

ROBUST CONTENT IDENTIFICATION AND DE-DUPLICATION WITH SCALABLE
FISHER VECTOR IN VIDEO WITH TEMPORAL SAMPLING

A THESIS IN
Electrical Engineering

Presented to the Faculty of the University
of Missouri-Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
LAKSHMI PRASANNA GADIPARTHI
B. TECH., Acharya Nagarjuna University, 2014

Kansas City, Missouri
2017

© 2017

LAKSHMI PRASANNA GADIPARTHI

ALL RIGHTS RESERVED

ROBUST CONTENT IDENTIFICATION AND DE-DUPLICATION WITH SCALABLE
FISHER VECTOR IN VIDEO WITH TEMPORAL SAMPLING

- A VIDEO PROCESSING TECHNIQUE USING *MATLAB*

Lakshmi Prasanna Gadiparthi, Candidate for the Master of Science Degree

University of Missouri-Kansas City, 2017

ABSTRACT

Robust content identification and de-duplication of video content in networks and caches have many important applications in content delivery networks. In this work, we propose a scalable hashing scheme based Fisher Vector aggregation of selected key point features, and a frame significance function based non-uniform temporal sampling scheme on the video segments, to create a very compact binary representation of the content fragments that is agnostic to the typical coding and transcoding variations. The key innovations are a key point repeatability model that selects the best key point features, and a non-uniform sampling scheme that significantly reduces the bits required to represent a segment, and scalability from PCA feature dimension reduction and Fisher Vector features, and Simulation with various frame size and bit rate video contents for DASH streaming are tested and the proposed solution have very good performance of precision-recall, achieving 100% precision in duplication detection with recalls at 98% and above range.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Computing and Engineering have examined a thesis titled ‘Robust Content Identification and De-Duplication with Scalable Fisher Vector In video with Temporal Sampling’ presented by Lakshmi Prasanna Gadiparthi, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Zhu Li, Ph. D., Committee Chair

Department of Computer Science & Electrical Engineering

Sejun Song, Ph.D.,

School of Computing and Engineering

Cory Beard Ph.D.,

School of Computing and Engineering

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF ILLUSTRATIONS.....	vii
ACKNOWLEDGEMENTS.....	ix
1 INTRODUCTION.....	1
2 SOFTWARE DESCRIPTION	6
2.1 Overview of MATLAB:.....	6
2.2 Features of MATLAB:.....	6
3 IMAGE PROCESSING	8
4 SIFT FEATURE EXTRACTION	10
4.1 Introduction.....	10
4.2 Feature.....	11
4.2.1 Types of Features	11
4.3 Feature Detection	12
4.4 Feature Extraction.....	12
4.5 SIFT Feature Extraction.....	122
5 PRINCIPAL COMPONENT ANALYSIS.....	16
5.1 Introduction.....	16
5.2 Analysis.....	16
5.3 Eigen Values and Eigen Vectors.....	18
5.4 Dimension Reduction.....	21
6 FISHER VECTOR AGGREGATION	23
6.1 Fisher Vector.....	23
6.2 Fisher Kernel.....	24
7 De-duplication	26
7.1 Introduction.....	26
7.2 Hashing	28
8 SIMULATION RESULTS AND DISCUSSIONS	29
9 CONCLUSION AND FUTURE WORK.....	37
9.1 Conclusion	37
9.2 Future Work	37

APPENDIX.....	38
BIBILOGRAPHY.....	41
VITA.....	44

LIST OF ILLUSTRATIONS

Figure	Page
1-1: Process of Fisher Aggregation of video.....	3
1-2: Process of deduplication using hashing.....	4
4-1: Detection of Feature Descriptors.....	12
4-2: Scale-space Extreme Detection.....	13
4-3: SIFT for different octaves of the image.....	13
4-4: Representation of Keypoint in image.....	14
5-1: Triangles in oval shape arrangement.....	17
5-2: Triangles in oval shape with projected lines.....	17
5-3: Triangles projecting on Horizontal Line.....	18
5-4: The principal component of the oval.....	19
5-5: Representing Principal Component with splitting line.....	19
5-6: Representation of second Eigen vector.....	20
5-7: Framing the data into Eigen vectors.....	20
5-8: Three Dimensional Representation of data points.....	21
5-9: Representation of three eigen vectors with zero eigen value.....	21
5-10: Representation of Dimension Reduction.....	22
7-1: Efficient video storage techniques.....	27
8-1: Process of Installation.....	29
8-2 : Installation of FFMPEG.....	30
8-3: Checking the installation of FFMPEG.....	30
8-4: Conversion of video into images with resolution 320×240.....	31
8-5: Conversion of video into images with resolution 640×360.....	32
8-6: Plot of Frame Significance.....	34

8-7: Plot of cumulative frame significance	34
8-8: ROC curve for fisher aggregation of dataset with resolution 640×360	35
8-9: ROC curve for Sequence Level	36

ACKNOWLEDGEMENTS

Primarily, I am thankful to Almighty for the well-being and health that were required to complete this research. It is my pleasure to acknowledge everyone who were with me in this work.

I would like to convey my deep gratitude to my advisor Dr. Zhu Li for his immense knowledge, patience and for continuous support provided by him. I am thankful to him for his continuous guidance to complete this research. I extend my gratitude to Dr. Cory Beard and Dr. Sejun Song for spending their valuable time for providing the feedback by reviewing my work.

I am very thankful to Nancy Hoover for helping me in formatting my dissertation and I want to extend my sincere thanks to all my lab mates and friends who helped me to simulate the thoughts for this research work.

Finally, I must express my gratitude to my parents Mr. Gadiparthi Sambasiva Rao and Mrs. Gadiparthi Narayamma for providing me support throughout my years of study. I am very thankful to my brothers and sisters for continuous encouragement through the process of researching and writing this thesis.

I would like to express my sincere thanks to everyone who helped me directly or indirectly in this research.

CHAPTER 1

INTRODUCTION

As many techniques are developing in internet like softwares for video editing, devices for video capturing, online videos are growing rapidly. Out of which the duplicate videos are growing exponentially. There we have a requirement to manage those videos and to protect the intellectual rights. Many duplicate videos are being uploading and sharing everyday which is leading to the need of duplicate detection. The duplicate detection technique must be robust in such a way that it should detect the duplicate videos even after transforming them into different forms.

In an uncontrolled video event and action, recognition is very challenging due to some variations like duration of action performed. Background Clutter is the other factor in image recognition. Motion clutter and camera modulation caused by moving background objects confronts the performance of the system. Evaluating high-level features leads to complexity. Therefore, instead of those complex models, we evaluate the low-level features. For low-level descriptors, we use Fisher vectors by which aggregation is performed. The purpose of this work is to provide the low-level descriptors using MPEG Compact Descriptors for Video Analysis (CDVA). In this, some key frames are used by which global and local features are extracted. Then global descriptor similarity is used for temporal sampling.

In this work, we propose temporal sampling of fisher vector for video, which is taken from a video dataset. An approach to describe an image for retrieval and classification is to extract local and global descriptors, encoding them into vector of high dimension and then generating a signature. As our dataset is a video file, I used a freeware software FFMPEG to convert into set of images. Thumbnails are used to generating the signature, which evaluates

the most significant values. Then for SIFT descriptors CDVS is used. Compact Descriptors for Visual Search (CDVS) is initiated by the Moving Picture Experts Group (MPEG) to extract the compact descriptors.

The main objective of CDVS standard is to provide detailed description of images. MPEG CDVS has many breakthroughs for the performance to be high and for less complexity. CDVS is applicable in visual search applications using SIFT descriptors. In this, the local descriptors are quantized into set of elements with prototype. In this work, instead of encoding Fisher Kernel framework is proposed in which we use Gaussian Mixture Model (GMM). This representation is called Fisher Vector and it is efficient in terms of complexity and performance.

In this work, the local features which describes the image effectively are mapped into global features and then the hashing is performed on the aggregated global features. Then by using average number of frames for sequence level matching we can find the effectiveness of the proposed technique.

For the evaluation of efficiency of this system ROC curves are plotted. ROC curves have a characteristic named area under curve which is commonly called as AUC. It is mainly used in the analysis of the classification systems. It is plotted by using true and false positive rates. The ideal area under curve for a good classifier is 1. The ideal classifier can classify the inputs in a better way under any circumstances. If the area under curve is less than 0.5 then it may not be useful in any application. The explanation for every process is explained in the following chapters whereas the complete process is explained in the two block diagrams which are shown below.

FISHER VECTOR AGGREGATION PROCESS

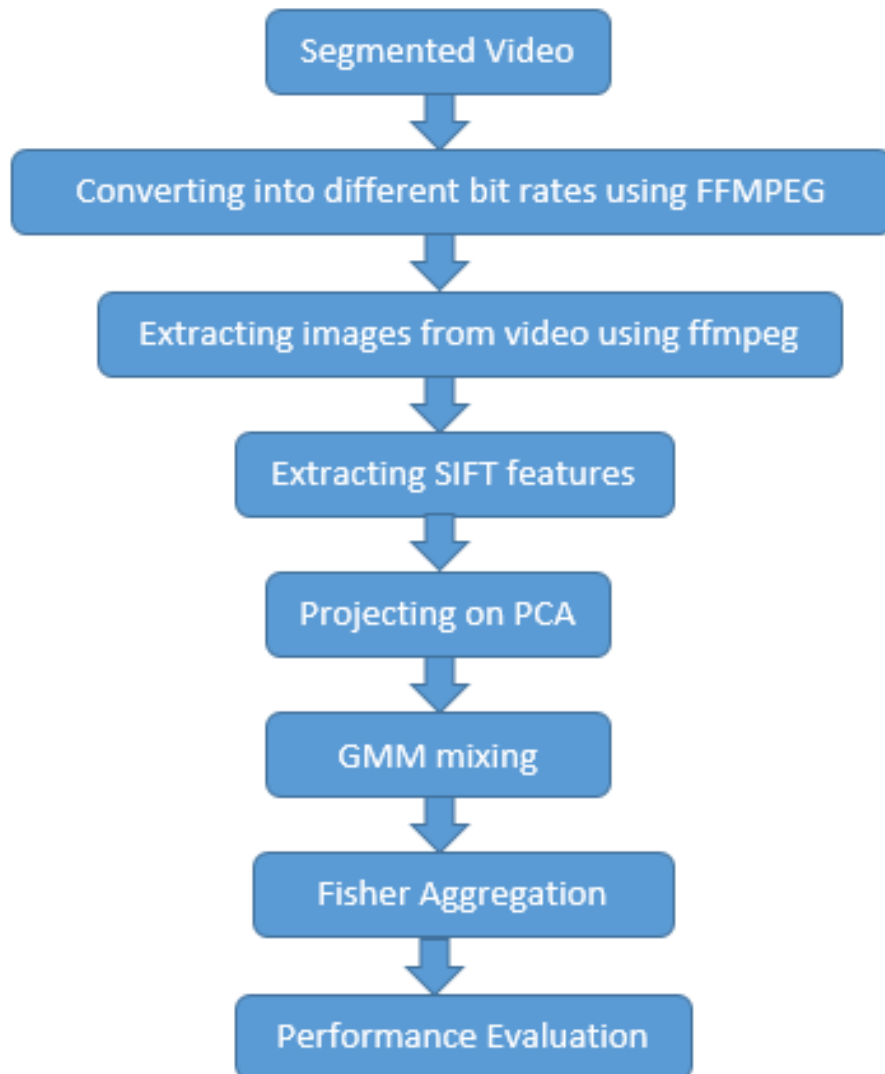


Figure 1-1: Process of Fisher Aggregation of video

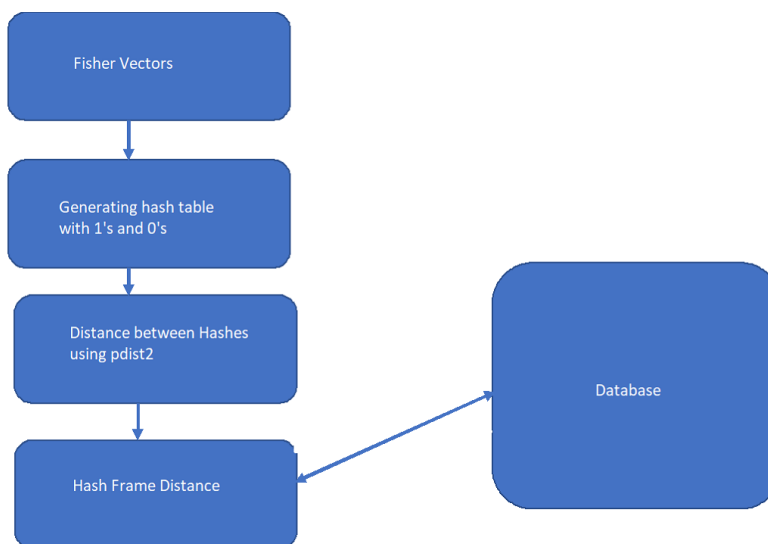


Figure 1-2: Process of deduplication using hashing

The software programs we have used in implementing this model are discussed as a part of chapter 2. Apart from these explanations, other software tools that are used in this paper are explained in this chapter.

In this thesis work, the third chapter deals with the explanation of dataset of images used in this paper. Along with that, we are going to discuss the processing of the dataset using prescribed software.

The fourth chapter in this work focusses on the features of the dataset and the need for feature extraction. This chapter includes a detailed explanation of selection of the important features of images in dataset. This deals with feature detection and feature extraction. In addition to that, it includes the most prominent technique for feature extraction.

The work in fifth chapter especially focusses on the PCA (Principal Component Analysis) which is used to differentiate the items in dataset by finding the most unique features of those items.

The sixth chapter deals with the Gaussian Mixture Model, which is used for classification of images. The fisher vector and its aggregation is explained in chapter seven. In the final chapter, the analysis procedure I had followed during my entire thesis work has been clearly explained.

Seventh chapter explains the fisher vectors and fisher aggregation. Eighth chapter explains deduplication followed by hashing scheme. The followed chapter shows the simulation results and discussions. The conclusion and future work are explained in next chapter.

CHAPTER 2

SOFTWARE DESCRIPTION

2.1 Introduction

MATLAB is a tool developed by Cleve Moler *et al.*, 1970 at the University of New Mexico. Fortran libraries are used for numerical computing. FORTRAN language is used to access the FORTRAN libraries. He invented MATLAB to give his students easy access to FORTRAN libraries, which for numerical computing without writing FORTRAN. Then he co-founded Math Works with Jack Little. MATLAB provides more efficient and high performance technical computing environment. It has large library set of math and graphics functions. It is language which is understandable to human. In MATLAB text file is stored as “.m” as its extension. M-file is created by using m editor on MATLAB. The execution of file is done in command window.

MATLAB is the acronym for Matrix Laboratory, which is developed by Math Works. It provides complete numerical computing environment. It allows manipulations, computations and plotting of data. MATLAB supports Object-Oriented Programming which includes classes, in-heritance, and pass-by-value and pass-by-reference semantics. In MATLAB, the applications can be developed by using Graphical User Interface. It has integrated graph-plotting features.

2.2 Features of MATLAB

MATLAB is preferred over other programming languages based on the following differences:

- Command Style like the one Linux Commands are supported by MATLAB by which we can call some functions without the use of parenthesis. Instead of these

parenthesis spaces can be used to separate a function and its arguments. There is no need to use single quotation mark to call the function.

- The use of ; is mandatory in other languages whereas in MATLAB its use is optional.
- The MATLAB statements can be continued by three dots (...) which is different from other languages.
- The cost of convenience is efficient in MATLAB.
- Instead of packages in all other languages, MATLAB contains toolbox. There is no need to include the libraries to include a specific function in MATLAB. We can use those functions as long as the Toolbox is installed.
- It is only case sensitive for in-built names and functions.
- Loops in MATLAB is faster than loops in other languages.
- It is very easy language. In Matlab, I would create a new Matlab file called test.m and in it I would write this as:

```
I=imread('image.png');
```

```
Imshow(I)
```

- **Speed:** Matlab is a pretty high-level scripting language, meaning that you don't have to worry about memory management or other lower-level programming issues. However, this is due to the fact that Matlab is built on Java, and Java is built upon C.
- **Simple Code:** Matlab features a very concise and informative help section in their IDE. Here, each function and its usage is outlined, along with some sample code demonstrating their use. The standard image processing functions are available along with a number of the newer image processing methods bundled in the various toolboxes

CHAPTER 3

IMAGE PROCESSING

An image can be defined as a two-dimensional function, $f(x, y)$ where x and y are spatial co-ordinates. The amplitude of 'f' at any pair of co-ordinates (x, y) is called the intensity or gray-level of the image at that point. Digital image processing is an ever expanding and dynamic area with applications reaching out into our everyday life such as medicine, space exploration, surveillance, authentication, automated industry inspection and many more areas. Implementing such applications on a general-purpose computer can be easier, but not very time efficient due to additional constraints on memory and other peripheral devices.

Digital image is preferred because of its Good quality for storage and transmission Interactivity, Variable- rate transmission on demand easy software conversion from one standard to another Integration of various video applications Editing capabilities, such as cutting and pasting, zooming, removal of noise and blur, Robustness to channel noise and ease of encryption, Trend of DSP Problem with a digital video system.

Any digital image is composed of a finite number of elements. Each element has a location and value. These elements are referred to as picture elements, image elements or pixels. Images play a significant role in human perception. However, human beings are limited to only the visual band of the electromagnetic spectrum.

Digital image processing is the use of computer algorithms to create process, communicate and display digital images. The processing algorithms of digital image can be used to:

- Convert signals from image sensor into digital images.

- Improve clarity and remove noise from images.
- Extract the size, scale or number of objects in a scene.
- Prepare images for printing or display.
- Compress images for communication across a network.

CHAPTER 4

SIFT FEATURE EXTRACTION

4.1 Introduction

Computer Vision is a field, which deals with the way of gaining high-level understanding from digital images or videos in computers. According to engineering, it is similar to the human visual system. The tasks in computer vision includes acquiring, processing, analyzing and understanding of digital images in order to produce symbolic or numerical information in the form of decisions. The Visual images, which are the input of the retina, can interface with other processes through descriptions. Computer Vision is the theory to extract the information from images. The image data can be various forms like video sequences, data from medical scanner, multi-dimensional data or different views from cameras. Sub-domains of computer vision include event detection, reconstruction, video tracking, object recognition, image restoration and motion estimation.

Computer vision began with artificial intelligence. It was a human visual system to endow robots with intelligent behavior. Applications ranges from machine vision to artificial intelligence. Medical computer vision is most prominent application, which is known as medical image processing. Image data in medical image processing is in the form of angiography images, X-ray images, tomography images and ultrasonic images. Applications of computer vision includes enhancement of images. For instance, X-rays are processed to reduce the noise. Computer vision is the second application area, which is sometimes called as machine vision. This application is supported by manufacturing process.

4.2 Feature

Definition of feature depends on the type of application. Feature is the most interesting part of an image and these are starting point for many computer vision algorithms. Feature is distinctive attribute of image. For instance, in number plate recognition system the features in number plate are alphabets or numerical in the license plate.

4.2.1 Types of Features

According to the type of application, there are few types of features like edges, corner, blob and feature description.

Edges:

Edges are the intensive parts of the images. Each and every pixel in the image is checked for the intensity and the pixels with high intensity are declared as edges. The Edges are detected by using techniques like Canny and Sobel.

Corners:

As corner is the common point of two edges, the intersection of two most intensive points is a corner. The corners can be detected by using Harris detector which is rotation invariant.

Blob:

Each and every pixel is checked for its properties like brightness, color, intensity. If the properties of one pixel changes from other pixels it is declared as blob. Laplacian detectors generally detect it. These are all different types of features which are extracted by using a specific technique. Each type of detection has its own specific algorithm.

4.3 Feature Detection

An algorithm is used to extract the features. There are two types of features like local and global features. Local features describe the complex features whereas global ones describe the external features. The unique features of an image are detected by using sift technique or hog techniques.

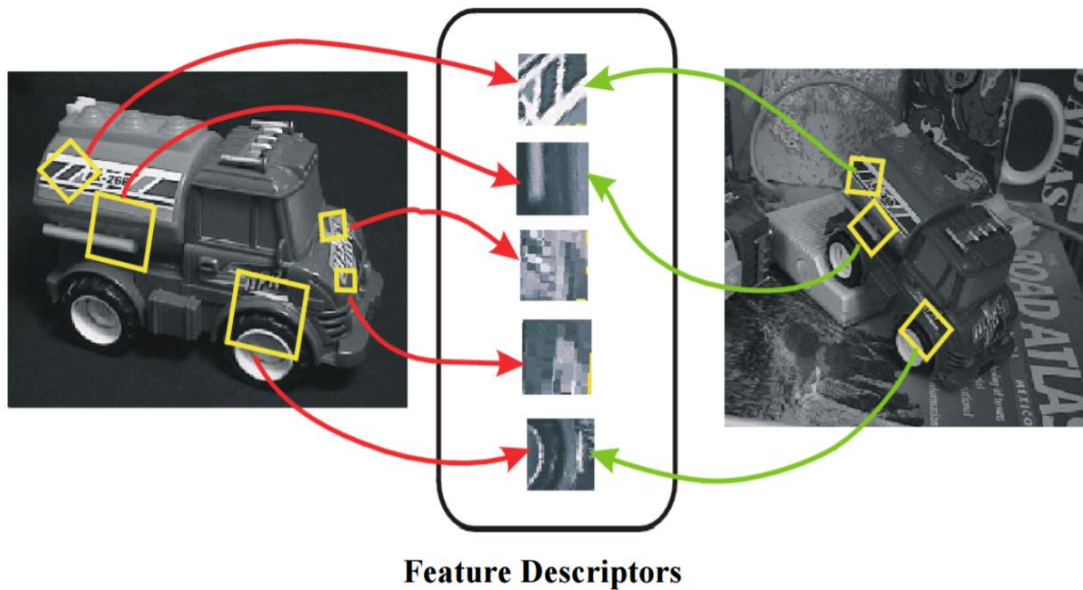


Figure 4-1: Detection of Feature Descriptors

In the above figure, the most important features are extracted and those features are represented as feature descriptors.

4.4 Feature Extraction

When input is too large to perform image-processing techniques, feature extraction is used. The most important features are detected from the dataset and then the processing is done on those features. Transforming the input data into set of features is Feature Extraction.

4.5 SIFT Feature Extraction

In 2004, D.Lowe from University of British Columbia came up with a new algorithm for image scaling called Scale Invariant Feature Transform (SIFT).

Different steps involved in this algorithm are as below.

1. Detection on scale-space

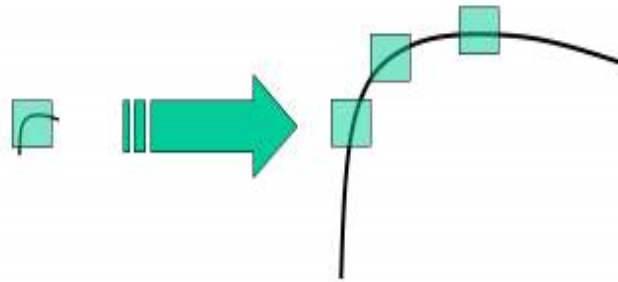


Figure 4-2: Detection on scale-space

In this figure, we cannot detect the keypoints using the same window. It is OK with small corner. Filtering of type scale-space is used. LoG acts as a blob detector, which blobs in various sizes due to change in σ which acts as a scaling parameter. For instance, in the above image gaussian kernel with high value for small corner while Gaussian kernel with high σ sets for large corners. Local maxima is calculated for the keypoints. SIFT for images in a pyramid which is in gaussian representation is done in different octaves as below.

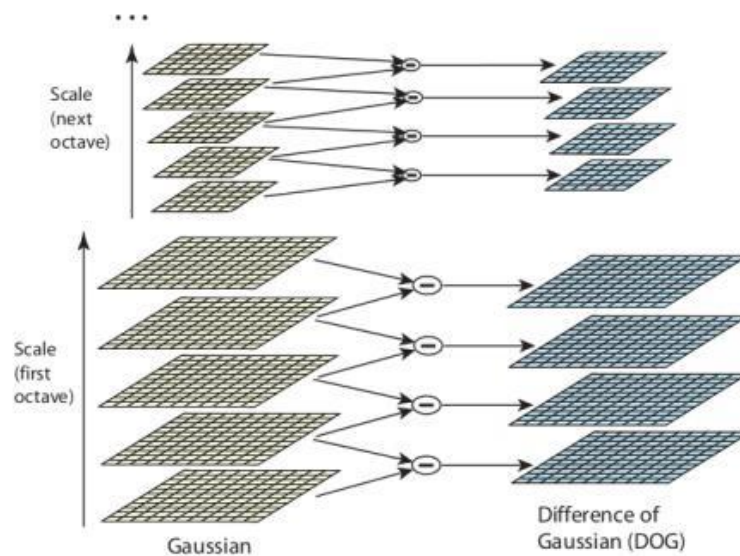


Figure 4-3: Different Octaves

Then image search is performed over given space. For example, one pixel in an image is compared with its eight neighbors as well as nine pixels in next scale and nine pixels in previous scales. Potential keypoint is detected if it is local extrema on which a keypoint is represented. It is shown in below image:

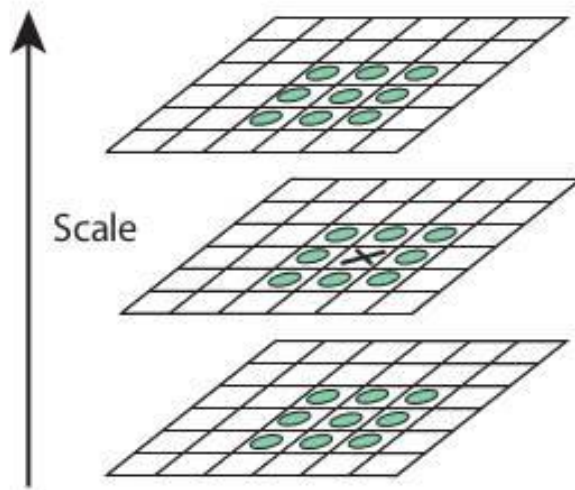


Figure 4-4: Representation of Keypoint in image

2. Keypoint Localization

Once potential keypoints locations are found, they must be refined to get results that are more accurate. The important points in an image are called keypoints. To extract the features, we must extract the important points. This is a local feature representation which are very complex. Taylor series is used to get correct positions. It is mainly based on the intensity and a threshold is set for this process. If the given point has intensity more than threshold, it is considered as key point. The points with high intensity are treated as corners and edges. The orientation is assigned according to the requirement.

3. Keypoint Descriptor

After the creation of keypoint descriptor, a neighborhood is created around that keypoint. It is divided into 16 sub-blocks of 4x4 size. Then half a size of histogram is created which results in bin values which is created as descriptor. It is represented as robust descriptor against climate changes and many other variations.

4. Keypoint Matching

It is often used for matching. If the keypoints of two different images are matched they can be treated as same images. This is used in some classifier technique. As local features are unique features for a image, they can be solely used for classification process.

CHAPTER 5

PRINCIPAL COMPONENT ANALYSIS

5.1 Introduction

Principal components analysis is a procedure for identifying a smaller number of uncorrelated variables, called "principal components", from a large set of data. The goal of principal components analysis is to explain the maximum amount of variance with the fewest number of principal components. Principal components analysis is commonly used in the social sciences, market research, and other industries that use large data sets. Principal components analysis is commonly used as one-step in a series of analyses. PCA diminishes the variables which reduces multicollinearity and observations.

Eigen values and vectors are used to analyze the PCA. This paper will give a very broad overview of PCA, describing eigenvectors and eigenvalues and showing how one can reduce the dimensions of data using PCA. It is a neat tool to use in information theory, and even though the math is a bit complicated, we only need to get a broad idea of what's going on to be able to use it effectively.

5.2 Analysis

PCA is used to measure the data w.r.t principle components. In general, they are represented in x-y axis. PCA is used to represent the structure of the data as shown below.

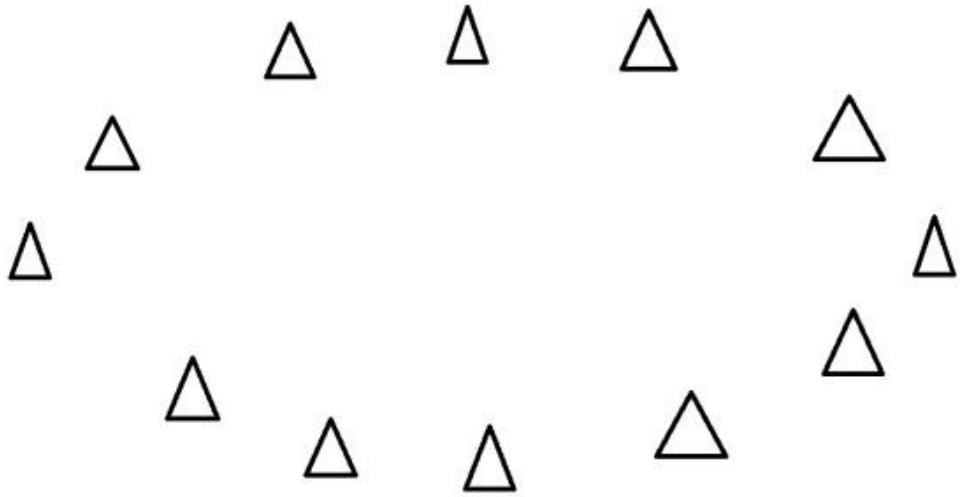


Figure 5-1: Triangles in oval shape

Let us consider these triangles as point of data. To find PCA, we must find the underlying structure. To do this we must analyze the data directions. In this example, a straight line which is vertical is chosen to analyze the directions.

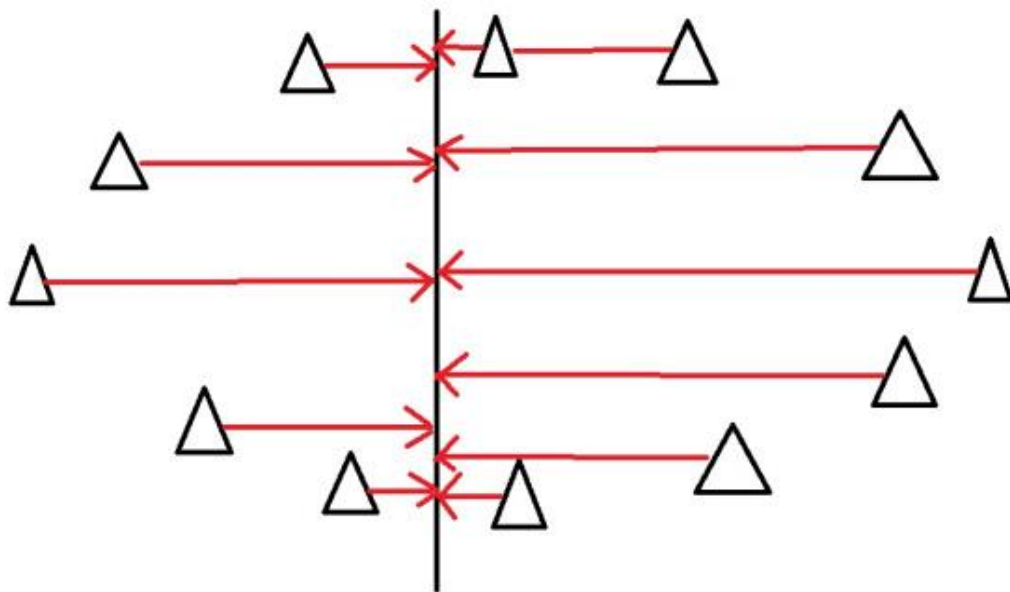


Figure 5-2: Triangles in oval shape with projected lines

The data is not very spread out here; therefore, it does not have a large variance. So, the vertical line solely not enough to find the principal component. So, we are projecting a horizontal line which is as below.

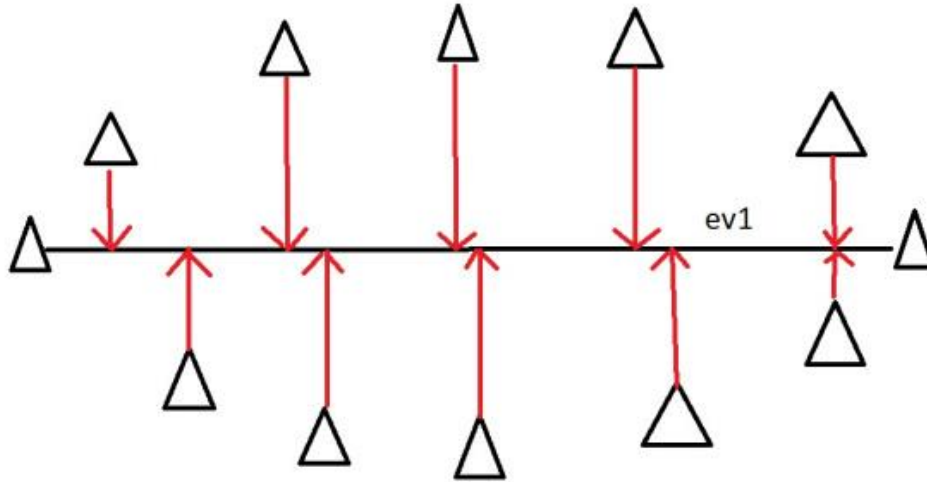


Figure 5-3: Triangles projecting on horizontal line

On this line, the data is more spread out, it has a large variance. In this example, all the data points are not directly connected to the vertical line whereas they are directly connected to horizontal line. So, horizontal line is good enough to represent these data points. This works for only content aware datasets. If we are content unaware we have to use some math which are called as eigen values and vectors.

5.3 Eigen Values and Eigen Vectors

We can compose the data points into eigen values and vectors. They always exist in couples. Every eigen vector has its own eigen value. Eigen vector represents the direction whereas eigen value represents the alignment of the data in that direction. Principal component is the eigenvector which has high eigen value.

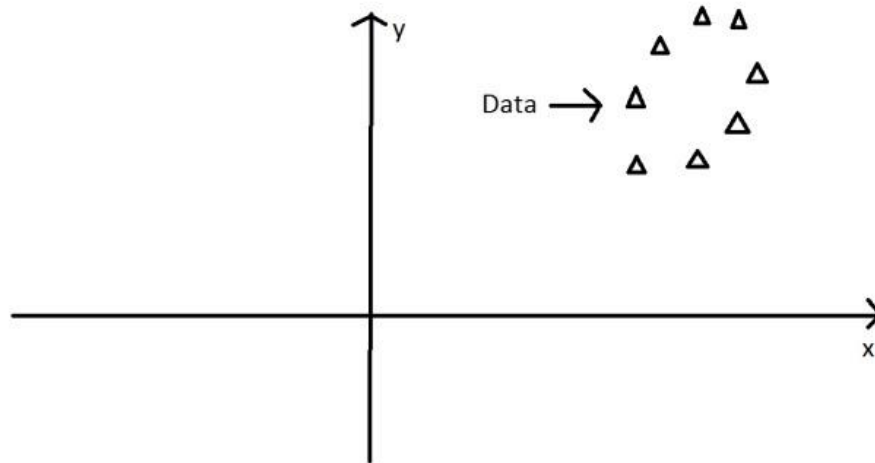


Figure 5-4: The principal component of the oval

At the moment, the oval is on an x-y axis. But the principal component of this data is splitting in many ways as shown.

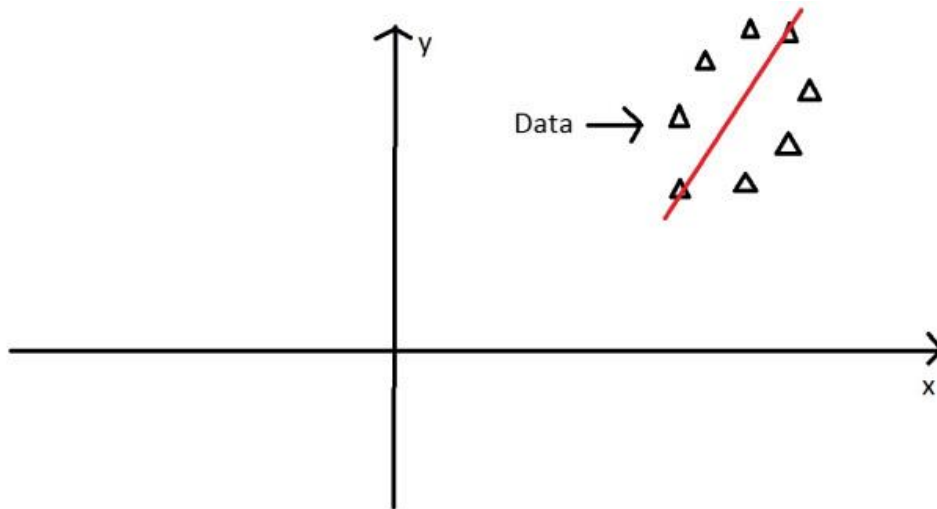


Figure 5-5: Representing Principal Component with splitting line

But this vector is not enough to represent so we consider another vector to cover both x-y area.

So, we consider another line which is perpendicular to it. So, the eigen vector is as below.

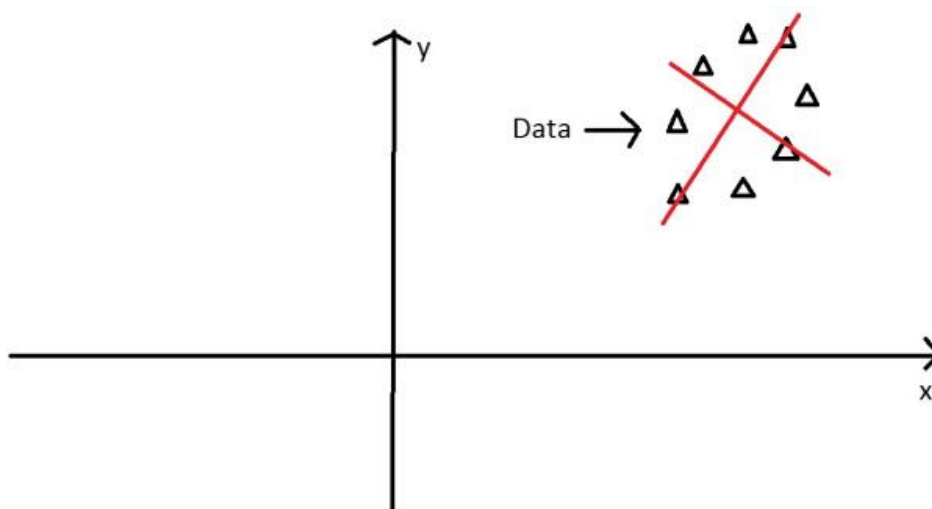


Figure 5-6: Representation of second Eigen vector

After covering both x and y axis, we frame the dataset into these dimensions as shown below.

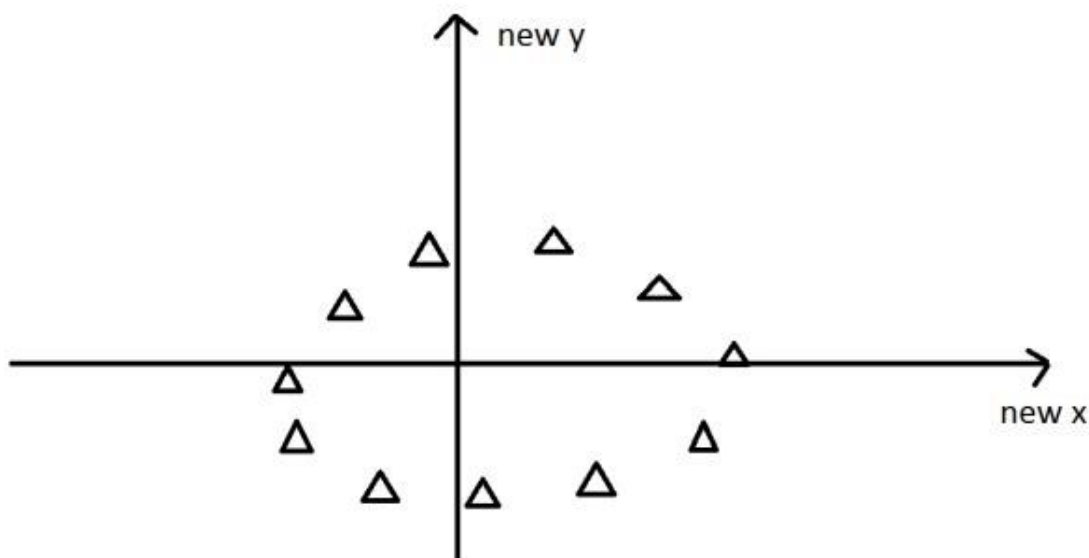


Figure 5-7: Framing the data into Eigen vectors

Up to now we just analyzed the data. We can shape the data now as we framed the complete data into the axes. When there is no data, the eigen value for the vector is zero. So, we have to shape the data in a better way.

5.4 Reduction of Dimension

Now, we should diminish the data set dimensions. Diminishing is nothing but reducing. It is stripping the unwanted dimensions.

If there are 3 variables so it is a 3D data set. In a 3D data, we have x, y and z axes. The representation is as below.

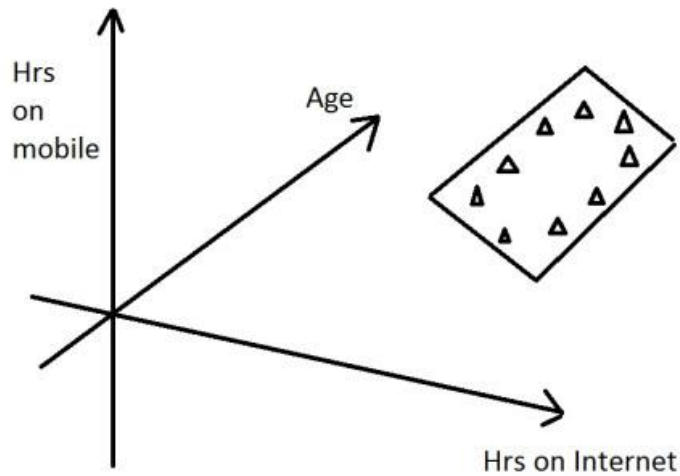


Figure 5-8: Three-Dimensional Representation of data points

Let us consider that we have 3 axes i.e., 3 eigen vectors. Out of these vectors, one of the eigenvectors value is zero. That vector can be neglected.

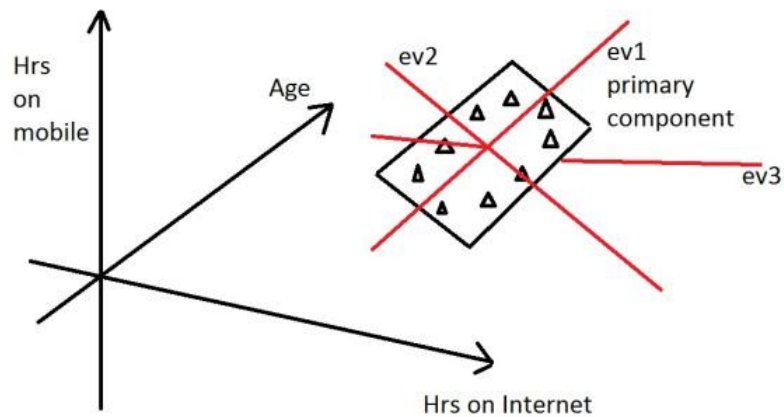


Figure 5-9: Eigen vector representation

As we have a vector with value zero, we can neglect that eigen vector. So, we can represent the data using 2 dimensions i.e., two vectors instead of 3 dimensions.

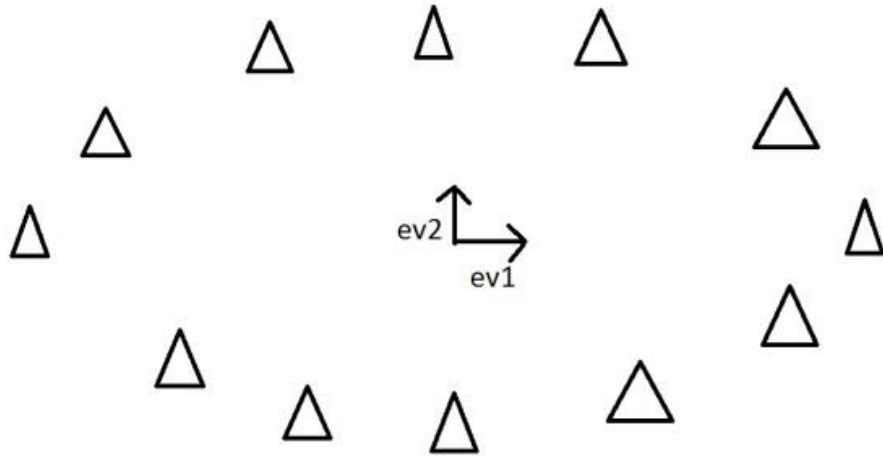


Figure 5-10: Representation of Dimension Reduction

Reducing dimensions helps to simplify the data and makes it easier to visualize. If in case one of the eigen vectors is not zero, we can neglect the vector with small value.

CHAPTER 6

FISHER VECTOR AGGREGATION

Fisher Vector aggregates against a GMM model. Series of partial derivatives are performed to find the direction in which the model must be changed to fit the datapoints in a better way.

The means and variances from the GMM are:

$$u_{jk} = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^N q_{ik} \frac{x_{ji} - \mu_{jk}}{\sigma_{jk}},$$
$$v_{jk} = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^N q_{ik} \left[\left(\frac{x_{ji} - \mu_{jk}}{\sigma_{jk}} \right)^2 - 1 \right]$$

6.1 Fisher Vector

The images in the dataset can be represented by using Fisher Vector (FV). It came into existence after BOV and VLAD. It has all the features of BOV and VLAD but apart from them it has some unique features which made this representation popular in these days. To model vocabulary PDF is used by which a gradient is calculated. GMM is used in this work. Fisher has better performance records than the techniques.

Categorization is involved with classification in which labels are assigned to images based on its content. Image Classification is very challenging task because of the scene variations, different viewpoints, lighting and occlusion. Hence, image classification remained as an open problem. The most popular technique to classify an image is by describing the image by using bag-of-words and then classifying them by using support vector, which is non-linear. Several modifications were done to the original approach for more efficiency. The most common trend is to use a combination of a set of different patch detectors, spatial pyramids and local descriptors, then to train generally non-linear classifiers on the corresponding high-level descriptors and finally, to combine the output of the classifiers. If there exists a huge number of images, we need scalable system to classify the data.

To reduce this disadvantage, we need to propose an algorithm which reduces the cost of training. Additive kernels are used to approximate the training set. These algorithms have same efficiency as SVM classifiers which are non-linear. There are many classifiers to classify the data like bag-of -words, VLAD encoding and fisher vectors. For BOW, we need large dataset, so VLAD and fisher vectors are widely used. But for probabilistic classification we are using fisher vectors which is a linear classifier.

Firstly, we describe the Fisher Kernel followed by the adaption of the Fisher Kernel to image classification. Then we relate fisher vector to several patch-encoding techniques.

6.2 Fisher Kernel

Let $X = \{x_t, t = 1 \dots T\}$ be a sample kernel for T observations $x_t \in \chi$. Let μ_λ be a probability density function which models the generative process of elements in χ where $\lambda = \{\lambda_1, \dots, \lambda_M\} \in \mathbb{R}^M$ denotes the vector of M parameters of μ_λ . In statistics, the score function is given by the gradient of the log-likelihood of the data on the model:

$$G_\lambda^X = \nabla_\lambda \log_{\mu_\lambda}(X)$$

The information of fisher vector is represented as

$$F_\lambda = E_{x \sim \mu_\lambda} [G_\lambda^X G_\lambda^{X^t}]$$

Based on these two observations we can define Fisher Kernel as

$$K_{FK}(X, Y) = G_\lambda^{X^t} F_\lambda^{-1} G_\lambda^Y$$

By using Cholesky decomposition Fisher Kernel can be written as

$$K_{FK}(X, Y) = \mathfrak{Z}_\lambda^{X^t} \mathfrak{Z}_\lambda^Y$$

Where $\mathfrak{Z}_\lambda^X = L_\lambda G_\lambda^X = L_\lambda \nabla_\lambda \log_{\mu_\lambda}(X)$

This is the normalized gradient vector which is called as Fisher Vector (FV). The dimension of fisher vector is \mathfrak{Z}_λ^X whereas the dimension of gradient vector is represented by G_λ^X . Both

the dimensionalities are equal. A non-linear kernel machine using K_{FK} as a kernel is equal to linear kernel machine using \mathfrak{F}_λ^X as its feature vector. This model is applied to the features of the images which are extracted from the scale invariant feature transform.

CHAPTER 7

DE-DUPLICATION

7.1 Introduction

The content which appears in more than a place on the internet can be treated as duplicate content. Here “a place” refers to the location which has a unique URL. In general, if more than one copy of the same content appears in one location on the internet it is referred as duplicate content. The content may be of different types like images, videos. There are many reasons for this rapid increase of the duplicate content like rapid growth in devices for capturing videos, software’s for editing the videos. Along with that the rapid growth in devices, gadgets leading to the increase in video data. This is increasing the data in public repositories and clouds. In many of these repositories, multiple copies of the same video are existing. This existence of multiple copies of same video is duplication. This is sometimes known as redundancy.

To remove this duplicate content or redundant content de-duplication techniques came into existence. They succeeded in removing duplicate data which is of same video content and which has same name, audio language and resolutions. But, they failed to remove the duplicate data which has different names, languages. For de-duplication with all these factors, we need an effective duplication technique which must have an effective area under curve. There are many techniques to remove the redundant data. The following figure shows the classification of efficient video storage techniques without redundancy. The result of our study must be applicable to both content-aware and content-unaware cases.

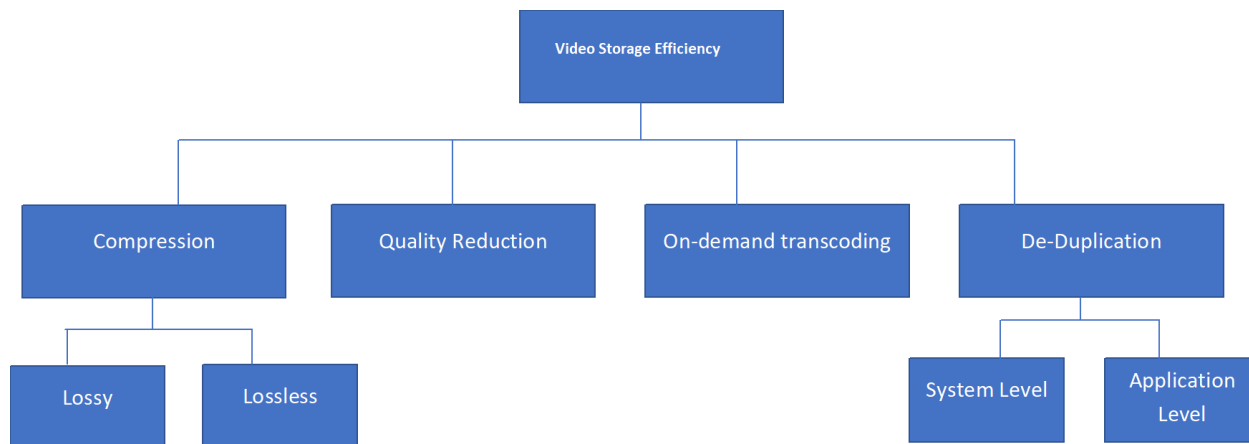


Figure 7-1: Efficient Video Storage Techniques

Compression is a technique which compresses the large video data into small video data. For video compression, it uses some coding algorithms. By applying discrete cosine and discrete wavelet transform it removes the redundancy. Temporal motion identifies changes in frames and it stores the videos in a specific format of reference frame. Lossy compression and lossless compression are the two types of techniques in compression. We try to do lossless compressions but every technique is somewhat lossy.

Quality reduction is the technique which reduces the size of file by diminishing the resolution or by limiting the frames. This makes the content worse than original which is still acceptable in less applications. Converting from an encoding to another is transcoding. It encodes according to the requirement of the user. Instead of storing the same video in different resolutions, Netflix encodes the video in any format as required by the user.

Deduplication is a technique which eliminates the redundancy in any file which exist repository. This can be done in either application level or system level. If we are aware of the content, we can do the de-duplication at application level. If it is content unaware, we can use system level deduplication.

Deduplication is also known as neighbor search. The process of finding the neighbor which is close to it is known as similarity search. For deduplication, it costs more to maintain large database and to find the distance between query images and the content in database. In traditional techniques, there occurs a lot of problems practically. Out of many solutions, hashing is one of best approach which involves in transforming the content into a representation which is low-dimensional.

7.2 Hashing

Low-dimensional representation is the short code representation which is a sequence of bits. It contains only binary numbers 0's and 1's. By using hashing, we can perform the deduplication in two approaches. Indexing the data items and matching it with the code in hash buckets and by using the distance which is computed by short codes calculating the distance. Hashing is used for similarity search which is fast process. By using this hashing, we have many hashing algorithms. Based on the generation of functions of hash we can categorize the methods.

In this process, we are using the global fisher aggregated vectors to generate a hash function. The distance between the frames is selected out of which some frames are selected to generate a sequence. In sequence level, we are going to perform similarity check.

CHAPTER 8

SIMULATION RESULTS AND DISCUSSIONS

The dataset used in this process is a 5min 33sec video. This video is converted into images by using FFMPEG. By using FFMPEG the video is converted into .mp4 file and then it converts .mp4 file into set of .png files as per our requirement. Here I considered 10,000 images as input. The steps involved in downloading, installing and functionality of FFMPEG is shown as below.

FFMPEG Installation

Download FFMPEG from <https://ffmpeg.zeranoe.com/builds/>. It is recommended to download 64-bit Downloads (Download FFMPEG git-30d1213 64-bit Static). Based on the platform we download the required version of FFMPEG.

After downloading the FFMPEG we have to install the FFMPEG. To download the FFMPEG we have to make some changes in the properties of our computer. We have to add a new variable in the environment variables and the address of bin folder in FFMPEG is added as path to that new variable. The steps involved in the installation of FFMPEG are as shown below.

PC → Properties → Advanced System Settings → Environment Variables → new → variable name: Path → variable value:
address(C:\users\balac\Documents\MATLAB\thesis\ffmpeg\bin) → ok → ok → ok

Figure 8-1: Process of Installation

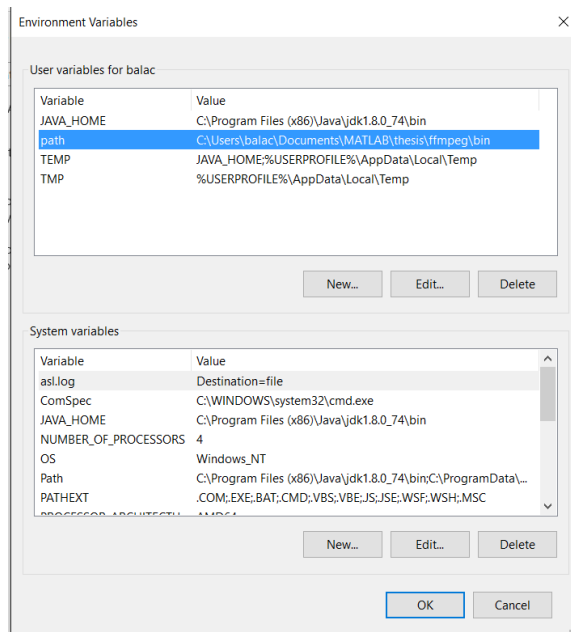


Figure 8-2: Installation of FFMPEG

After installing FFMPEG as shown in above figure, we can check the installation of ffmpeg by using the command: **ffmpeg -version**. After using that command in command window we will get the result as follows.

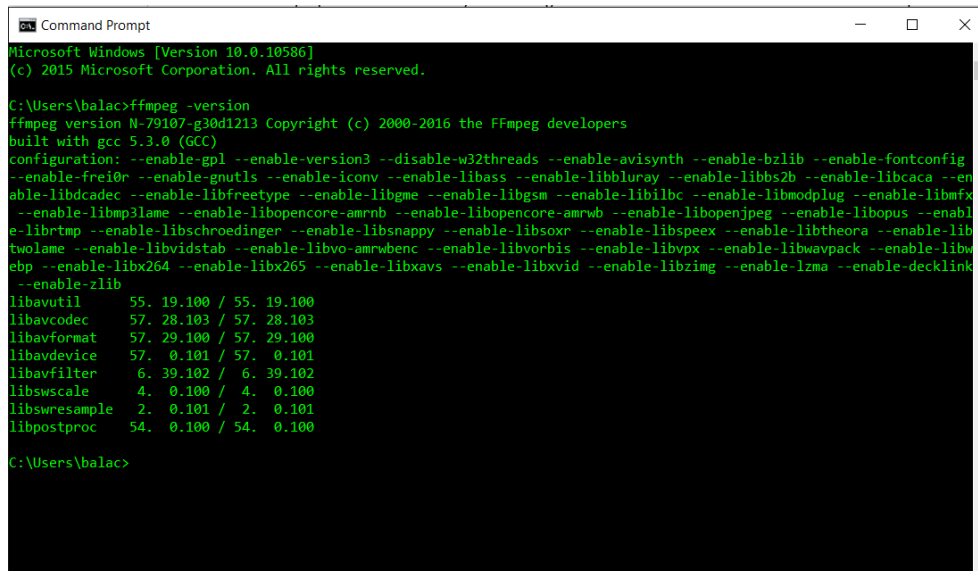


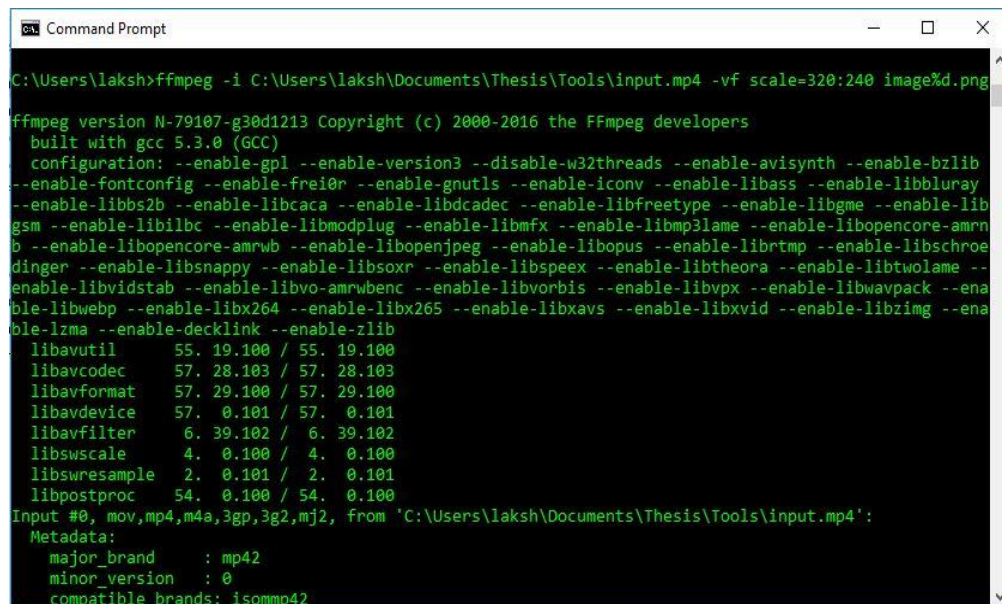
Figure 8-3: Checking the installation of FFMPEG

The input video must be converted into set of images. In this paper we are going to convert the input video into images with two resolutions. We are going to convert the video into images with resolution of 320×240 and 640×360.

The .yuv file cannot be directly decomposed into images. So, .yuv file can be converted into .mp4 file and then it is converted into .png files by using FFMPEG. The video which is in .yuv format is converted into .mp4 file by using the following command. In the command *i* is the input, *-vf* is textual representations and *scale* is the aspect ratio of the output.
Command: `ffmpeg -i C:\Users\laksh\Documents\Thesis\Tools\input.yuv -vf scale=320:240 input.mp4.`

This command results in .mp4 file named input-320kbps. Then we have to convert this .mp4 file to .png files. The video which is in .mp4 format is converted into .png files by using the following command.

Command: `ffmpeg -i input-320kbps.mp4 image%d.png`



```
Command Prompt
C:\Users\laksh>ffmpeg -i C:\Users\laksh\Documents\Thesis\Tools\input.mp4 -vf scale=320:240 image%d.png

ffmpeg version N-79107-g30d1213 Copyright (c) 2000-2016 the FFmpeg developers
  built with gcc 5.3.0 (GCC)
  configuration: --enable-gpl --enable-version3 --disable-w32threads --enable-avisynth --enable-bzlib
--enable-fontconfig --enable-frei0r --enable-gnutls --enable-iconv --enable-libass --enable-libbluray
--enable-libs2b --enable-libcaca --enable-libcdcodec --enable-libfreetype --enable-libgme --enable-lib
gsm --enable-libilbc --enable-libmodplug --enable-libmfx --enable-libmp3lame --enable-libopencore-amrnb
b --enable-libopencore-amrwb --enable-libopenjpeg --enable-libopus --enable-librtmp --enable-libschr
dinger --enable-libsnappp --enable-libsoxr --enable-libspeex --enable-libtheora --enable-libtwolame --
enable-libvidstab --enable-libvo-amrwbenc --enable-libvorbis --enable-libvpx --enable-libwavpack --ena
ble-libwebp --enable-libx264 --enable-libx265 --enable-libxavs --enable-libxvid --enable-libzimg --ena
ble-lzma --enable-decklink --enable-zlib
  libavutil      55. 19.100 / 55. 19.100
  libavcodec     57. 28.103 / 57. 28.103
  libavformat    57. 29.100 / 57. 29.100
  libavdevice    57.  0.101 / 57.  0.101
  libavfilter     6. 39.102 /  6. 39.102
  libswscale      4.  0.100 /  4.  0.100
  libswresample  2.  0.101 /  2.  0.101
  libpostproc   54.  0.100 / 54.  0.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'C:\Users\laksh\Documents\Thesis\Tools\input.mp4':
Metadata:
  major_brand      : mp42
  minor_version    :  0
  compatible_brands: isommp42
```

Figure 8-4: Conversion of video into images with resolution 320×240

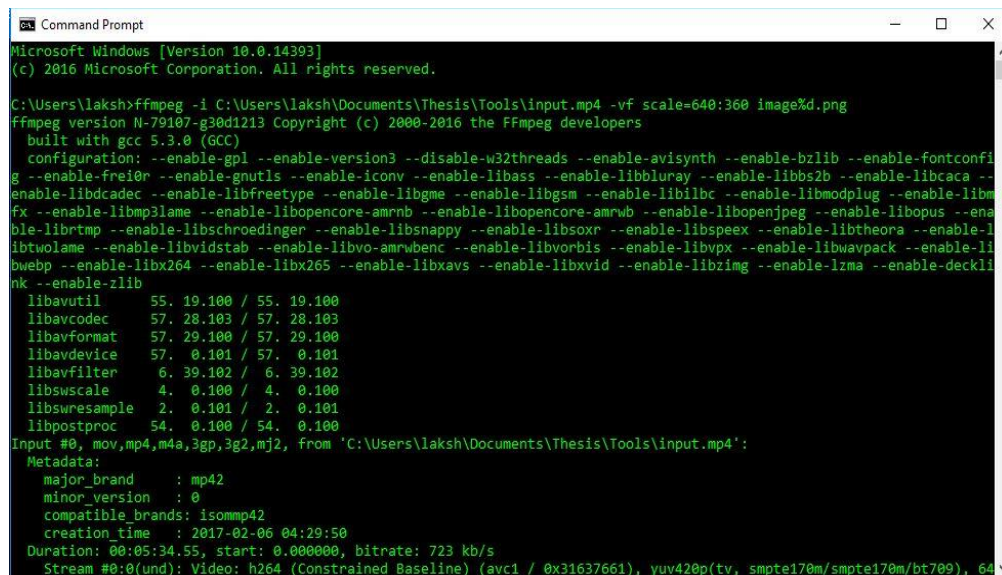
This operation results in 2.5k images of resolution 320×240.

The .yuv file cannot be directly decomposed into images. So, .yuv file can be converted into .mp4 file and then it is converted into .png files by using FFMPEG. The video which is in .yuv format is converted into .mp4 file by using the following command. In the command *i* is the input, *-vf* is textual representations and *scale* is the aspect ratio of the output.

Command: `ffmpeg -i C:\Users\laksh\Documents\Thesis\Tools\input.yuv -vf scale=640:360 input.mp4`.

This command results in .mp4 file named input-640kbps. Then we have to convert this .mp4 file to .png files. The video which is in .mp4 format is converted into .png files by using the following command.

Command: `ffmpeg -i input-640kbps.mp4 image%d.png`



```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\laksh>ffmpeg -i C:\Users\laksh\Documents\Thesis\Tools\input.mp4 -vf scale=640:360 image%d.png
ffmpeg version N-79107-g30d1213 Copyright (c) 2000-2016 the FFmpeg developers
  built with gcc 5.3.0 (GCC)
  configuration: --enable-gpl --enable-version3 --disable-w32threads --enable-avisynth --enable-bzlib --enable-fontconfi
g --enable-frei0r --enable-gnutls --enable-iconv --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca --
enable-libcdcodec --enable-libfreetype --enable-libgme --enable-libgsm --enable-libilbc --enable-libmodplug --enable-libm
fx --enable-libmp3lame --enable-libopencore-amrnb --enable-libopencore-amrwb --enable-libopenjpeg --enable-libopus --ena
ble-librtmp --enable-libschrödinger --enable-libsnappp --enable-libsoxr --enable-libspeex --enable-libtheora --enable-l
ibtwolame --enable-libvidstab --enable-libvo-amrwbenc --enable-libvorbis --enable-libvpx --enable-libwavpack --enable-li
bwebp --enable-libx264 --enable-libx265 --enable-libxavs --enable-libxvid --enable-libzimg --enable-lzma --enable-deckli
nk --enable-zlib
  libavutil      55. 19.100 / 55. 19.100
  libavcodec     57. 28.103 / 57. 28.103
  libavformat    57. 29.100 / 57. 29.100
  libavdevice    57.  0.101 / 57.  0.101
  libavfilter    6. 39.102 / 6. 39.102
  libswscale     4.  0.100 / 4.  0.100
  libsresample   2.  0.101 / 2.  0.101
  libpostproc   54.  0.100 / 54.  0.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'C:\Users\laksh\Documents\Thesis\Tools\input.mp4':
Metadata:
  major_brand      : mp42
  minor_version    : 0
  compatible_brands: isommp42
  creation_time    : 2017-02-06 04:29:50
Duration: 00:05:34.55, start: 0.000000, bitrate: 723 kb/s
Stream #0:0(und): Video: h264 (Constrained Baseline) (avc1 / 0x31637661), yuv420p(tv, smpte170m/smpte170m/bt709), 64
```

Figure 8-5: Conversion of video into images with resolution 640×360

A set of 2.5k images are generated and those images are stored in a folder. As we have large dataset, it is difficult and takes lot of time to process every image in the dataset. So, we have to limit the dataset by using Sampling. Sampling is the change of data by transforming continuous into discrete or discrete into continuous. There are two types of samplings

upsampling and downsampling. Downsampling is decreasing the sample rate whereas upsampling is increasing the sample rate. In this paper, we have to reduce the dataset so we are using downsampling in this paper. In this paper, I have a dataset of 2.5k images. So, I used downsampling by the rate of 5.

```
no_img=2500;  
x=1:no_img;  
Indx=downsample(x, 5);
```

After downsampling, the resulting dataset contains 500 images. After downsampling we have to take an empty cell of 500×1 by using the command.

```
my_celll=cell(size(indx,2),1)
```

After taking the empty single dimensional array, we have to insert the dataset into that one-dimensional array. Then we have to perform all operations by including vl_feat. Vl_feat contains the demo codes for sift, fisher, vlad and so many other operations. cdvs_sift_aggregation_test data is loaded for sift_cdvs and sift_offs files. Then we have to load the trained data of pca and test data of sift. The dataset is to be scaled to 480 pixels in order to perform fisher aggregations. For fisher vectors the image is tuned to gray scale image for better classification. Fisher vectors are found by using gmm which contains means, covariance and priors. These are found by using number of clusters and data.

The size of fisher vectors is 500×1536 . As we got more fisher vectors in which some are repetitive and unnecessary. So, we must find my significant fisher vectors. This is done by using a function called getFVSamples(). This is used to compute the frame significance value of each frame and a threshold is set up. If the frame significance value is greater than threshold, then it is taken into count. The frames which are greater than threshold are the

significant frames. The plots of frame significance and cumulative frame significance values are as follows:

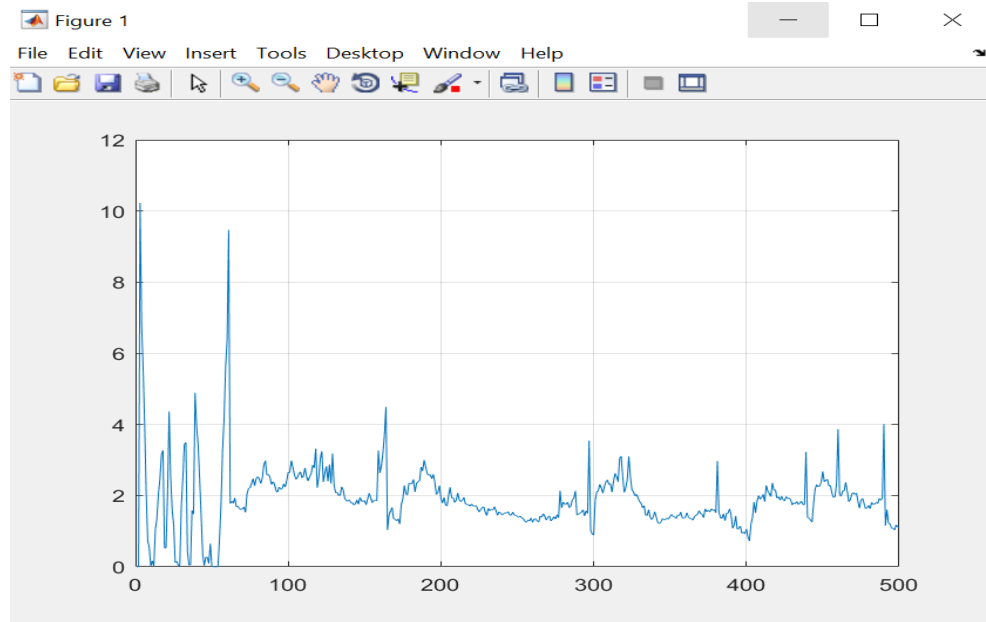


Figure 8-6: Plot of Frame Significance

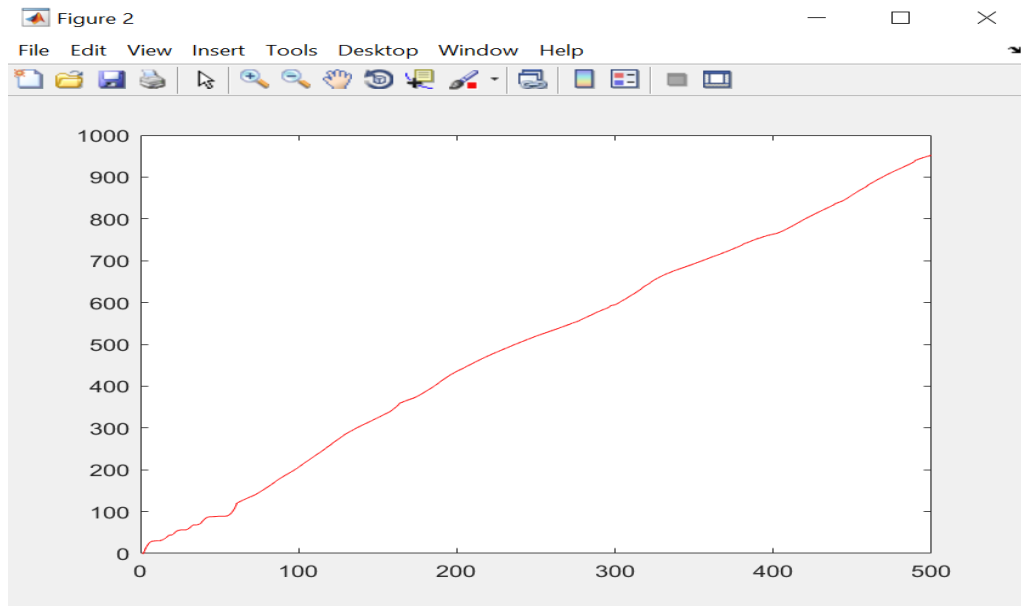


Figure 8-7: Plot of Cumulative frame significance

But these significant frames consist of some repetitive frames. So, fisher aggregation is to be performed out of which the result is 20 frames and it is named as fv. The performance of the entire technique is evaluated by using ROC plots by using True positive rate and false positive rate.

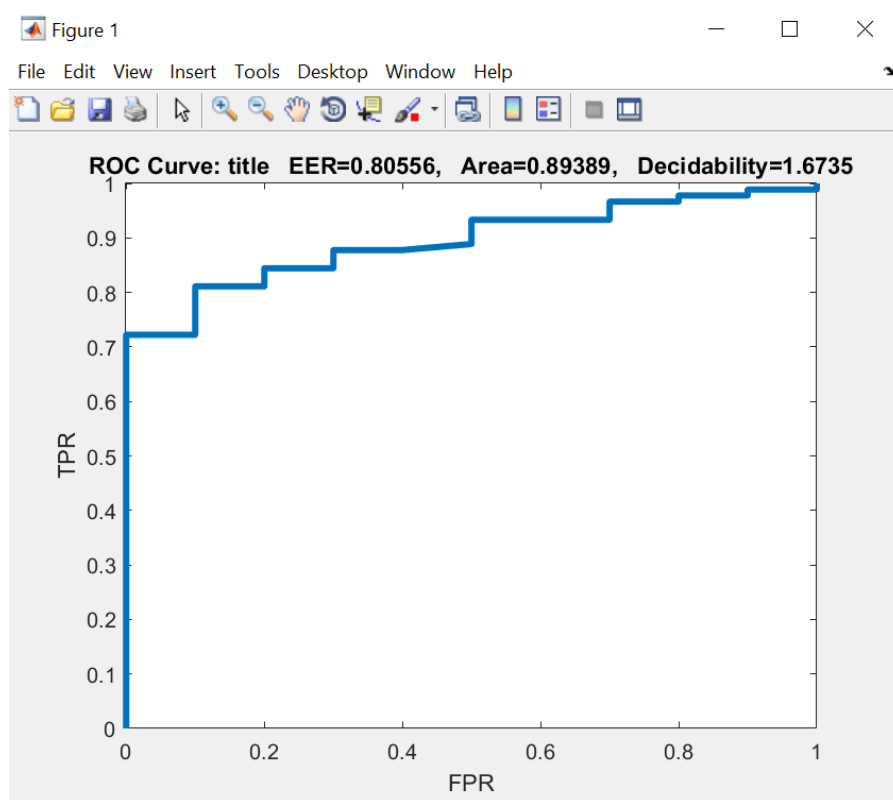


Figure 8-7: ROC curve for fisher aggregation of dataset with resolution 640×360

This is the ROC curve for 640×360 resolution. This is plotted against the True Positive Rate and False Positive Rates. By applying a threshold value to the classifier True positive and false positive rates can be obtained by which Receiver Operating Characteristics can be plotted. The AUC (Area under curve) for this plot is 0.89 which means that the classifier used in this paper is best classifier.

After aggregating the fisher vectors, we are going to convert these fisher vectors into hash codes which consists of 1's and 0's. By using hash frame distance, we are going to generate a sequence. Now, we are conducting the similarity check for deduplication. For deduplication, we are training some videos in database then checking the existing video with the videos in database.

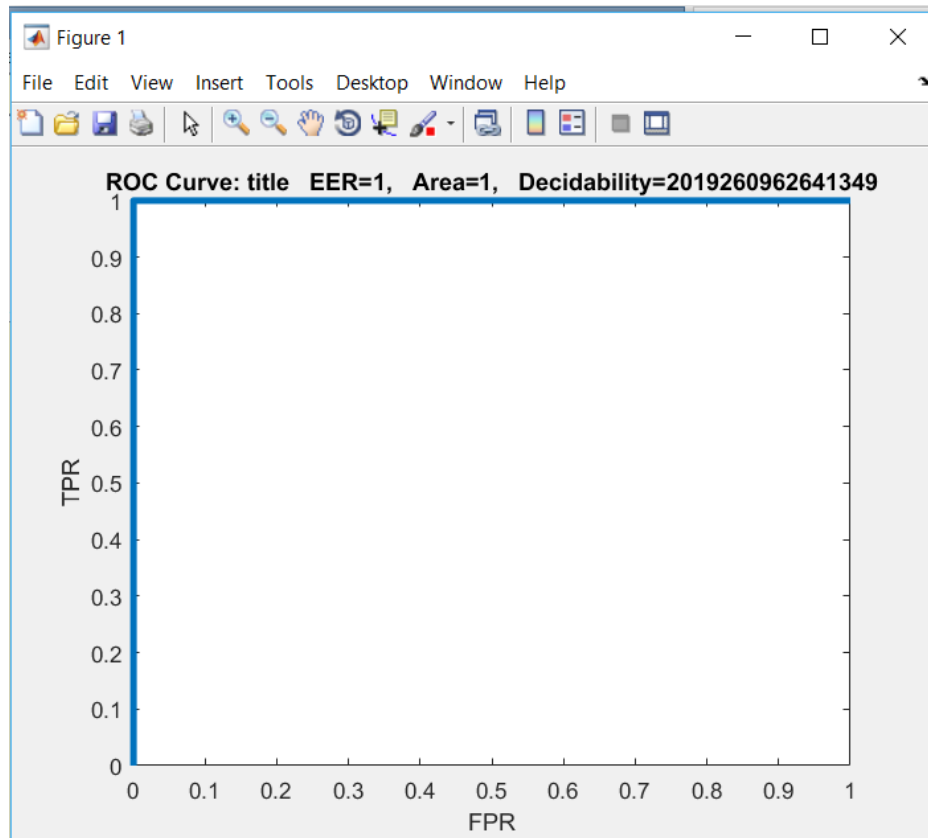


Figure 8-8: ROC curve for sequence level

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 Conclusion

In this work, the images with different results are taken as dataset for which we got the same results. As we have 2.5k images in each dataset, it is difficult to classify every image. So, down sampling is done by rate 5 which results in 500 images. The features of the resultant images are extracted by using SIFT. Before classifying the data, the most significant frames are considered by calculating the frame significance value. Then the resultant frames are perfectly classified by using Fisher Vector which gives much better results than VLAD classification. As the above operation results in more fisher vectors, we have to aggregate those fisher vectors. We are aggregating 500 frames into 20 frames by using fisher aggregation.

The performance of the fisher aggregation can be determined by using ROC curves by using True Positive Rate and False Positive Rate. This is plotted by True positive rate on Y-axis and False positive rate on X-axis. The area under the curve for 640×360 is 0.893. After fisher aggregation, we developed a hashing technique for sequence level hashing for which we got the ideal ROC curve. From this, I can conclude that the better deduplication can be performed at sequence level rather than frame level.

9.2 Future Work

Further this work can be extended by maintaining a data base which has the sequences of the datasets. This database can be used as reference and the future videos can be checked with the videos in database for similarity.

APPENDIX

```
function [sift fishsif]=getVideoSiftFV(mp4_fname, no_img,indx, w,code,gmm)
no_img=2500;
x=1:no_img;
indx=downsample(x,5);
code='1.2';
w='360';
mp4_fname='images';
sift=cell(size(indx,2),1);
gmm=[16 32];
kd=gmm(1);
nc=gmm(2);
path='C:\Users\Nandeeep\Documents\Prasanna\images' ;
c=1;
my_cell=cell(500,1);
for i = 1 :5: no_img
    str = strcat(path, '\image',int2str(i),'.png');
    img = imread(str);
    my_cell{c,1}=img;
    c=c+1;
end
if (1)
    run('C:\Users\Nandeeep\Documents\Prasanna\Thesis\vlfeat-
0.9.20\toolbox\vl_setup.m');
end
my_cell1=cell(500,1);
numrows=280;
numcols=340;
c=1;
for i=1:5:no_img
    my_cell1{c,1} = imresize(my_cell{c,1},[numrows numcols]);
    c=c+1;
end

load('C:\Users\Nandeeep\Documents\Prasanna\Thesis\cos_pca.mat');
load('C:\Users\Nandeeep\Documents\Prasanna\Thesis\sift_test.mat');
gmm=[24 32];
kd=gmm(1);
nc=gmm(2);
fishsif=[];
c=1;

peak_thresh=3;
[m1, cov1,pr1] = getgmm1(kd,nc);
for i=1:5:no_img
    I=single(rgb2gray(my_cell{c,1}));
    [f d] = vl_sift(I, 'PeakThresh', peak_thresh);
    d=d';
    c=c+1;
    sifpr=double(d)*Cos(:,1:kd);
    x=vl_fisher(sifpr',m1,cov1,pr1);
    fishsif=cat(1, fishsif, single(reshape(x,1,numel(x))));
end
m=10;
```

```

[s]=getFVSSamples(fishsisf, m);
c=2;
s1(1)=s(1);
for i=2:size(s,2)
    if s(i)>s(i-1)
        s1(c)=s(i);
        c=c+1;
    end
end
[n d]=size(fishsisf);
c=1;
for i=1:500
    if s1(c)==i
        fv(c,:)=fishsisf(i,:);
        c=c+1;
    end
end
[kd,nc]=size(fv);
H=zeros(kd,nc);
H1=zeros(kd,nc);
H(find(fv>0))=1;
[l]=getFrameHashDistance(H,m);
D=pdist2(H,H1,'hamming');
h_dist = sum(D(: ));
end

```

FVSSamples Function:

```

function [s]=getFVSSamples(fishsisf, m)
dbg=1;
n_frame=500;
load('C:\Users\Nandeep\Documents\Prasanna\fishsisf.mat');
x=fishsisf;
[n, kd]=size(x);
fs = zeros(1,n); fs(2:end) = diag(pdist2(x(1:end-1,:), x(2:end, :)));
m=20;
cfs = cumsum(fs);
step_size = sum(fs)/(m-1);
s(1) = 1;
for k=1:m-1
    offs = find(cfs>=step_size*k);
    s(k+1) = offs(1);
end
figure;
plot(fs);hold on; grid on;
figure;
plot(cfs, '-r');
return;

```

ROC Plots Function:

```

clear;
clc;
ind=[14 9 14 19 21 16 16 19 15 12 10 13 19 12 11 11 15 15 17 13];
cum=cumsum(ind);
T=zeros(54,54);
c=1;

```



```

flag=1;
for i=1:54
    for j=flag:cum(c)

        T(i,j)=1;
    end
    if i==cum(c) flag=flag+ind(c);
        c=c+1;
    end
end
load('C:\Users\Nandeep\Documents\Prasanna\fv.mat');
b=zeros(10,10);
for i=1:10
    for j=1:10
        dist=getSegSiftFVDistance(fv(i,:),fv(j,:));
        if i==j
            dist_final=min(dist);
        else
            dist_final=dist(2);
        end
        b(i,j)=dist_final;
    end
end
TT=zeros(10,10);
for i=1:10
    for j=1:10
        if i==j TT(i,j)=1;
        end
    end
end
ezroc3(b,TT,2,'title',1);

```

BIBLIOGRAPHY

[1] K.L. Lim and H.Wang, "Sparse coding based Fisher vector using a Bayesian approach," *Journal of IEEE Signal Processing*, vol 83, pp. 91-95, Jan 2017.

[2] Y. Song, Q. li, H. huang, D. Feng, M. Chen and W. Cai, "Low Dimensional Representation of Fisher Vectors for Microscopy Image Classification," in *IEEE Transactions on Medical Imaging*, vol.PP, no.99, pp.1-1

doi: 10.1109/TMI.2017.2687466

[3] I. Ilea, L.Bombrum, C. Germain, R. Terebes, M.Borda and Y. Berthoumieu, "Texture image classification with Riemannian fisher vectors," *2016 IEEE International Conference on Image Processing(ICIP)*, Pheonix, AZ, 2016, pp 3543-3547.

doi: 10.1109/ICIP.2016.7533019

[4] R.D.C.Silva, G.A.P.The, FN.S.de Medeiros, "Geometrical and statistical feature extraction of images for rotation invariant classification systems based on industrial devices," *2015 21st International Conference on Automation and Computing(ICAC)* , Glasgow, 2015, pp. 1-6.

doi:10.1109/IConAC.2015.7313946

[5] S.A.A Bukhari and S. Iqbal, " A Hardware Architecture for Difference of Gaussian Calculation in Image Feature Extraction," *2014 12th International conference on Frontiers of Information Technology*, Islamabad, 2014, pp.342-345.

doi: 10.1109/FIT.2014.70

[6] M.Makar, C.L. Chang, D. Chen, S.S. Tsai and B. Girod, "Compression of image patches for local feature extraction," *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, 2009, pp. 821-824.

doi: 10.1109/ICASSP.2009.4959710

[7] J. You, E.Pissaloux, J.L Hellec and P. Bonnin, “A guided image matching approach using Hausdorff distance with interesting points detection,” *Proceedings of 1st International Conference on image processing*, Austin, TX,1994, pp. 968-972 vol.1.

doi: 10.1109/ICIP.1994.413253

[8] G. Carneiro and A.D. Jepson, “Multi-scale phase-based local features,” *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, 2003, pp. I-736-I-743 vol.1.

doi: 10.1109/CVPR.2003.1211426

[9] T. Deselaers, D.Keysers and H.Ney, “Discriminative training for object recognition using image patches,” *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, 2005, pp.157-162 vol.2.

doi: 10.1109/CVPR.2005.134

[10] D. Parikh and C.L. Zitnick, “The role of features, algorithms and data in visual recognition,” *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Fransisco, CA, 2010, pp.2328-2335.

doi: 10.1109/CVPR.2010.5539920

[11] T. Tuytelaars, “Dense interest points,” *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, 2010, pp.2281-2288.

doi: 10.1109/CVPR.2010.5539911

[12] Y.L.Boureau, F.Bach, Y.LeCun and J.Ponce, “Learning mid-level features for recognition,” *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, 2010, pp. 2559-2566.

doi: 10.1109/CVPR.2010.5539963

[13] M. H. Ferris, M. McLaughlin, S. Grieggs, S. Ezekiel, E. Blasch, M. Alford, M. Cornacchia and A. Bubalo, "Using ROC curves and AUC to evaluate performance of no-reference image fusion metrics," *2015 National Aerospace and Electronics Conference (NAECON)*, Dayton, OH, 2015, pp.27-34.

doi: 10.1109/NAECON.2015.7443034

VITA

Lakshmi Prasanna Gadiparthi received her degree of Bachelors (B.Tech) from Acharya Nagarjuna University in the stream of Electronics and Communication Engineering in May 2014. In Bachelors, she worked on various projects in Image Processing Techniques using MATLAB. She worked on Automatic License Plate Recognition System (ALPR) using MATLAB and the image recognition techniques using FPGA (Field Programmable Gate Array). In January 2016, she came to the United States of America to pursue her Master of Science degree at University of Missouri-Kansas City (UMKC) in the stream of Electrical Engineering, specializing in Wireless Communications. During her time in UMKC, she is familiar with different MATLAB centric courses like Multimedia Communications under Dr. Zhu Li. After that she is working on thesis under Dr. Zhu Li. To get more technical exposure, currently she is working as System Analyst.