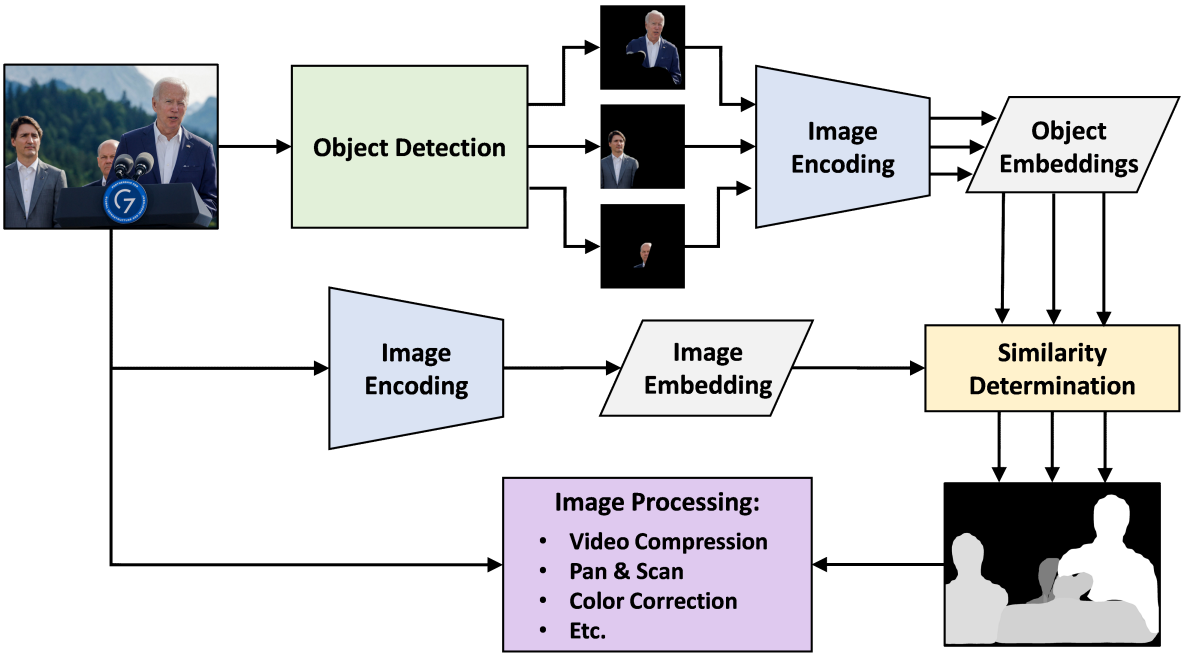


Figure 4.2: Component Diagram of the proposed method for finding the semantic region of interest

4.2.1 Object detection using GRoIE

The first step of our method is to find all the known and unknown objects in the image using an object detection model and create object masks. For this task, we selected GRoIE Rossi et al. (2021) which performs better than other object detection models. The advantage of this model is that it performs well, compared to older models like Mask R-CNN, in detection of unknown objects like abstract sculptures. Fig. 4.3 shows an example of an abstract sculpture which was successfully detected by GRoIE.

As can be observed in Fig. 4.3, although this kind of sculpture is unseen in all data sets categories used for training, GRoIE completely detects it as an object.

4.2.2 Image embedding using OpenAI CLIP

The Contrastive Language-Image Pre-training (CLIP) model was developed in 2021 to perform semantic search on texts and images. It has a text encoder and image encoder to encode images and texts into a 512-dimensional vector embedding. The model was trained on 400 million (image, text) pairs including different categories and concepts. The pretrained model is also available. To

(a) (b)

Figure 4.3: An example of detecting unknown abstract object by GRoIE.

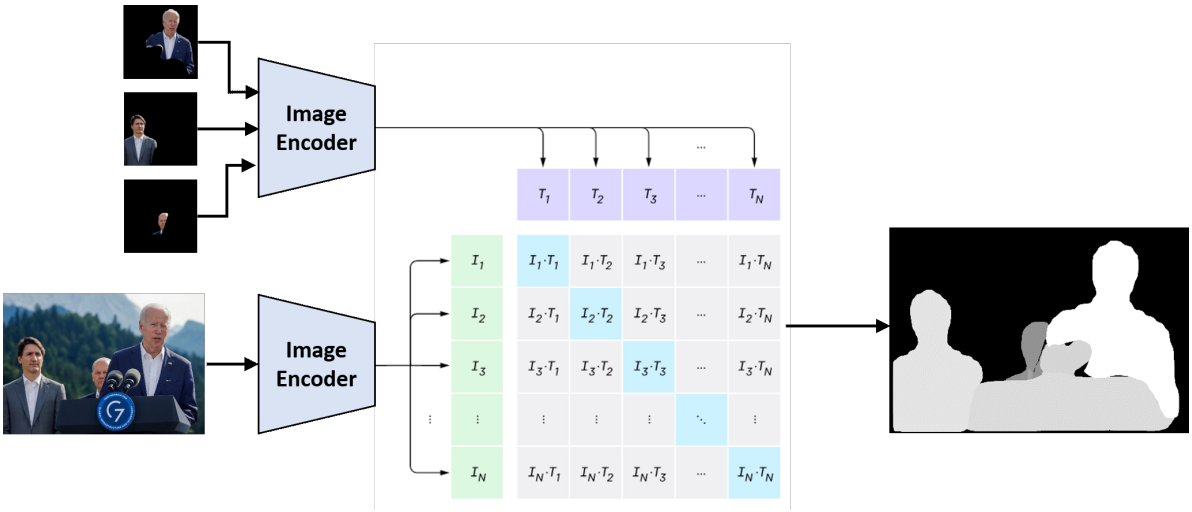
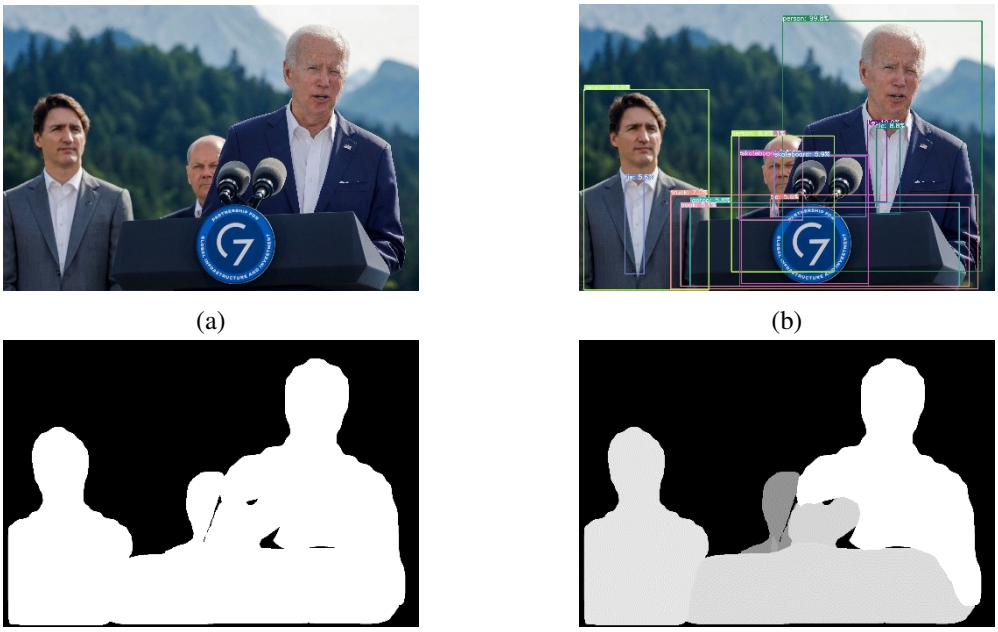


Figure 4.4: Image embedding using Clip model diagram

compare the similarity between image and text embeddings, the cosine similarity is applied, and the highest cosine similarity represents the most related match.

The second step of our method is generating an importance map by comparing each object mask to the whole image and calculating the weighted score of each mask using OpenAI CLIP model. Fig. 4.4 shows this process. Each object mask embedding is generated through CLIP image encoder and is compared to the image embedding. The cosine similarities which are probabilities between 0 and 1 are generated and the object masks are sorted by greyscale color (white is the most important) to produce the saliency maps. If we want to summarize the image in a few words, the similarities show which of the objects are more similar to the concept of image. Because of its ability to semantic search, CLIP can find the similarity and best match within the image. As an



(a) (b)

(c) (d)

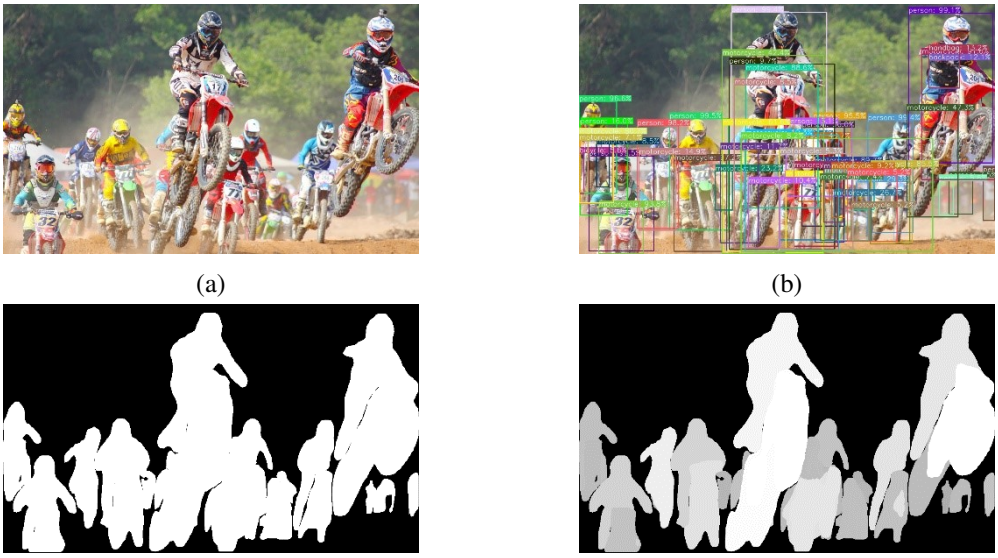
Figure 4.5: Example of Semantic Region of Interest

example, in Fig. 4.5, as the concept of image is about giving a talk, the main speaker (Joe Biden) is more important, however, both Justin Trudeau and Joe Biden are equally famous. Fig. 4.5 and Fig. 4.6 demonstrate some sample images in which GRoIE finds their objects and CLIP generates their semantic saliency maps.

As can be seen in Fig. 4.5, Joe Biden is giving a talk and he is the most interesting person in the whole image as indicated by a white mask in the saliency map. In the same image, Justin Trudeau and the other person are important but less than the main speaker, so they have a grey color in the generated saliency map. In Fig. 4.6, there are some motorcycles that are more important than the background and trees as shown in the saliency map. Two motorcyclists are closer to the camera and so they are brighter on the map.

4.3 Applications and Results

In this section, some image processing applications of our method are elaborated.



(a) (b)

(c) (d)

Figure 4.6: Example of Semantic Region of Interest

4.3.1 Image and video compression

One way to improve the visual quality of compression methods without changing the compression ratio and PSNR is to encode the SRoI parts with a higher bitrate and the rest of the image with a lower bitrate. First, we apply our method to the images to find SRoI. Then we apply the improved JPEG compression Wallace (1992), Prakash et al. (2017) method. First, the number of quality levels is defined as K. The lowest quality level is Q_{low} and the highest quality level is Q_{high} which can be in the range of (1, 100). In our experiment, we defined 9 quality levels from 10 to 90, which are achieved according to the following equation:

$$q_i = Q_{low} + i * \frac{(Q_{high} - Q_{low})}{k} \tag{16}$$

Therefore, the average quality is 50. The generated saliency map based on GRoIE-CLIP model is a greyscale image having pixel values in the range of (0, 255). We normalize the saliency map to the range of (0, 1) and then the level of quality (q_l) which should be used for each part of the image is obtained by:

Comparing an image compressed by standard JPEG with a quality value of 50 with the same image compressed using improved JPEG with average quality of 50 reveals that the visual quality

Algorithm 1

1: **for** $l = 0 \rightarrow K$ **do**
2: **if** $\frac{l}{K} \leq saliency_value \leq \frac{l+1}{K}$ **then**
3: $level = l$
4: **end if**
5: **end for**
6: $q_l = quality[level]$

(a) Original image (b) Our method (c) Standard JPEG

Figure 4.7: A sample image and detail (a) which is compressed using our method (b) and standard JPEG (c)

is enhanced. At the same time, the compression ratio and PSNR remain roughly the same. Fig. 4.7 shows a compressed sample image using our proposed method and the standard JPEG.

As can be observed in Fig. 4.7, in our method, the quality of the image is better in the more interesting area and is lower in the background while the compression ratio, 70:1, is the same in both images. The PSNR metrics for using our method and standard JPEG compression are 33.325 and 34.122, respectively. As these measures show, the goal is not to use traditional compression ratio and quality measured over the whole image to judge the quality, but to perceptually enhance the quality in regions where our natural attention will be, at the cost of quality in other visually less important regions.

4.3.2 Image and video enhancement

The contrast amplitude of an image can be shown using a Bezier histogram which indicates the distribution of luma values. We can enhance the contrast of the image or video by modifying its Bezier curve. There are several known methods for contrast enhancement by modification of the Bezier curve F.-C. Cheng and Huang (2013), Subramani, Bhandari, and Veluchamy (2021). More traditional image processing tactics for image and video enhancements utilize histogram-based techniques such as the global Histogram Equalization Umbaugh (1997), or the more locally adaptive histogram equalization techniques such as CLAHE Pizer et al. (1987).

In this chapter, we focus on the SRoI to enhance the contrast of the image. In this method, the average luminance value of the SRoI is calculated and the lookup table is changed accordingly. Therefore, the image contrast is adjusted to clearly show the SRoI. Fig. 4.8 shows a sample of an enhanced image using the contrast correction scheme focused on the SRoI. In this sample, the woman who plays violin is the SRoI and her hair color and dress color are improved (Fig. 4.8b) with respect to the original (Fig. 4.8a) image. Fig. 4.9 shows the lookup table which is generated according to the SRoI and used to enhance the image in Fig. 4.8. Techniques such as this will be familiar to those in the video and film color correction and visual effects world, where processing masks are used apply filters and localized color corrections.

4.3.3 Image and video reformatting

When we try to use an image or video in landscape format in some social media which only accept portrait format, some parts of the image/video are cropped. Usually, the center remains but we need to keep the other parts of the image inside the frame. Using the SRoI provides this advantage of focusing on interesting parts. To reformat the image using the SRoI, we check it to see which frame has the most semantically important part and keep that frame in the portrait mode. To keep the video smooth over time and remove the abrupt changes of the scene, we use an average sliding window offset of 61 video frames for each frame (30 frames before and 30 frames after the specific frame).

Fig. 4.10 shows the difference of selected offsets according to the SRoI before smoothing and

(a)

(b)

Figure 4.8: A sample of an enhanced image using a Bezier curve modification scheme focused on the SRoI

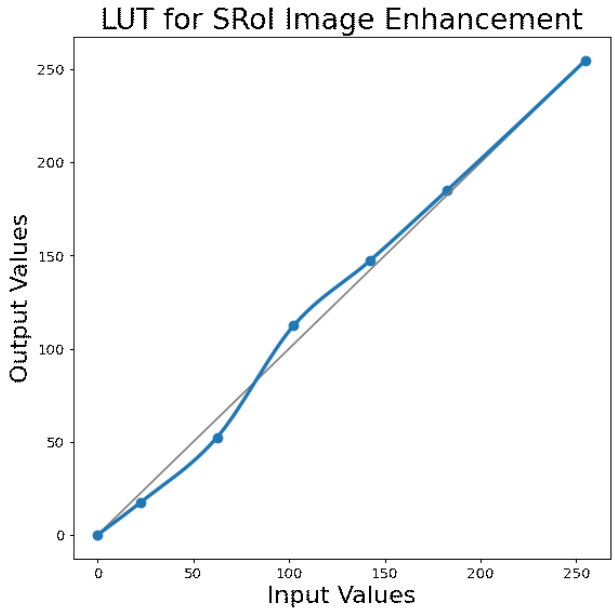
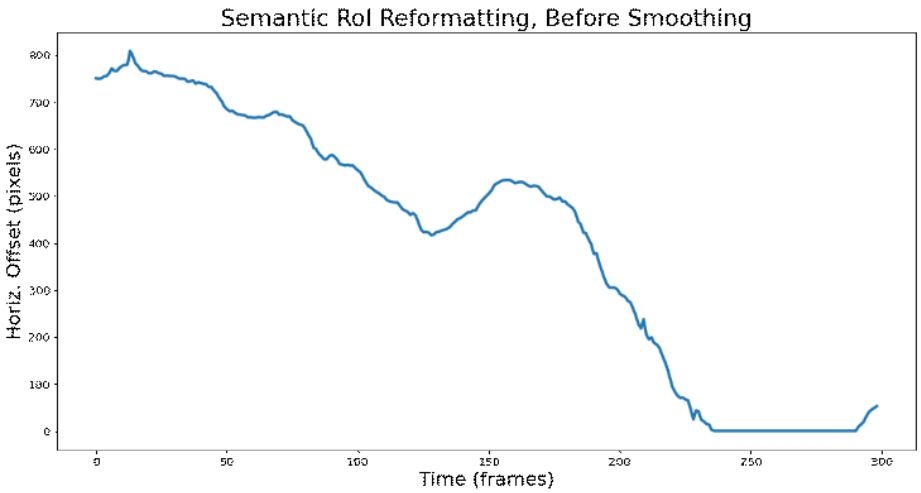
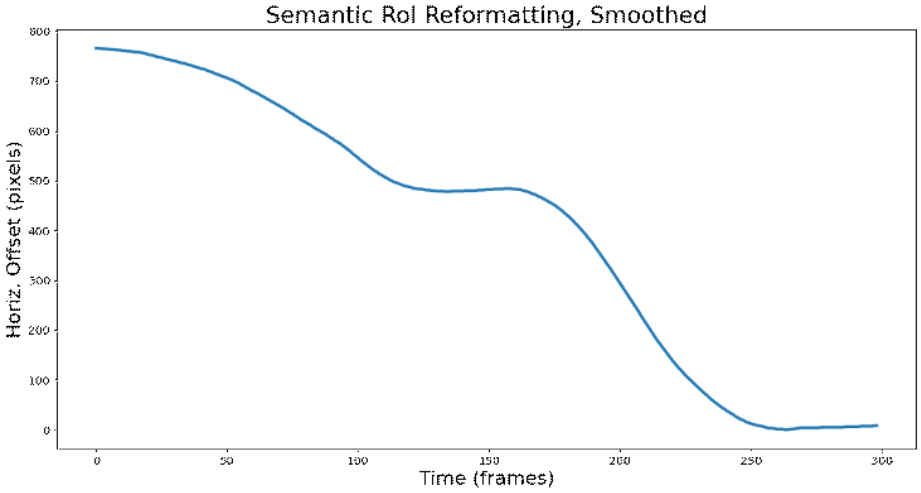


Figure 4.9: Lookup-table used for SRoI image enhancement



(a)



(b)

Figure 4.10: Reformatting offsets before (a) and after smoothing (b).

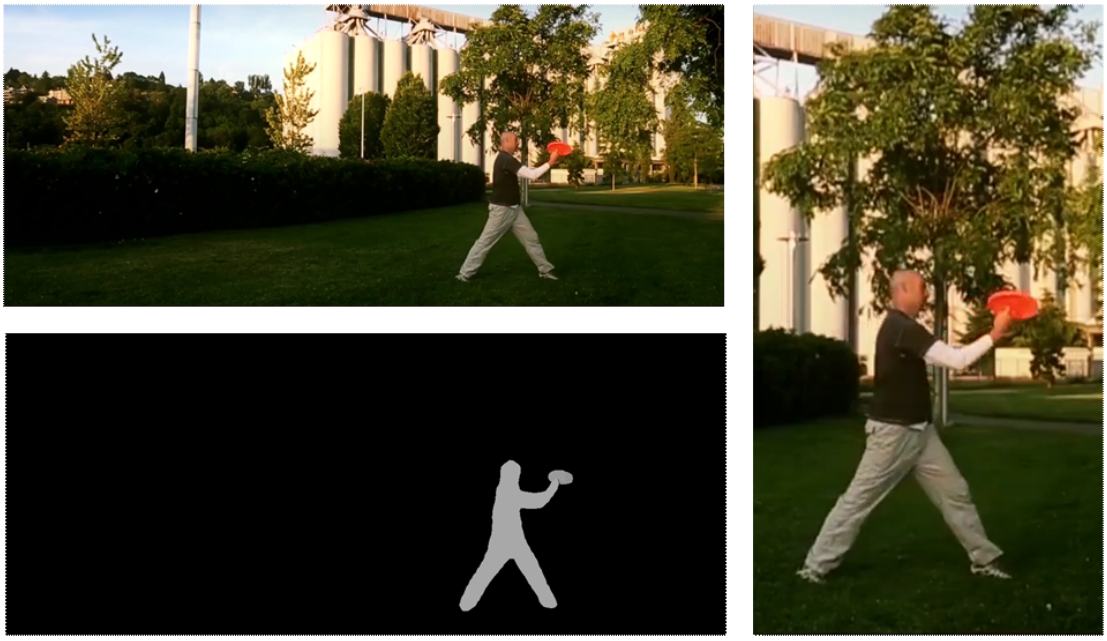


Figure 4.11: The most interesting object in the extracted saliency map for reformatting frames

after smoothing; you can see that the pulses turned into a nice smooth curve. Fig. 4.11 shows the extracted saliency map which indicates the most interesting object for reformatting frame and (the person playing frisbee). Some samples of video frames which are reformatted using our method and regular centering method are shown in Fig. 4.12.

As can be seen in Fig. 4.12, in the regular centering method the person who plays frisbee (SRoI) is out of the frame in some sections of the video while using our method, he remains inside the frame in all frames of the video.

4.4 Conclusion

We have presented a method for extracting the SRoI within images/videos. The method employs GRoIE object detection to extract objects, Open AI's CLIP model to encode objects to embeddings, perform semantic search by comparing the objects' similarity to the whole image using cosine similarity, and find the most important objects within the context of the image. The proposed method does not need any training, and pretrained models are available.

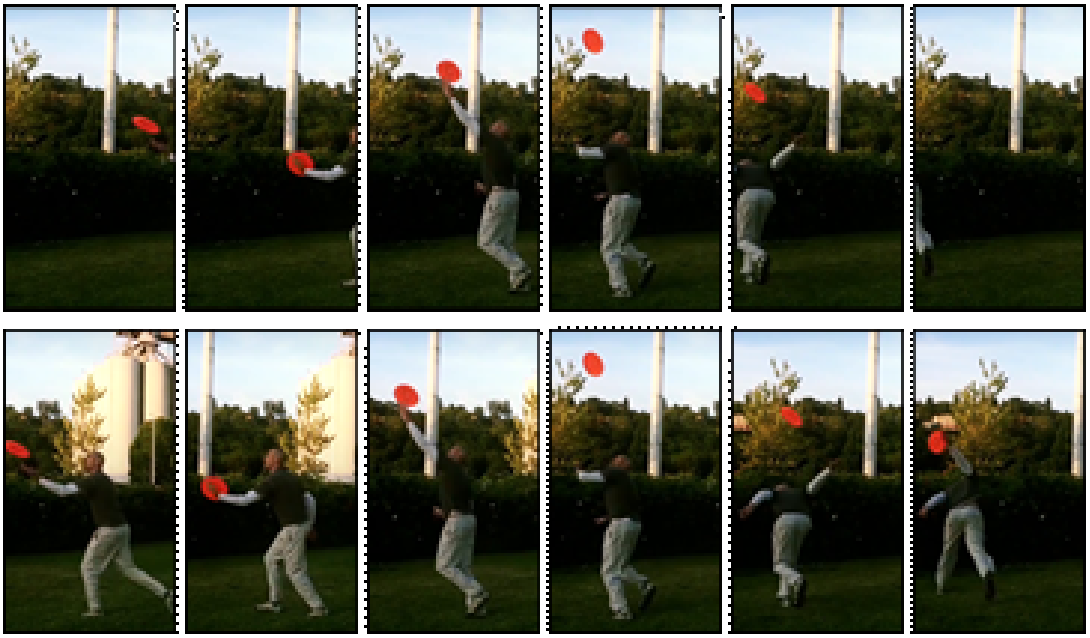


Figure 4.12: Samples of video frames which are reformatted using our method (bottom) and regular centering method (top)

We have also presented three applications that use the new method with example results. Knowing the SRoI, we can compress the image/video with a better visual quality while the compression ratio remains the same. This is done by compressing important parts with higher quality and the less important parts with lower quality. As another application of the proposed method, we can focus on the SRoI in image color correction and reformatting processes. In reformatting the images from landscape to portrait, using the SRoI always keeps the interesting part of the image in the cropped image.

Chapter 5

Conclusion

The research in this thesis concentrated on video compression and finding semantic regions of interest in images/videos using deep learning methods.

Initially, a video compression method based on RNN has been introduced which includes CNN, generalized divisive normalization method, and RNN layers and LSTM cells in encoder and decoder parts. The proposed RNN-based method has less complexity compared to the recent CNN-based methods which are very deeper. Using binarizer for quantization and using LSTM cells in encoder/decoder parts for reducing useless information improved the performance of the proposed model. Our experiments show that the proposed method outperforms the recognized methods such as H.264 and H.265 in terms of PSNR, VMAF, and SSIM.

In another work, a video compression method based on INN has been introduced which uses a feature enhancement method for enhancing the nonlinear representation, INN to decrease the information loss problem, and attention-squeeze module to stabilize feature dimension adjustment in the training process. INN can persevere the information and helps to have reconstructed videos with more clear details without making the network more complex compared to traditional auto-encoders. Experimental results of the proposed method on two standard public datasets show better quality in terms of PSNR, VMAF, and SSIM as compared to the recognized methods such as H.264 and H.265.

In the last research work, a method for extracting the SRoI within images/videos is proposed and three applications of it are presented. GroIE object detection model is used to extract the objects

and Open AI's CLIP model is used to find the most important objects within the context of the image by encoding objects to embeddings and performing semantic search by calculating cosine similarity between objects and the whole image. The pre-trained models are available and so the proposed method does not need any training. By finding SRoI, image/video can be compressed with better visual quality with the same compression ratio. Also, we can perform better color correction by focusing on the SRoI in the image. Furthermore, to reformat images/videos from landscape to portrait, we can use the SRoI to keep the interesting part of the image in the cropped frame. Another image processing application that can be considered as future work is selective blurring to keep the SRoI objects sharp but blurring the less important objects and the background imagery.

Appendix A

List of publications

- Z. Montajabi, V. K. Ghassab and N. Bouguila, ”Invertible Neural Network-Based Video Compression”, Submitted for 2023 International Conference on Pattern Recognition Applications and Methods (ICPRAM).
- Z. Montajabi, V. K. Ghassab and N. Bouguila, ”Recurrent Neural Network-Based Video Compression”, Accepted in 2022 IEEE International Conference on Machine Learning and Applications (ICMLA).
- Z. Montajabi, R. Gonsalves and N. Bouguila, ”Using ML to Find the Semantic Region of Interest”, Accepted in 2022 SMPTE Media Technology Summit.

References

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, *2*(4), 433–459.

Agustsson, E., Mentzer, F., Tschannen, M., Cavigelli, L., Timofte, R., Benini, L., & Van Gool, L. (2017). Soft-to-hard vector quantization for end-to-end learning compressible representations. In *Proceedings of the 31st international conference on neural information processing systems* (p. 1141–1151). Red Hook, NY, USA: Curran Associates Inc.

Agustsson, E., Tschannen, M., Mentzer, F., Timofte, R., & Gool, L. V. (2019). Generative adversarial networks for extreme learned image compression. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 221–231).

Ahmed, N., Natarajan, T., & Rao, K. (1974). Discrete cosine transform. *IEEE Transactions on Computers*, *C-23*(1), 90-93. doi: 10.1109/T-C.1974.223784

Amin, A., Qureshi, H. A., Junaid, M., Habib, M. Y., & Anjum, W. (2011). Modified run length encoding scheme with introduction of bit stuffing for efficient data compression. In *2011 international conference for internet technology and secured transactions* (p. 668-672).

Amrutha, I. S., Shylaja, S. S., Natarajan, S., & Murthy, K. N. B. (2009). A smart automatic thumbnail cropping based on attention driven regions of interest extraction. In S. Sohn et al. (Eds.), *Proceedings of the 2nd international conference on interaction sciences: Information technology, culture and human (ICIS 2009), seoul, korea, 24-26 november 2009* (Vol. 403, pp. 957–962). ACM. Retrieved from https://doi.org/10.1145/1655925.1656099 doi: 10.1145/1655925.1656099

Andrews, H., & Pratt, W. (1968). Fourier transform coding of images. In *Proc. hawaii int. conf.*

system sciences (pp. 677–679).

Apostolico, A., Landau, G., & Skiena, S. (1997). Matching for run-length encoded strings. In *Proceedings. compression and complexity of sequences 1997 (cat. no.97tb100171)* (p. 348-356). doi: 10.1109/SEQUEN.1997.666929

Ardizzone, L., Kruse, J., Wirkert, S. J., Rahner, D., Pellegrini, E. W., Klessen, R. S., ... Köthe, U. (2018). Analyzing inverse problems with invertible neural networks. *CoRR*, *abs/1808.04730*. Retrieved from http://arxiv.org/abs/1808.04730

Ardizzone, L., Lüth, C., Kruse, J., Rother, C., & Köthe, U. (2019). Guided image generation with conditional invertible neural networks. *CoRR*, *abs/1907.02392*. Retrieved from http://arxiv.org/abs/1907.02392

Arif, M., & Anand, R. S. (2012). Run length encoding for speech data compression. In *2012 ieee international conference on computational intelligence and computing research* (p. 1-5). doi: 10.1109/ICCIC.2012.6510185

Avramović, A., & Reljin, B. (2010). Gradient edge detection predictor for image lossless compression. In *Proceedings elmar-2010* (p. 131-134).

Babu, S. A., Eswaran, P., & Kumar, C. S. (2016). Lossless compression algorithm using improved rlc for grayscale image. *Arabian Journal for Science and Engineering*, *41*, 3061-3070.

Ballé, J. (2018). Efficient nonlinear transforms for lossy image compression. In *PCS* (pp. 248–252). IEEE.

Ballé, J., Laparra, V., & Simoncelli, E. P. (2016). Density modeling of images using a generalized normalization transformation. *CoRR*, *abs/1511.06281*.

Ballé, J., Laparra, V., & Simoncelli, E. P. (2017). End-to-end optimized image compression. *ArXiv*, *abs/1611.01704*.

Ballé, J., Minnen, D., Singh, S., Hwang, S. J., & Johnston, N. (2018). Variational image compression with a scale hyperprior. *CoRR*, *abs/1802.01436*.

Bellard, F. (2014). *Bpg image format.* Retrieved 2022-11-29, from https://bellard.org/bpg/

Cai, Z., & Vasconcelos, N. (2021). Cascade r-cnn: High quality object detection and instance

segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*, 1483-1498.

Chen, T., Liu, H., Shen, Q., Yue, T., Cao, X., & Ma, Z. (2017). Deepcoder: A deep neural network based video compression. In *2017 ieee visual communications and image processing (vcip)* (p. 1-4). doi: 10.1109/VCIP.2017.8305033

Cheng, F.-C., & Huang, S.-C. (2013). Efficient histogram modification using bilateral bezier curve for the contrast enhancement. *Journal of Display Technology*, *9*, 44-50.

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2018a). Deep convolutional autoencoder-based lossy image compression. *2018 Picture Coding Symposium (PCS)*, 253-257.

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2018b). Performance comparison of convolutional autoencoders, generative adversarial networks and super-resolution for image compression. In *Cvpr workshops.*

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2019a). Deep residual learning for image compression. In *Cvpr workshops.*

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2019b). Learning image and video compression through spatial-temporal energy compaction. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10063-10072.

Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. (2020). Learned image compression with discretized gaussian mixture likelihoods and attention modules. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7936-7945.

Chung, K.-L., Hsu, T.-C., & Huang, C.-C. (2017). Joint chroma subsampling and distortion-minimization-based luma modification for rgb color images with application. *IEEE Transactions on Image Processing*, *26*(10), 4626-4638. doi: 10.1109/TIP.2017.2719945

Ding, D., Ma, Z., Chen, D., Chen, Q., Liu, Z., & Zhu, F. (2021). Advances in video compression system using deep neural network: A review and case studies. *Proceedings of the IEEE*, *109*(9), 1494-1520. doi: 10.1109/JPROC.2021.3059994

Dinh, L., Krueger, D., & Bengio, Y. (2015). Nice: Non-linear independent components estimation. *CoRR*, *abs/1410.8516*.

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2016). Density estimation using real NVP. *CoRR*,

abs/1605.08803. Retrieved from http://arxiv.org/abs/1605.08803

Dony, R., & Haykin, S. (1995). Neural network approaches to image compression. *Proceedings of the IEEE*, *83*(2), 288-303. doi: 10.1109/5.364461

Dony, R., et al. (2001). Karhunen-loeve transform. *The transform and data compression handbook*, *1*(1-34), 29.

Dosselmann, R., & Yang, X. D. (2005). Existing and emerging image quality metrics. In *Canadian conference on electrical and computer engineering, 2005.* (p. 1906-1913). doi: 10.1109/CCECE.2005.1557355

Duda, J. (2009). Asymmetric numeral systems. *CoRR*, *abs/0902.0271*. Retrieved from http://arxiv.org/abs/0902.0271

Edwards, T. (1991). Discrete wavelet transforms: Theory and implementation. *Universidad de*, 28–35.

Elakkiya, S., & Thivya, K. S. (2021). Comprehensive review on lossy and lossless compression techniques. *Journal of The Institution of Engineers (India): Series B*.

Farley, P., Buck, A., Sharkey, K., Christiani, T., & Kennedy, D. (2022). *Smart-cropped thumbnails.* Retrieved 2022-11-17, from https://learn.microsoft.com/en-us/azure/cognitive-services/computer-vision/concept-generating-thumbnails?tabs=3-2

Gavrilova, Y. (2021). *What are convolutional neural networks?* Retrieved 2022-11-29, from https://serokell.io/blog/introduction-to-convolutional-neural-networks

Girshick, R. B. (2015). Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1440-1448.

Golomb, S. (1966). Run-length encodings (corresp.). *IEEE Transactions on Information Theory*, *12*(3), 399-401. doi: 10.1109/TIT.1966.1053907

Guo, Y., Lu, C., Allebach, J. P., & Bouman, C. A. (2017). Model-based iterative restoration for binary document image compression with dictionary learning. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (p. 606-615). doi: 10.1109/CVPR.2017.72

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2020). Mask r-cnn. *IEEE Transactions on Pattern*

Analysis and Machine Intelligence, *42*(2), 386-397. doi: 10.1109/TPAMI.2018.2844175

He, K., Zhang, X., Ren, S., & Sun, J. (2016, jun). Deep residual learning for image recognition. In *2016 ieee conference on computer vision and pattern recognition (cvpr)* (p. 770-778). Los Alamitos, CA, USA: IEEE Computer Society. Retrieved from https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.90 doi: 10.1109/CVPR.2016.90

Hochreiter, S., & Schmidhuber, J. (1997, 11). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780. Retrieved from https://doi.org/10.1162/neco.1997.9.8.1735 doi: 10.1162/neco.1997.9.8.1735

Horé, A., & Ziou, D. (2010). Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition* (p. 2366-2369). doi: 10.1109/ICPR.2010.579

Huang, G., Liu, Z., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261-2269.

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, *40*(9), 1098-1101. doi: 10.1109/JRPROC.1952.273898

Hussain, A., Al-Fayadh, A., & Radi, N. (2018). Image compression techniques: A survey in lossless and lossy algorithms. *Neurocomputing*, *300*, 44-69. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925231218302935 doi: https://doi.org/10.1016/j.neucom.2018.02.094

An image format for the web. (n.d.). Retrieved 2022-11-29, from https://developers.google.com/speed/webp

Islam, K., Dang, L. M., Lee, S., & Moon, H. (2021). Image compression with recurrent neural network and generalized divisive normalization. In *2021 ieee/cvf conference on computer vision and pattern recognition workshops (cvprw)* (p. 1875-1879). doi: 10.1109/CVPRW53098.2021.00209

Jain, A., & Lakhtaria, K. I. (2016). Comparative study of dictionary based compression algorithms on text data. *International Journal of Computer Science and Network Security (IJCSNS)*, *16*(2), 88.

Jiang, F., Tao, W., Liu, S., Ren, J., Guo, X., & Zhao, D. (2018). An end-to-end compression framework based on convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*, *28*(10), 3007-3018. doi: 10.1109/TCSVT.2017.2734838

Jiang, J. (1999). Image compression with neural networks – a survey. *Signal Processing: Image Communication*, *14*(9), 737-760. Retrieved from https://www.sciencedirect.com/science/article/pii/S0923596598000411 doi: https://doi.org/10.1016/S0923-5965(98)00041-1

Johnston, N., Vincent, D., Minnen, D., Covell, M., Singh, S., Chinen, T., ... Toderici, G. (2018). Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4385–4393).

Kabir, M. A., & Mondal, M. R. H. (2017). Edge-based transformation and entropy coding for lossless image compression. In *2017 international conference on electrical, computer and communication engineering (ecce)* (p. 717-722). doi: 10.1109/ECACE.2017.7912997

Khaitan, S., & Agarwal, R. (2019). Multi-fractal image compression. In *2019 international conference on machine learning, big data, cloud and parallel computing (comitcon)* (p. 340-345). doi: 10.1109/COMITCon.2019.8862190

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980*.

Kingma, D. P., & Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, *31*.

Klopp, J. P., Wang, Y., Chien, S.-Y., & Chen, L.-G. (2018). Learning a code-space predictor by exploiting intra-image-dependencies. In *British machine vision conference.*

Kothuri, S., Annapurna, P., & Lukka, S. (2013, 08). Digit recognition using freeman chain code. *International Journal of Application or Innovation in Engineering Management 2319 - 4847*, *2*, 362-365.

Krishna, K., Ramakrishnan, K., & Thathachar, M. (1997). Vector quantization using genetic k-means algorithm for image compression. In *Proceedings of icics, 1997 international conference on information, communications and signal processing. theme: Trends in information*

systems engineering and wireless multimedia communications (cat. (Vol. 3, pp. 1585–1587).

Lee, J., Cho, S., & Beack, S.-K. (2019, May). Context-adaptive entropy model for end-to-end optimized image compression. In *the 7th int. conf. on learning representations.*

Li, M., Zuo, W., Gu, S., Zhao, D., & Zhang, D. (2018). Learning convolutional networks for content-weighted image compression. In *2018 ieee/cvf conference on computer vision and pattern recognition* (p. 3214-3223). doi: 10.1109/CVPR.2018.00339

Li, Y., Liu, D., Li, H., Li, L., Wu, F., Zhang, H., & Yang, H. (2018). Convolutional neural network-based block up-sampling for intra frame coding. *IEEE Transactions on Circuits and Systems for Video Technology*, *28*, 2316-2330.

Lin, C.-H., Chung, K.-L., & Fang, J.-P. (2014). Adjusted 4:2:2 chroma subsampling strategy for compressing mosaic videos with arbitrary rgb color filter arrays in hevc. *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*, 1-7.

Liu, D., Li, Y., Lin, J., Li, H., & Wu, F. (2021, jan). Deep learning-based video coding. *ACM Computing Surveys*, *53*(1), 1–35. Retrieved from https://doi.org/10.1145%2F3368405 doi: 10.1145/3368405

Liu, Y. K., Wei, W., Jie Wang, P., & Žalik, B. (2007). Compressed vertex chain codes. *Pattern Recognition*, *40*(11), 2908-2913. Retrieved from https://www.sciencedirect.com/science/article/pii/S0031320307001082 doi: https://doi.org/10.1016/j.patcog.2007.03.001

Lugmayr, A., Danelljan, M., Gool, L. V., & Timofte, R. (2020). Srflow: Learning the super-resolution space with normalizing flow. In *European conference on computer vision* (pp. 715–732).

Ma, S., Zhang, X., Jia, C., Zhao, Z., Wang, S., & Wang, S. (2019). Image and video compression with neural networks: A review. *IEEE Transactions on Circuits and Systems for Video Technology*, *30*(6), 1683–1698.

Martucci, S. (1990). Reversible compression of hdtv images using median adaptive prediction and arithmetic coding. In *Ieee international symposium on circuits and systems* (p. 1310-1313 vol.2). doi: 10.1109/ISCAS.1990.112371

Menassel, R., Nini, B., & Mekhaznia, T. (2018). An improved fractal image compression using wolf pack algorithm. *Journal of Experimental & Theoretical Artificial Intelligence*, *30*(3), 429-439. Retrieved from https://doi.org/10.1080/0952813X.2017.1409281 doi: 10.1080/0952813X.2017.1409281

Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., & Van Gool, L. (2018). Conditional probability models for deep image compression. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4394–4402).

Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., & Van Gool, L. (2019). Practical full resolution learned lossless image compression. In *2019 ieee/cvf conference on computer vision and pattern recognition (cvpr)* (p. 10621-10630). doi: 10.1109/CVPR.2019.01088

Mentzer, F., Toderici, G., Tschannen, M., & Agustsson, E. (2020). High-fidelity generative image compression. In *Proceedings of the 34th international conference on neural information processing systems*. Red Hook, NY, USA: Curran Associates Inc.

Mercat, A., Viitanen, M., & Vanne, J. (2020). Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. *Proceedings of the 11th ACM Multimedia Systems Conference*.

Minnen, D., Ballé, J., & Toderici, G. D. (2018). Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, *31*.

Motta, G., Storer, J. A., & Carpentieri, B. (2000). Lossless image coding via adaptive linear prediction and classification. *Proceedings of the IEEE*, *88*(11), 1790–1796.

Murn, L., Blasi, S., Smeaton, A., & Mrak, M. (2021). Improved CNN-based learning of interpolation filters for low-complexity inter prediction in video coding. *IEEE Open Journal of Signal Processing*, *2*, 453–465. Retrieved from https://doi.org/10.1109%2Fojsp.2021.3089439 doi: 10.1109/ojsp.2021.3089439

Murn, L., Blasi, S., Smeaton, A. F., O'Connor, N. E., & Mrak, M. (2020). Interpreting cnn for low complexity learned sub-pixel motion compensation in video coding. In *2020 ieee international conference on image processing (icip)* (p. 798-802). doi: 10.1109/ICIP40778.2020.9191193

Networking, C. V. (2016). Cisco global cloud index: Forecast and methodology, 2015-2020. white paper. *Cisco Public, San Jose*, 2016.

Ohm, J.-R., & Sullivan, G. J. (2018). Versatile video coding–towards the next generation of video compression. In *Picture coding symposium* (Vol. 2018).

Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., ... Zuiderveld, K. (1987). Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, *39*(3), 355-368. Retrieved from https://www.sciencedirect.com/science/article/pii/S0734189X8780186X doi: https://doi.org/10.1016/S0734-189X(87)80186-X

Prakash, A., Moran, N., Garber, S., DiLillo, A., & Storer, J. A. (2017). Semantic perceptual image compression using deep convolution networks. *2017 Data Compression Conference (DCC)*, 250-259.

Pratt, W., Kane, J., & Andrews, H. (1969). Hadamard transform image coding. *Proceedings of the IEEE*, *57*(1), 58-68. doi: 10.1109/PROC.1969.6869

Pu, I. M. (2006). Chapter 1 - introduction. In I. M. Pu (Ed.), *Fundamental data compression* (p. 1-17). Oxford: Butterworth-Heinemann. Retrieved from https://www.sciencedirect.com/science/article/pii/B9780750663106500040 doi: https://doi.org/10.1016/B978-075066310-6/50004-0

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *CoRR*, *abs/2103.00020*. Retrieved from https://arxiv.org/abs/2103.00020

Rahman, M., & Mohamed, M. (2022, 07). A prediction-based lossless image compression procedure using dimension reduction and huffman coding. *Multimedia Tools and Applications*, 1-25. doi: 10.1007/s11042-022-13283-3

Rassool, R. (2017). Vmaf reproducibility: Validating a perceptual practical video quality metric. In *2017 ieee international symposium on broadband multimedia systems and broadcasting (bmsb)* (p. 1-2). doi: 10.1109/BMSB.2017.7986143

Rippel, O., & Bourdev, L. D. (2017). Real-time adaptive image compression. In *International conference on machine learning*.

Rossi, L., Karimi, A., & Prati, A. (2021). A novel region of interest extraction layer for instance

segmentation. *2020 25th International Conference on Pattern Recognition (ICPR)*, 2203-2209.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing Management*, *24*(5), 513-523. Retrieved from https://www.sciencedirect.com/science/article/pii/0306457388900210 doi: https://doi.org/10.1016/0306-4573(88)90021-0

Sanchez-Cruz, H., & Rodriguez-Dagnino, R. M. (2005). Compressing bilevel images by means of a three-bit chain code. *Optical Engineering*, *44*(9), 097004. Retrieved from https://doi.org/10.1117/1.2052793 doi: 10.1117/1.2052793

Santurkar, S., Budden, D., & Shavit, N. (2018). Generative compression. In *2018 picture coding symposium (pcs)* (pp. 258–262).

Sharma, K., & Gupta, K. (2017). Lossless data compression techniques and their performance. In *2017 international conference on computing, communication and automation (iccca)* (p. 256-261). doi: 10.1109/CCAA.2017.8229810

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., ... Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1874–1883).

Shukla, J., Alwani, M., & Tiwari, A. K. (2010). A survey on lossless image compression methods. In *2010 2nd international conference on computer engineering and technology* (Vol. 6, p. V6-136-V6-141). doi: 10.1109/ICCET.2010.5486344

Skodras, A., Christopoulos, C., & Ebrahimi, T. (2001). The jpeg 2000 still image compression standard. *IEEE Signal Processing Magazine*, *18*(5), 36-58. doi: 10.1109/79.952804

Subramani, B., Bhandari, A. K., & Veluchamy, M. (2021). Optimal bezier curve modification function for contrast degraded images. *IEEE Transactions on Instrumentation and Measurement*, *70*, 1-10. doi: 10.1109/TIM.2021.3073320

Sullivan, G. J., Ohm, J.-R., Han, W.-J., & Wiegand, T. (2012). Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1649-1668. doi: 10.1109/TCSVT.2012.2221191

Theis, L., Shi, W., Cunningham, A., & Huszár, F. (2017). Lossy image compression with compressive autoencoders. *ArXiv*, *abs/1703.00395*.

Toderici, G., O'Malley, S. M., Hwang, S. J., Vincent, D., Minnen, D., Baluja, S., ... Sukthankar, R. (2016). Variable rate image compression with recurrent neural networks. In Y. Bengio & Y. LeCun (Eds.), *4th international conference on learning representations, ICLR 2016, san juan, puerto rico, may 2-4, 2016, conference track proceedings*. Retrieved from http://arxiv.org/abs/1511.06085

Toderici, G., Vincent, D., Johnston, N., Hwang, S. J., Minnen, D., Shor, J., & Covell, M. (2017). Full resolution image compression with recurrent neural networks. In *CVPR* (pp. 5435–5443). IEEE Computer Society.

Umbaugh, S. E. (1997). *Computer vision and image processing: A practical approach using cviptools with cdrom* (1st ed.). USA: Prentice Hall PTR.

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on machine learning* (p. 1096–1103). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/1390156.1390294 doi: 10.1145/1390156.1390294

Wallace, G. (1992). The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, *38*(1), xviii-xxxiv. doi: 10.1109/30.125072

Wang, Y., Inguva, S., & Adsumilli, B. (2019). Youtube ugc dataset for video compression research. *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, 1-5.

Wang, Y., Xiao, M., Liu, C., Zheng, S., & Liu, T. (2020). Modeling lost information in lossy image compression. *CoRR*, *abs/2006.11999*. Retrieved from https://arxiv.org/abs/2006.11999

Wang, Z., Simoncelli, E., & Bovik, A. (2003, 12). Multiscale structural similarity for image quality assessment. In (Vol. 2, p. 1398 - 1402 Vol.2). doi: 10.1109/ACSSC.2003.1292216

Wiegand, T., Sullivan, G., Bjontegaard, G., & Luthra, A. (2003). Overview of the h.264/avc video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*(7), 560-576. doi: 10.1109/TCSVT.2003.815165

Witten, I. H., Neal, R. M., & Cleary, J. G. (1987, jun). Arithmetic coding for data compression. *Commun. ACM*, *30*(6), 520–540. Retrieved from https://doi.org/10.1145/214762.214771 doi: 10.1145/214762.214771

Xiao, M., Zheng, S., Liu, C., Wang, Y., He, D., Ke, G., ... Liu, T.-Y. (2020). Invertible image rescaling. In *European conference on computer vision* (pp. 126–144).

Xie, Y., Cheng, K. L., & Chen, Q. (2021). Enhanced invertible encoding for learned image compression. In *Proceedings of the 29th acm international conference on multimedia* (p. 162–170). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3474085.3475213 doi: 10.1145/3474085.3475213

Xue, T., Chen, B., Wu, J., Wei, D., & Freeman, W. T. (2019, feb). Video enhancement with task-oriented flow. *International Journal of Computer Vision*, *127*(8), 1106–1125. Retrieved from https://doi.org/10.1007%2Fs11263-018-01144-2 doi: 10.1007/s11263-018-01144-2

Žalik, B., Mongus, D., Liu, Y.-K., & Lukač, N. (2016). Unsigned manhattan chain code. *Journal of Visual Communication and Image Representation*, *38*, 186-194. Retrieved from https://www.sciencedirect.com/science/article/pii/S1047320316300050 doi: https://doi.org/10.1016/j.jvcir.2016.03.001