B.TECH PROJECT REPORT

on

Video Summarization for anomaly detection using Deep Learning

By

Jagruthi Patibandla, 180001021
Shravya Ramasahayam, 180001052

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING,
INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2022

Video Summarization for anomaly detection using Deep Learning

A PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Jagruthi Patibandla, 180001021
Shravya Ramasahayam, 180001052

Discipline of Computer Science and Engineering,
Indian Institute of Technology Indore

Guided by:

Dr. Aruna Tiwari,

Professor,

Computer Science and Engineering,
IIT Indore

INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2022

Candidate's Declaration

We hereby declare that the project entitled **"Video Summarization for anomaly detection using Deep Learning"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering completed under the supervision of **Dr. Aruna Tiwari, Professor, Computer Science and Engineering,** IIT Indore is an authentic work.

Further, we declare that we have not submitted this work for the award of any other degree elsewhere.

Jagruthi Patibandla
Shravya Ramasahayam

Certificate by BTP Guide

It is certified that the above statement made by the student is correct to the best of my/our knowledge.

Dr. Aruna Tiwari
Professor
Discipline of Computer Science and Engineering
IIT Indore

Preface

This report on **“Video Summarization for anomaly detection using Deep Learning"** is prepared under the guidance of Dr. Aruna Tiwari, Professor, Computer Science and Engineering, IIT Indore.

Through this report, We have tried to provide a detailed description of our approach, design, and implementation of Video summarization using dynamic image generation. We also tried to perform key frame selection using Fuzzy C-Means Clustering and the Deep Neuro-Fuzzy model. We explained the proposed solution to the best of our abilities.

Jagruthi Patibandla
Shravya Ramasahayam
B.Tech. IV Year
Discipline of Computer Science Engineering
IIT Indore

Acknowledgments

We want to thank our B.Tech Project supervisor, Dr. Aruna Tiwari, for her guidance and constant support in structuring the project and for providing valuable feedback throughout this project. Furthermore for giving us an opportunity to deal with advanced technologies. We would also like to thank all the faculty members of the Discipline of Computer Science and Engineering for their invaluable support during the presentations. We are grateful to the Institute for providing the necessary tools and utilities to complete the project. We wish to express our sincere gratitude to Mr. Rituraj for his guidance throughout the project and for helping us at every stage. Lastly, we offer our sincere thanks to everyone who helped us complete this project, whose name we might have forgotten to mention.

Jagruthi Patibandla
Shravya Ramasahayam
B.Tech. IV Year
Discipline of Computer Science Engineering
IIT Indore

Abstract

Video summarization creates a concise summary of the content of a lengthier video document, by selecting and presenting the most helpful or intriguing components. It has wide range of applications in various domains, like surveillance, entertainment, education, advertisement and more. In the current work, we propose video summarization of anomalous content in surveillance videos. Crime has become a concerning problem in today's evolving world. Close scrutiny of our daily visiting places, be it house, office, malls, etc., has become a necessity for us with crimes increasing every day. This inspection is being achieved by installing surveillance cameras everywhere. The videos recorded this way comes in handy for crime investigations. But the problem is to go through the complete recorded video to find if it has some anomalous content, which is both time and effort-consuming. This brings in a requirement for some mechanism that can save our resources without compromising the information.

A similar approach is to summarize the anomalous content of a video in terms of a single image. To be more precise, we need a 2-step pipeline, i.e., a system which can precisely identify the abnormal content in a video and then combine this extracted content into a single image. In this project, we try to build a deep learning model which can identify anomalous frames and a rank pooling algorithm that can combine these anomalous frames into a single dynamic image.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recording surveillance cameras is of prime importance today to ensure public safety and speed up police investigations. These recordings also serve as a shred of good material evidence for the police to solve criminal cases quickly. When there is information of some crime incident at a place, the police recover the related surveillance footage and inspect it to find any abnormal events. Usually, this requires us to go through the complete footage before taking necessary action. To speed up such processes, we can develop techniques to summarize a given video.

A video is composed of a series of still images known as frames. Video summary aids in the extraction of important information from a video. It comes in handy when we don't want to watch the entire video but want to get the most out of it. In the current work, we perform summarization by considering only the frames of our interest, i.e., frames including an abnormal activity. In the domain of surveillance, an anomalous event can be any criminal activity like abuse, assault, burglary, shoplifting, vandalism, etc. A summary of a video we are trying to achieve here is a single image called a dynamic image. We will briefly discuss the motivation behind video summarization in the following section 1.1.

1.1 Motivation

Video summaries have a wide range of uses in a variety of areas, including surveillance, entertainment, education, and advertising. Video summarization has the following advantages:

1. **Minimal Time and Human effort:** It is both time and effort taking for a person to go through a complete video to locate anomalous content. This is because the person inspecting the video may not know beforehand which

part of the video has an anomaly if an anomaly exists. This becomes a hectic task when there are many long videos to analyze.

2. **Reduced need for resources:** Video processing demands heavy computational resources. Also, sometimes, we might not want to perform heavy computation on the whole video. In such cases, video summarization gives the freedom to choose the content of our interest.

3. **Easy transmission:** Long videos require high bandwidth for transmission over devices. Summary of the video helps reduce bandwidth consumption.

So far, we have discussed the benefits of video summarization. In the upcoming section 1.2, we will go over the problem statement in detail.

1.2 Problem Statement

Video summarization is a field of growing research and interest. This work aims to create an architecture, which can identify the anomalous frames in a video and generate their summary, a single RGB image called a dynamic image. We further propose deep learning techniques to select the abnormal frames in the video and rank pooling algorithms to compute the dynamic image summary of a given video. Experimentation of the architecture was done on the UCF Crime dataset. We have discussed over the problem statement so far. The objectives will be discussed in the next section 1.3.

1.3 Objectives

a) *Preprocessing videos from dataset to be used as an input for designed models:* This involves generating frames from all the videos of the considered dataset.

b) *Extracting features from the preprocessed data:* The frames extracted are used to generate features using ResNet and VGG16 models pre-trained on the ImageNet dataset.

c) *Unsupervised Clustering using Fuzzy C-Means Algorithm:* The existing Fuzzy C-Means clustering algorithm is used to cluster the frames based on

the features generated from the pre-trained models. The number of clusters chosen is two, analogous to normal and anomalous clusters.

d) *Classification using Deep Neuro-Fuzzy Network:* The existing architectures of the Deep Neuro-Fuzzy Network are modified to perform classification on the frames of the videos. The model predicts if a given frame is anomalous or not.

e) *Performance analysis and comparison of Fuzzy C-Means Algorithm and Deep Neuro Fuzzy Network:* We will evaluate the performance of both methods using various performance metrics.

f) *Generation of Dynamic Image:* The output of the best of the two methods, after comparison, is given as input for creating dynamic images. We generate a dynamic image, which is an RGB image, to summarize the dynamics of the entire video.

In this chapter 1, we have discussed about problem statement and related things in detail. We will be describing about video summarization techniques, deep learning models and key aspects in the next chapter 2.

Chapter 2

Literature Review

In this chapter, we describe the work related to video summarization, Deep Neuro Fuzzy models for supervised learning, K-Means and Fuzzy C-Means clustering algorithms, principal component analysis for dimensionality reduction, and techniques to handle imbalanced data. In the upcoming section 2.1, we will discuss about Video Summarization in detail.

2.1 Video Summarization

Video Summarization aims to create a short summary of the original video. Understanding video content can be a time-consuming task, as videos in everyday life can be very long. A brief summary of the video makes it easy for viewers to identify the video content. Video summarization can be categorized into two types [14]: static and dynamic summarization. Static summarization includes key-frame selection, and dynamic summarization includes the creation of short video skims. Previous works used video tags to summarize the video and identify key-frames. The relevance of each frame is checked against the assigned video tag [21].

A rank pooling approach was proposed by Fernando et al. [8], where they learned to rank the video frames. The new representation captures the dynamics of the video. The extension of this work was proposed by Bilen et al. [6] to create a dynamic image summary of a video clip. Image pixels in a video frame already contain information that can be overloaded to accommodate the long-term dynamics of the video. Dynamic images are powerful, efficient, yet simple video representations generated by rank pooling of frames. A dynamic image is a typical RGB image summarizing the dynamics and content of the entire video sequence. Further, an approximate rank pooling operation was proposed to fasten the dynamic image computation.

2.1.1 Advantages of Video Summarization

The primary advantages of summarizing a video using the dynamic image are:

- It is a compact summary as a single RGB image can be generated to summarize the entire video dynamics.
- A dynamic image can be fed as an input to existing CNN models used for still images with fine-tuning.
- This reduces the video classification task to that of an image classification task and is significantly efficient for video analysis.

In our problem, in order to summarize the anomalous content of a video, we first need to identify the anomalous frames amongst all the video frames. In the upcoming sections, we will discuss clustering and classification models to identify the anomalous frames. We start by defining deep neural networks in the section 2.2.

2.2 Deep Neural Network

Deep learning is a sub-paradigm under the domain of machine learning. It has a good ability to learn and improve on its own by analyzing computer algorithms without human intervention. Deep learning has proved to be efficient at pattern recognition in our data and make predictions accordingly. It helps in deriving generalizations in our data by making inferences. Deep learning systems are feed-forward artificial neural networks with many hidden processing layers through a system of weighted connections between input and output layers. These networks have the capability of learning and modeling hidden nonlinear and complex relationships between inputs and outputs.

Warren McCulloch and Walter Pitts et al. [19] designed the first neural network. They developed a simple neural network making use of electrical circuits, modeling the working of neurons. Later, Kunihiko Fukushima et

al. [9] developed the first true, multilayered neural network. The neural network approach came from the idea of imitating the human brain to solve real problems. Later on, neural networks deviated from a strictly biological approach and were developed to support diverse tasks like speech recognition, computer vision, machine translation, etc. With the advancements in Big Data analytics and High-performance GPUs that have a parallel architecture, sophisticated neural networks also can be trained on large amounts of data. The number of layers in the deep learning network defines how deeper the network is. Deep networks can hold an enormous number of hidden layers(as many as 150), whereas traditional neural networks usually contain 2-3 hidden layers.

Nodes in neural networks are analogous to neurons in the human brain. Artificial neural network functions by assigning corresponding weights to connections when signals travel between nodes. Similar to neurons in terms of behavior, nodes in ANN get activated when there are stimuli or input. The connections between these artificial neurons function as simple synapses, enabling information signals to be transmitted from one to another. When exposed to an input, the neurons carry out mathematical calculations to decide on the information to be transmitted on to the next neuron.

In a simple neural network, the neuron "fires" and activates the connected neurons if the sum of the received data inputs is greater than a certain threshold value. These summations decide how far a signal must propagate through the network in order to affect the final output. A node with a heavier weight will have a greater impact on the next layer of nodes. The weighted inputs are compiled in the last layer to produce an output. The neural network weights are adjusted by an algorithm based on how precisely it recognizes a particular pattern.

2.2.1 Advantages of using Deep Learning

- Manual feature extraction is not necessary for the deep learning model, thus saving time and effort for the user.
- Deep learning techniques are robust. They learn natural variations in data automatically.
- We can utilize the same neural network approach for training multiple applications.
- GPU-based massive parallel calculations can handle vast amounts of data. Moreover, these networks deliver better performance results when the amount of data is huge.
- The deep learning architecture is adaptable to new issues in the future.

Deep learning models usually work with crisp values in their computations. But fuzzy logic is more helpful, when we need to find how accurate is a given statement. Fuzzy logic comes into play when we want to capture uncertainty and imprecision within an environment. We will cover the details of the fuzzy logic in the section 2.3.

2.3 Fuzzy Systems

Sometimes, it's hard to conclude whether a statement is true or false in specific decision-making problems. Given the uncertainty, fuzzy logic can provide flexibility in such reasoning instances. "Fuzzy" means indefinite, uncertain, unclear, or vague. Fuzzy logic is a computing approach that generates output based on degrees of possibility rather than definite categories. Fuzzy logic was proposed by Lotfi A. Zadeh et al. [25] of the University of California at Berkeley. The introduction of fuzzy sets theory helps in modeling vague and ambiguous real-world problems. The imprecise terms in natural language can be efficiently represented and processed using fuzzy sets and fuzzy logic theories.
A fuzzy set is a mapping of the elements of a universe of discourse (X) to the unit interval $[0, 1]$. This mapping is termed a membership function [15].

$$A : X \rightarrow [0, 1]$$

In data analysis applications, membership grade can be interpreted as a degree of resemblance, preference, and uncertainty. When discovering similarities, membership grade represents the degree of agreement of an element in the universe of discourse $(x \in X)$ with representative elements of A. Fuzzy if-then rules are expressions represented as IF A THEN B, where A is a fuzzy set and B is either a function of inputs or a fuzzy set. These rules are unique in that they try to incorporate human-level decision-making procedures into a system in order to capture unpredictability and imprecision of the environment. Many kinds of research were being conducted on fuzzy expert systems integration of fuzzy logic with neural networks to build more efficient learning models.

We can introduce uncertainity and imprecision in our neural networks by fusing it with fuzzy logic. We will discuss about such deep fuzzy neural networks in the upcoming section 2.4

2.4 Deep Fuzzy Neural Network

In the domain of artificial intelligence, neuro-fuzzy refers to the integration of artificial neural networks with fuzzy logic. It is a hybrid learning machine that exploits the approximation techniques and learning algorithms from neural networks to determine the parameters of a fuzzy system (i.e., fuzzy sets, fuzzy rules). It combines the learning capabilities of a neural network with the noise-handling abilities of Fuzzy Learning. These networks differ from common neural networks in terms of connection weights, propagation, and activation functions. The weights between the layers can be represented as fuzzy sets, with the membership value in each set decided by the relative weights, which were altered using particular neural network training algorithms. Usually, transfer functions are continuous and pass real values through the network to the output layer.

These were interpreted as degrees of membership in fuzzy sets based on the firing of fuzzy rules in the hidden layer.

As already known, every neuron in ANN is connected with other neurons through connection links, and each link is associated with weight-holding information about the input signal. Fuzzy logic is largely used to define these weights from fuzzy sets. We already know that we improve the performance of neural networks by training and learning. When we employ fuzzy logic in neural networks, we don't need values to be crisp, and the processing can be done in parallel. Modern neuro fuzzy systems are usually modeled using special multilayer feed-forward neural networks, for example, models like ANFIS [13], GARIC [4], FuNe [11], or NEFCLASS and NEFCON [16].

The strength of neuro-fuzzy systems is examined through the lens of two contradicting fuzzy modeling requirements: interpretability and accuracy. In real-world practices, one of the two traits takes precedence. The neuro fuzzy in the fuzzy modeling research field is separated into two areas: fuzzy linguistic modeling, which is focused on interpretability, namely the Mamdani model, and precise fuzzy modeling, which is focused on accuracy, namely the Takagi-Sugeno-Kang model. We follow a modified TSK model design in our proposed solution. Let us have a look at the working of the TSK model in the upcoming section 2.4.1.

2.4.1 Working

Fuzzy rule-based systems are modeled using neuro fuzzy system architectures. One of the most popular neuro fuzzy architectures is ANFIS, proposed by Jang et al. [13]. It is a layered feed-forward network that is based on TSK inferential system. The network parameters are learned by following a combination of gradient descent and least-square estimation. A TSK rule set is defined as [13]:

$$\text{If } x_1 \text{ is } A_{1n} \text{ and } x_2 \text{ is } A_{2n} \text{ And ... } x_d \text{ is } A_{dn} \text{ then } y_n = f_n(x)$$

where $n = 1, 2, ..., N$ and N represent the number of fuzzy rules, x_i represents i^{th} input variable, A_{in} denotes a fuzzy set for i^{th} input of the n^{th} rule And serves as a fuzzy conjunction operator, and $f_n(.)$ denotes the output

of n^{th} rule. We calculate the output of the system as:

$$y = \frac{\sum_{n=1}^{N} p_n(x).f_n(x)}{\sum_{n=1}^{N} p_n(x)} = \sum_{n=1}^{N} \overline{p_n(x)}.f_n(x)$$

where $p_n(x) = \prod_{i=1}^{d} p_{A_{in}}(x_i)$ and $p_{A_{in}}(xi)$ denotes membership grade, which measures the degree of similarity between x_i and A_{in}. Layers in the ANFIS network are given as follows [13]:

- Layer 1: Calculation of Membership grade of input: $p_{A_{in}}$
- Layer 2: Firing strength of n^{th} rule is obtained as: $\prod_{i=1}^{d} p_{A_{in}}(x_i)$
- Layer 3: Normalizing firing strength of each rule: $\overline{p_{in}} = \frac{p_{in}}{\sum_{j=1}^{N} p_{ij}}$
- Layer 4: Calculation of output of each rule: $\overline{p_n(x)}.f_n(x)$
- Layer 5: Calculation of final output of the network: $\sum_{n=1}^{N} \overline{p_n(x)}.f_n(x)$

Our proposed approach, incorporates the idea behind these layers with some modifications in the fuzzy operations used. We will now go through the clustering techniques to identify anomalous frames. In the next section 2.5, we will discuss about the K-Means clustering technique.

2.5 K-Means Clustering

In the domain of machine learning problems, unsupervised learning algorithms deal with the set of problems with the unlabelled dataset. Clustering is a very popular technique of unsupervised learning that aims to group similar objects into the same clusters and more distinct objects into different clusters. The similarity amongst the group of objects is computed using various similarity

measures like density, Euclidean distance, Minkowski distance, and Manhattan distance. Clustering has found its applications in the domains of data mining, market research, outlier detection, pattern recognition, forecasting, damage detection, document analysis, and more.

K-Means is a popular clustering method, first proposed in MacQueen et al. [18]. K-Means is used to group data points into K-clusters. The similarity measure used in this algorithm is Euclidean distance or $L2$ distance. The number of clusters K is chosen to minimize the variation within the cluster. K-Means algorithm works by reducing the objective function - which is the sum of distances of all the points from their cluster centroids.

The algorithm begins with randomly initializing K cluster centroids. Then, assign points to a cluster for which the centroid is the nearest. Then re-calculate the cluster centroids by taking the mean of the points assigned to each group. The algorithm proceeds iteratively till it converges when the cluster centroids in two consecutive iterations do not change. K-Means algorithm guarantees convergence, but not necessarily to the same global optimum. After every iteration, the new cluster centroids can be the same as the previous cluster centroids or different. If the cluster centroids change, then the new centroids have reduced cost or objective function value. If the centroids are the same, then they will remain unchanged for all the subsequent iterations. This guarantees that the algorithm converges. Algorithm 1 defines the K-Means algorithm below.

Notation	Description
x_i	i^{th} data point
N	number of data points
K	number of clusters
μ_i	centroid of i^{th} cluster
T	number of iterations for algorithm to converge
z_i	cluster to which x_i data point is assigned

Table 1: Major Math Symbols- K-Means Clustering

Algorithm 1 K-Means Algorithm

Require: Data points: $\{x_1, x_2..., x_N\}$, Number of clusters: K
Ensure: Set of K clusters

Initialize cluster centroids μ_1,μ_2...μ_K randomly
$T = 0$
repeat
    $T = T + 1$
    for $i = 1$ to N **do**
        $z_i = argmin_k ||\mu_k - x_i||^2$ ▷ Assign data points to clusters
    end for
    for $j = 1$ to K **do**
        $\mu_j = \frac{\sum_{i=1}^{N} \{z_i=k\} x_i}{\sum_{i=1}^{N} \{z_i=k\}}$ ▷ Update Cluster Centroids
    end for
until convergence

The asymptotic time complexity of the K-Means Algorithm is $\mathcal{O}(NKT)$. K-Means is a hard clustering method where the clusters are separated by a well-defined boundary, and each data point belongs to any one of the K clusters. In hard clustering methods, data points are grouped in an exclusive way, i.e., a data point belongs to a particular cluster and cannot belong to any other cluster. $K-$ Means algorithm doesn't handle noisy data and is sensitive to the initial cluster centroids chosen. The Fuzzy C-Means Clustering comes to rescue in such cases. We will cover the details of Fuzzy C-Means Clustering in the next section 2.6.

2.6 Fuzzy C-Means Clustering

In real-life problems, clusters tend to have overlapping boundaries. FCM clustering algorithm is a very popular soft clustering algorithm. The Fuzzy theory was introduced by Zadeh et al. [25] and later was introduced into clustering. The basics of fuzzy clustering were proposed by Bellman et al. [3] and Ruspini et al. [24]. There are three types of fuzzy clustering problems: based on fuzzy relations [10], based on fuzzy rule learning [23], and based on objective function optimization.

FCM Algorithm works by reducing the value of the objective function [5]. Here, C depicts the total number of clusters in the current problem. Each data point is given a fuzzy membership value. This value indicates the probability of a particular data point belonging to a certain cluster.

Unlike K-Means, the data samples don't exclusively belong to only one cluster. The membership values are assigned to each sample based on the distance of that data point from the cluster centroids. Here, for each data point, the summation of membership values equals one.

Fuzzy clustering has been used for image segmentation, classification, and compression, for instance, in the field of medicine for the segmentation of brain MR images.

It is important that we reduce the dimensionality of our data samples before performing clustering. In the next section 2.7, we will discuss about a very popular dimensionality reduction technique called Principal Component Analysis.

2.7 Principal Component Analysis

In machine learning problems, the higher the number of features, the more is the complexity of computation. The curse of dimensionality deals with the various consequences of high-dimensional feature spaces. The term was first used by Richard E. Bellman et al. [22]. Data points in high dimensional space can be considered to be extremely sparse, i.e., every data point is technically distant from the rest of the points. In such cases, distance metrics might not be informative. Also, it can happen that a few features might be irrelevant and can reduce the effect of features that are informative.

Dimensionality reduction is used to transform features from high dimensional space to low dimensional space while still preserving important information.

This helps in data visualization and also cluster analysis. Some popular methods for dimensionality reduction are Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA), Generalized Discriminant Analysis (GDA), non-negative matrix factorization, etc.

Principal Component Analysis was introduced by Karl Pearson et al. [20]. The method aims to maximize the variance in low-dimensional space. We try to represent the data points in terms of new variables called principal components. The new variables are chosen to represent the directions of the maximal amount of variance of data points. Amongst all such computed variables, only the variables with the highest variance are chosen to be the principal components. Most of the information of the data points is squeezed into these few variables, and to have distinct information captures; we try to have uncorrelated variables as the principal components.

Below are the steps involved in PCA [12]:

1. *Standardization*: This step standardizes the feature variables so that all the values contribute equally to the analysis. It transforms all variables to the same scale.

Notation	Description
x_{ij}	i^{th} data point value for j^{th} feature
μ_j	mean of j^{th} feature across all data points
σ_j	standard deviation for j^{th} feature across all data points
z_{ij}	value of x_{ij} after standardization

Table 2: Major Math Symbols - Standardization

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

2. *Covariance Matrix Computation*: Correlation between variables indicates information redundancy. A covariance matrix is a table that contains the covariance values of all possible pairs of features. If the data points have n features, then the covariance matrix is an $n \times n$ matrix. In the covariance matrix, the values on the diagonal of the matrix indicate the variance of

individual features.

3. *Eigenvector and Eigenvalue Computation*: Principal components are chosen to maximize the variance of the data points. The covariance matrix calculated in the previous step is given as an input to calculate eigenvectors. These eigenvectors are in the direction of maximum variance. So they can be chosen as the principal components. The variance along these principal components is given by the value of the eigenvalue associated with each of these eigenvectors. The greater the eigenvalue, the greater the variance, more significant is the corresponding principal component. Arrange the eigenvectors using the eigenvalues and pick the variables with the highest eigenvalues as the principal components.
4. *Recast data points along the principal components*: In the last step, the aim is to recast the data points from the original axes along the principal components. The feature vector matrix is constructed using the principal components decided in the previous step.

$$FinalDataSet = FeatureVector^T \times StandardizedOriginalDataSet$$

We will now understand the advantages of using PCA for dimensionality reduction in the upcoming section 2.7.1.

2.7.1 Advantages of Principal Component Analysis

- It helps in better data visualization and cluster analysis.
- It improves the training and performance of the ML model by eliminating correlated variables (that don't contribute to any decision making) and thus redundancy in data.
- Principal components are easily computable since it uses linear algebra.
- It prevents overfitting by decreasing the number of features.
- It results in high variance among the new variables and thus improves visualization and de-noises data.

It is highly likely to have a dataset that is unbalanced. This makes it necessary to check if the dataset is biased towards any single class as it affects the performance of our learning models. We will cover the techniques to handle imbalanced dataset in the upcoming section 2.8.

2.8 Handling Imbalanced Dataset

While dealing with classification scenarios, it is common to encounter an imbalanced dataset. A dataset is imbalanced if the number of samples belonging to one of the classification classes is significantly greater than the number of samples belonging to other classification classes. In such a case, analyzing the performance of the designed model in terms of accuracy as a performance metric can be misleading. The model can produce very good accuracy by classifying all the samples into a single majority class. Hence it becomes important to balance the dataset so as to correctly classify the minority class.

Resampling techniques are usually employed on the dataset to balance the number of samples of both the majority and minority classes. There are two different ways of resampling an imbalanced dataset, namely, oversampling and undersampling. Oversampling techniques are used to increase the number of samples in the minority classes, while undersampling techniques are used to decrease the number of samples in the majority class.

Oversampling Techniques: Random oversampling is a popular method used to oversample the minority class by selecting random samples and duplicating them. At times, all the samples of the minority class are duplicated. Synthetic Minority Over-sampling Technique (SMOTE) and Adaptive Synthetic Sampling approach(ADASYN) are a few more popular oversampling approaches.

Undersampling Techniques: Random undersampling involves randomly selecting samples from the majority class and deleting the data points from

the dataset. This is done until the number of samples in the majority class equals the number of samples in the minority class. Some other popular undersampling methods include Near Miss Undersampling [26] and TOMEK links for undersampling, amongst many others.

In the current work, we implement SMOTE for oversampling our dataset, attributing to its fast and efficient implementation. SMOTE was proposed by Nitesh Chawla et al. [7]. The following are the steps involved in SMOTE:

1. Pick a data point, p_i, randomly from the minority class of the dataset.
2. Find the k nearest neighbors of the randomly chosen point p_i in the previous step. The value of k is usually selected as 5.
3. Pick a point p_{ij} randomly from the set of k nearest neighbors found in the previous step.
4. Draw a line joining p_i and p_{ij}. Select a point randomly on this line. The new point can be represented as $p_{new} = \alpha\, p_{ij} + (1 - \alpha)p_i$.

This new point p_{new} obtained in the last step is the synthetically generated sample and is added to the dataset.

In this chapter 2, we have discussed the research and progress in different domains. Primarily, Video Summarization techniques, Deep learning models, Fuzzy Systems and Deep Neuro Fuzzy Networks, Hard and Soft Clustering methods were understood in detail. In the upcoming chapter 3 we will interpret the extension of the literature discussed to achieve our objectives discussed in section 1.3.

Chapter 3

Design and Analysis of Algorithms

In this chapter, we will discuss the various steps involved in designing our model which include Data Preprocessing, Feature Extraction, Fuzzy C-Means clustering, Deep Neuro Fuzzy model for classification and Dynamic Image generation. We will pictorially explain the various algorithms used. We begin with the various steps involved in data preprocessing in the section 3.1 .

3.1 Data Preprocessing

The 2 major steps in data preprocessing include frame generation and feature extraction. In our experiment, we considered UCF-Crime Dataset [1]. The videos are processed to generate frames of dimensions (224, 224, 3) to facilitate the network needs. Feature extraction is performed on these video frames to identify distinguishing characteristics. We use pre-trained models of ResNet50 and VGG16 to extract features. These models process the input to give an output vector of 1000 values. Figure 1 summarizes the process described above.

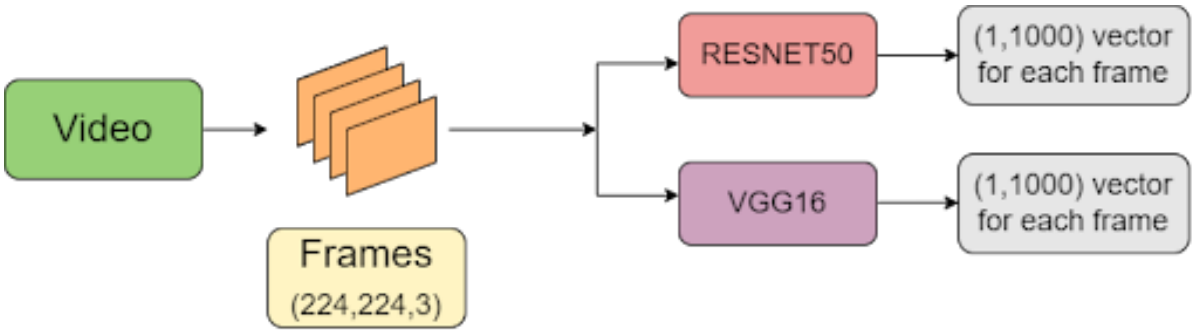

Figure 1: Block diagram for Feature Extraction

The VGG model was first proposed by the Visual Geometry Group. VGG16

is a Deep Convolutional Neural Network model with 16 weight layers. The model comprises multiple convolution and max-pooling layers. Towards the end, the network includes fully connected layers and softmax. VGG16 is a very large model with nearly 138 million parameters. Figure 2 depicts the various layers in the VGG16 architecture.

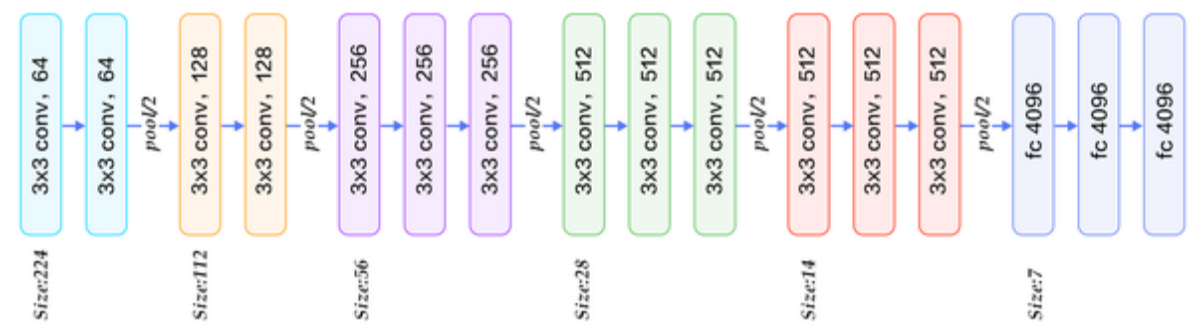

Figure 2: Architecture of VGG16 model [2]

ResNet stands for Residual Networks. ResNet50 model has 50 layers, out of which there are 48 convolution layers along with one max-pooling layer and one average pooling layer. The model consists comprises of skip-connections as their basic unit. The output of previous layers is added and fed as an input to the following layers. Figure 3 illustrates the architecture of the ResNet50 model.

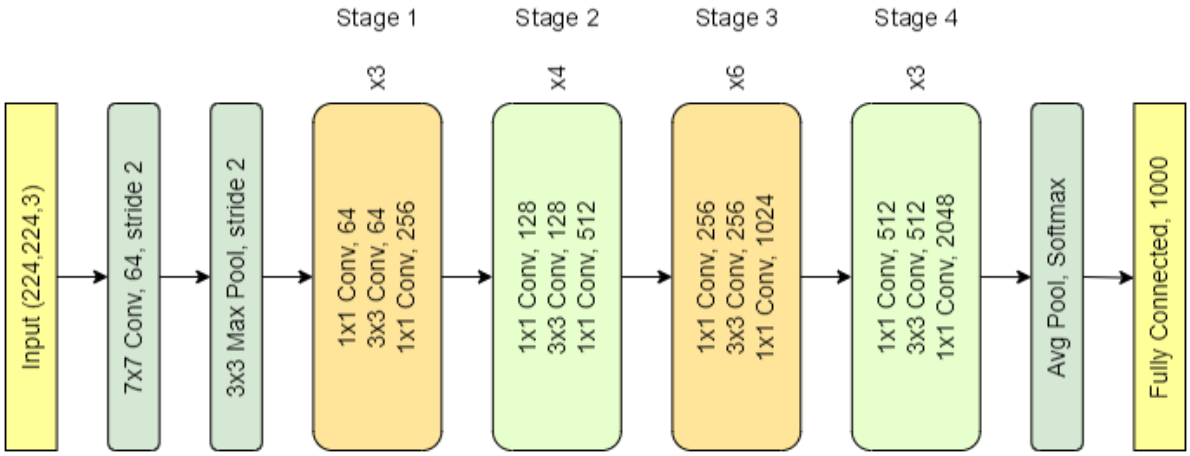

Figure 3: Architecture of ResNet50 model

Both the pre-trained networks can classify images into 1000 object categories present in the ImageNet dataset.

The features extracted are used to perform clustering. The section 3.2 discusses the FCM clustering algorithm used to identify the anomalous frames.

3.2 Fuzzy Clustering

As discussed in section 2.7, we use principal component analysis to reduce the dimensions of our feature vector. This is the input given to Fuzzy C-Means Clustering Algorithm. We want to cluster our frames into 2 clusters: Normal and Anomalous clusters. Figure 4 shows the various steps involved in Clustering.

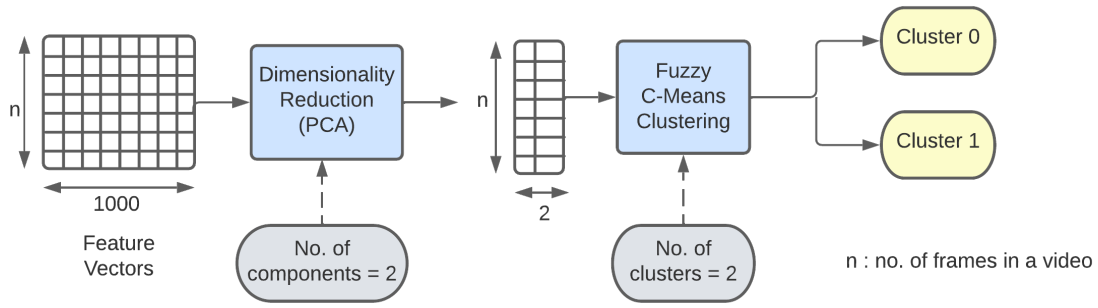

Figure 4: Block Diagram of Clustering

In a given input, let there be N samples; we would want to cluster them into C clusters. The algorithm begins with randomly initializing membership values for all data points. Re-calculate the cluster centroids by using membership values and data points. Then re-calculate the membership values again using the updated centroids and data points. Summation of membership values of each data should be equal to one. The closer the data point is to a cluster centroid, the more likely it is to belong to that cluster. The algorithm proceeds iteratively till the cluster centroids in two consecutive iterations do not change. We will mathematically define FCM Algorithm and the various symbols used.

Notation	Description
x_i	i^{th} data point
N	number of data points
C	number of clusters
p	fuzzification parameter
u_{ij}	membership of i^{th} data point to the j^{th} cluster
T	number of iterations for algorithm to converge
v_j	centroid of j^{th} cluster

Table 3: Major Math Symbols - FCM Clustering

Algorithm 2 Fuzzy C-Means Algorithm

Require: Points: $\{x_1..., x_N\}$, No. of clusters: K, Fuzzification parameter: p

Initialize Membership values μ_{ij} randomly
$T = 0$
repeat
    $T = T + 1$
    for $j = 1$ to C **do**
        $v_j = \frac{\sum_{i=1}^{N} (u_{ij})^p x_i}{\sum_{i=1}^{N} (u_{ij})^p}$ ▷ Update cluster centroid
    end for
    for $i = 1$ to N **do**
        for $j = 1$ to C **do**
            $u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{||x_i - v_j||}{||x_i - v_k||} \right)^{\frac{2}{p-1}}}$ ▷ Update Membership values
        end for
    end for
until convergence

The asymptotic time complexity of the algorithm is $\mathcal{O}(NKT)$.

In the algorithm, u_{ij} is the probability of x_i data point belonging to cluster j. Here p, the fuzzification parameter, is a hyperparameter and has to be chosen optimally. The larger the value of p, the more will be the fuzziness of the clusters. The value of p is usually chosen from the range $\{1, \infty\}$. The most commonly chosen default value for p is 2.

Another method to identify the anomalous frames is classification using the Deep Neuro Fuzzy Network discussed in section 3.3.

3.3 Deep Neuro Fuzzy Network

Two main operations were used to build this network: Fuzzy Inference operation and Fuzzy Pooling operation. We stack up these two operations, along with dense layers to build the deep neuro fuzzy network.

3.3.1 Fuzzy Inference Operation

Firstly, We calculate a matrix of membership grades (P_k) for each fuzzy rule by convolving a fuzzy set with the input image. These membership values lie in the range $[0, 1]$. Each element in P_k gives us the similarity between a subregion of the image and the fuzzy set A_k. These membership grades are normalized over the fuzzy rules set to obtain firing strength. Padding is performed on the whole membership matrix based on the difference between the size of the image and the size of the membership matrix. This is succeeded by a series of convolution operations. The final output is calculated by an element-wise multiplication of the output of the previous steps. Figure 5 shows the flow of operation in Fuzzy Inference Operation. Algorithm 3 given below, defines the steps in the Fuzzy Inference operation in detail.

Notation	Description
I	Array of pixels
K	Number of fuzzy sets in first layer
P_k	Membership Matrix for k^{th} fuzzy set
$\overline{P_k}$	Firing Strength for k^{th} fuzzy set
$F1, F2, F3, F4$	Fuzzy Sets

Table 4: Major Math Symbols- Fuzzy Inference Operation

Algorithm 3 Fuzzy Inference Operation

Require: Image: I, FilterSet1: $F1$, FilterSet2: $F2$, FilterSet3: $F3$, FilterSet4: $F4$

Ensure:

$K = len(F1)$
for $k = 1$ to K **do**
    P_k = $I \circledast F1_k$ ▷ Membership Matrix
    P_k = clip(P_k) ▷ Clipping to [0,1]
end for
for $k = 1$ to K **do**
    for $i = 1$ to $len(P_k)$ **do**
        $\overline{p_{ik}} = \frac{p_{ik}}{\sum_{j=1}^{K} p_{ij}}$ ▷ Firing Strength
    end for
end for
$T_1 = (\overline{P_1}\ \overline{P_2}$ ..... $\overline{P_k})$
$T_2 = Padding(T_1)$ ▷ Padding
$T_2 = T_2 \circledast F2$
$g = T_2 \circledast F3$
$f = I \circledast F4$
$y = g \circledast f$
return y

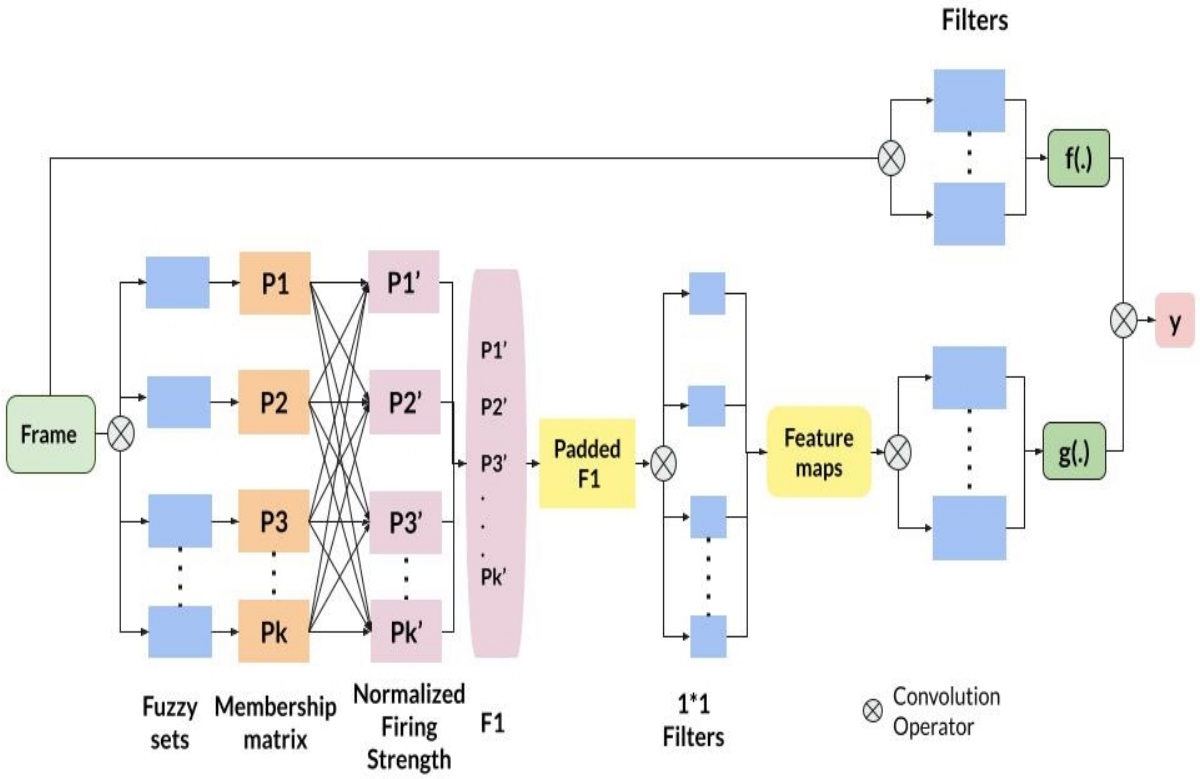

Figure 5: Network for Fuzzy Inference Operation

3.3.2 Fuzzy Pooling

The operations in the Fuzzy Pooling operation are similar to the Fuzzy Inference operation, but they differ in following steps:

- There is no padding in fuzzy pooling operation before convolving with $1*1$ filters.
- In fuzzy pooling operation, feature maps obtained after convolving with $1 * 1$ filters are directly taken as $g(.)$, without any further convolution.

Figure 6 shows the flow of operation in Fuzzy Pooling Operation. Algorithm 4 given below, defines the steps in the Fuzzy Pooling operation in detail.

Notation	Description
I	Array of pixels
K	Number of fuzzy sets in first layer
P_k	Membership Matrix for k^{th} fuzzy set
$\overline{P_k}$	Firing Strength for k^{th} fuzzy set
$F1, F2, F3$	Fuzzy Sets

Table 5: Major Math Symbols- Fuzzy Inference Operation

Algorithm 4 Fuzzy Pooling Operation

Require: Image: I, FilterSet1: $F1$, FilterSet2: $F2$, FilterSet3: $F3$

Ensure:

$K = len(F1)$

for $k = 1$ to K **do**

  P_k = $I \circledast F1_k$ ▷ Membership Matrix

  P_k = clip(P_k) ▷ Clipping to [0,1]

end for

for $k = 1$ to K **do**

  for $i = 1$ to $len(P_k)$ **do**

    $\overline{p_{ik}} = \frac{p_{ik}}{\sum_{j=1}^{K} p_{ij}}$ ▷ Firing Strength

  end for

end for

$T_1 = (\overline{P_1}\ \overline{P_2}$ ..... $\overline{P_k})$

$T_2 = T_2 \circledast F2$

$g = T_2$

$f = I \circledast F3$

$y = g \circledast f$

return y

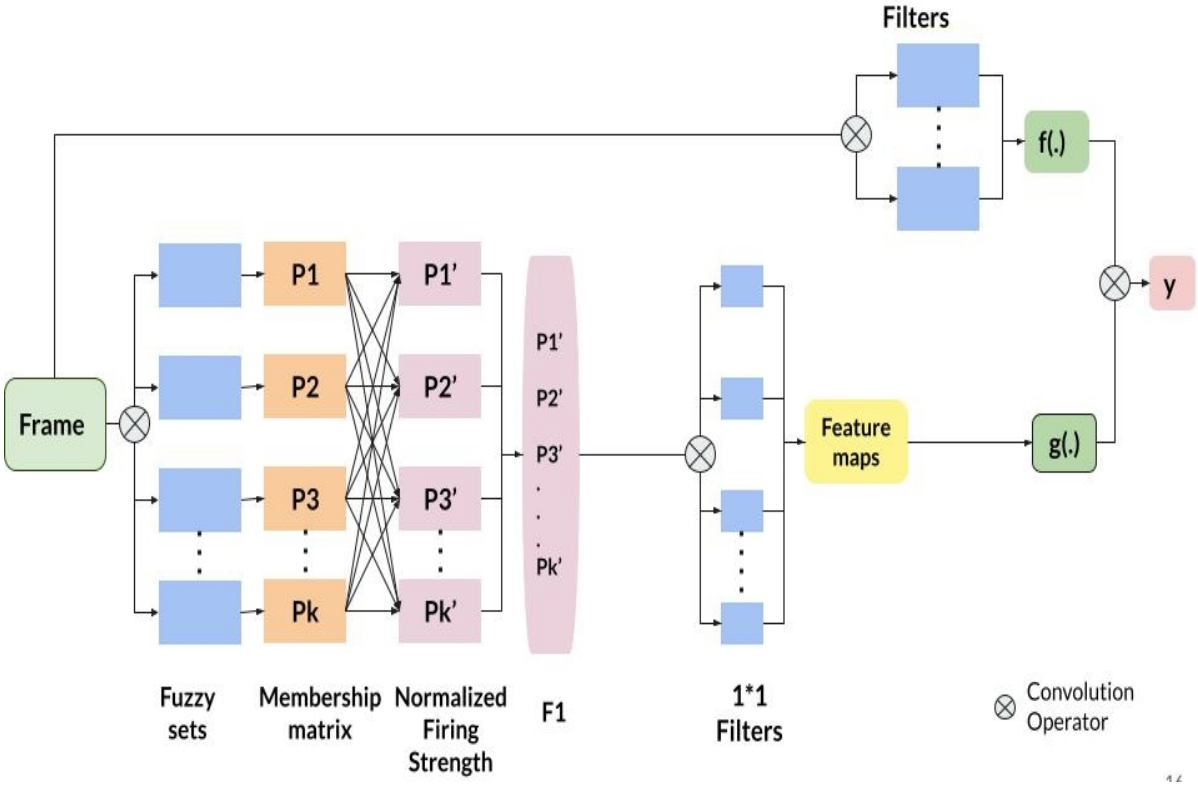

Figure 6: Network for Fuzzy Pooling Operation

These operations of fuzzy inference and fuzzy pooling were incorporated into the layers of Deep Neuro fuzzy architecture. These layers are stacked upon one another to complete the model. This architecture is discussed in the section 3.3.3.

3.3.3 Architecture

The model is built by stacking three types of layers: Fuzzy Inference Layer, Fuzzy Pooling Layer, and Dense Layer. The input to the network is an image of dimensions $(224, 224, 3)$. The number of output units in the final layer depends on the number of output classes; it is two in our case. The two classes are namely, normal and anomalous.

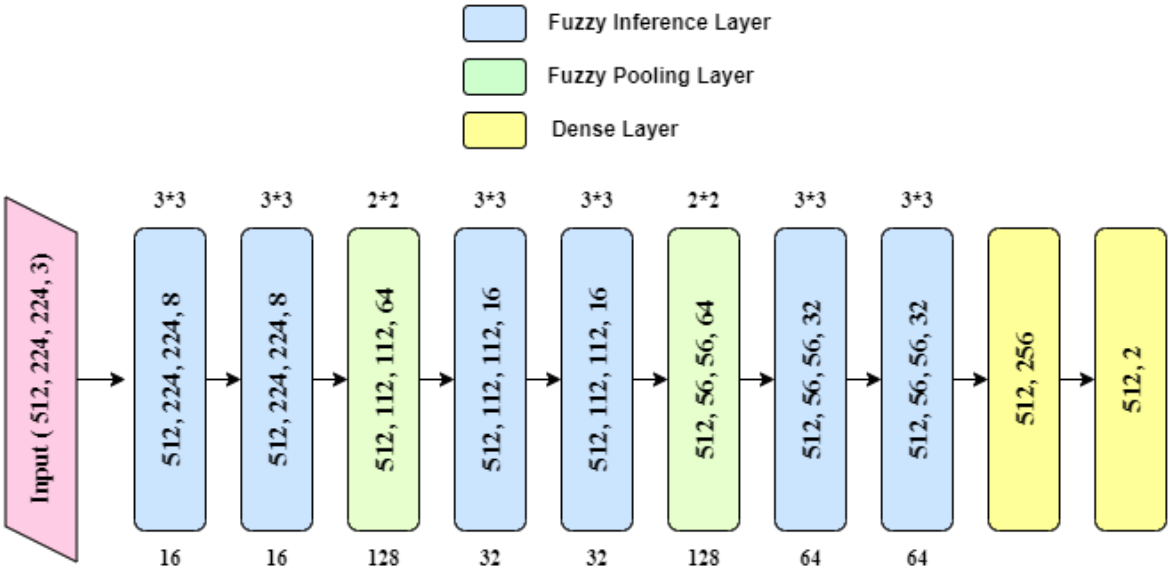

Figure 7: Architecture of Deep Neuro Fuzzy Network

The section 3.3.4 discusses in detail the loss function used and it's significance in multi-class classification.

3.3.4 Loss: Categorical Crossentropy

When training a Machine Learning or a Deep Learning model, loss/cost functions are used to optimize the model. Usually, the goal is to reduce the loss function to the smallest possible value. The lower the loss, the better is the model. Cross-Entropy loss is the most important cost function used to optimize multi-class classification tasks. It is particularly suitable for classification problems since one example may be deemed to belong to a certain category with probability 1 and to other categories with probability 0.

The Categorical Crossentropy loss function determines the loss of an example by evaluating the following sum:

$$Loss = -\sum_{j=1}^{n} y_j.\log\hat{y}_j$$

where $\hat{y}_j$ is the j^{th} scalar value in the model output, y_j is the corresponding target value, and n is the output size. In this context, y_j is the probability or

likelihood that event j occurs, and the sum of all y_j is 1, implying that only one event may occur. The minus sign assures that the loss becomes smaller when the distributions come closer to each other.

The Categorical Crossentropy loss function primarily recommends using Softmax as an activation function. The softmax activation rescales the model output to make it more appropriate. The purpose of the Cross-Entropy is to take the output probabilities and measure the distance from the true values.

The section 3.3.5 describes the optimizer used during the training of our deep neuro fuzzy model.

3.3.5 Optimizer: Adam

Gradient Descent is an optimization method that uses a first-order derivative to minimize the value of the objective or loss function. But for large datasets, it usually takes a huge number of computation steps. Adam stands for Adaptive Moment Estimation. Adam optimizer is used to enhance gradient descent to reach the optimum value in a fewer number of steps. It computes adaptive learning rates for each parameter. It is a combination of RMS propagation and Momentum optimizers.

After identifying the frames as anomalous, we use these to summarize a video by generating a single dynamic image. We discuss the mathematical formulation and also the details related to dynamic image generation in the section 3.4.

3.4 Dynamic Image

As discussed in section 2.1, we use Dynamic image generation to summarize a video using rank pooling techniques as formulated by Bilen et al. [6].

Let us consider a video to have frames, say, $F_1, F_2, ...F_C$. The number of frames in the whole video is C. We will consider the feature vector extracted

from F_i as $\phi(F_i) \in \mathbb{R}^d$. The 3 RGB components of an image are given as input to the operator $\phi(F_i)$. It stacks the pixel values onto a single vector.

Each frame, F_i is given a rank, and a video is represented in terms of the ranks of its frames. Let $\overline{\phi_i} = \frac{1}{i}\sum_{j=1}^{i}\phi(F_j)$. Let each frame F_i be given a score $\psi(i,\lambda) = \langle \overline{\phi_i}, \lambda \rangle$ where $\lambda \in \mathbb{R}^d$ is a vector of parameters. Frames that appear earlier in the video are given lower scores than the frames that appear later in the video, *i.e.*, $\psi(i,\lambda) < \psi(j,\lambda)$ for $i < j$. These function parameters λ^* are learned. The scores thus achieved reflect the rank of the frames in the video. Let us use the idea of RankSVM to learn λ^*. The optimization problem can be formulated as:

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}}\, U(\lambda). \tag{1}$$

$$U(\lambda) = \frac{\alpha}{2}||\lambda||^2 + \frac{2}{C(C-1)} \times \sum_{i>j} max\{1 + \psi(j,\lambda) - \psi(i,\lambda), 0\}$$

The second term is a soft counting hinge loss function that counts incorrectly ranked pairs of points. For this, we consider all points whose score values differ by at least one unit, *i.e.*, $\psi(i,\lambda) - \psi(j,\lambda) > 1$. We call this process of computing λ^* rank pooling.

Computing a dynamic image involves solving the optimization problem in eq.(1). To solve the optimization problem much faster, consider the first step in gradient-descent optimization. Starting with $\lambda = \vec{0}$, we obtain:

$$\lambda^* = \sum_{i=1}^{C} w_i \phi(F_i)$$

where the coefficients w_i are given by:

$$\delta_i = 2(C - i + 1) - (i + 1)(\tau_i - \tau_{\text{i-1}}) \tag{2}$$

where $\tau_i = \sum_{j=1}^{i} \frac{1}{j}$ is the i^{th} Harmonic number, and $\tau_0 = 0$. After approximation, the calculation of λ^* is equivalent to the weighted summation of the feature vectors $\phi(F_i)$. This is called approximate rank pooling.

The chapter 3 explained the various steps involved in the pipeline of our model. In further chapters we will understand the implementation of our proposed plan. The chapter 4 explains in detailed the various datasets used along with the software and hardware specifications.

Chapter 4

Experimental Walkthrough

In this chapter, we will explain the datasets used and the experimental setup, along with the specifications of our system used. The section 4.1 will mention the various datasets used to train and test our proposed method along with a brief overview of its contents.

4.1 Data Description

To evaluate the proposed method, the UCF-Crime dataset is used. The dataset comprises surveillance videos of varying lengths which cover 13 real-world anomaly classes. These classes include various criminal activities concerning public safety like Arrest, Burglary, Explosion, Assault, Road Accident, Arson, Robbery, Shoplifting, Fighting, Stealing, Abuse, Vandalism, and Shooting [1]. To evaluate the performance of Deep Neuro Fuzzy network, MNIST, Caltech101 and UCF11 datasets were used.

MNIST: Modified National Institute of Standards and Technology database(MNIST database) is a database of handwritten digits cummulated in the year 1998. It is a compilation of different images of the ten handwritten digits. It is a smaller version of a bigger dataset called NIST. It has around 60,000 training samples and 10,000 testing samples. All the black and white images from the original NIST database are resized to a standard size of (28,28) and are normalized.

UCF11: UCF11 dataset is a collection of videos from 11 action categories namely, basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog[17]. The dataset was compiled by UCF Center for Research in Computer Vision and was initially called UCF: Youtube Action Class dataset. Each class has more nearly 150 videos which

are divided into 25 subgroups having more than 4 videos each, all of which are in .mpeg4 format. It is a difficult dataset to work with because of the huge amount of variations in the camera angle, varying proportion of background and foreground, illumination and object poses. The frames per second in all the videos is adjusted to 29.97 fps.

Caltech101: Caltech101 dataset was compiled in the year 2003 at the California Institute of Technology by Marco Andreetto, Fei-Fei and others. It is a dataset with digital images from 101 object classes and one background clutter class. It is a very popular dataset used extensively in tasks related to Computer Vision domain especially object classification or image recognition problems. The object classes in the include crabs, ants, watches, faces amongst the others. The total number of images in the dataset is around 9146. Each object class has minimum 40 images and a maximum of 800 images. Faces is the most used class and has the highest number of samples. The dimensions of all the images is about (300, 200), thus reducing user preprocessing tasks.The dataset has detailed human labelled annotations which describe the bounding box of the objects in the images. The images in the dataset have a uniform presentation.

The Deep Neuro Fuzzy model was tested on above datasets, while the FCM clustering algorithm was implemented using only the UCF-Crime dataset.

4.2 Hardware and Software Specifications

All the algorithms and models were coded in python-3.7, and written on jupyter notebook and Google Colab. The deep neuro-fuzzy model was developed using TensorFlow-2.8, Keras-2.8, and PyTorch-1.10. While using Google Colab, the Tesla K80 GPU was used to enhance the speed of computation.

The chapter 5 will explain the results and the various hyperparameters decided upon.

Chapter 5

Experimentation and Results

In the current chapter, we will discuss the experimentation and results of the clustering algorithm, classification model, and dynamic image summarization. We will further compare the performance of Clustering and the classification models.

5.1 Performance Evaluation Metrics

The various performance metrics to evaluate our models are discussed below.

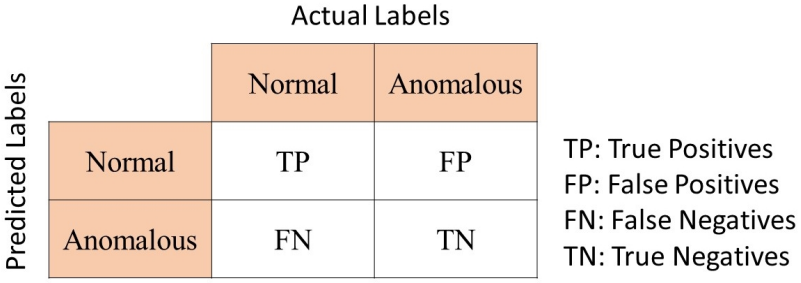

Figure 8: Confusion Matrix

We define :

- *Accuracy:* $\frac{TN+TP}{TP+TN+FP+FN}$
- *Specificity:* $\frac{TN}{TN+FP}$
- *Negative Predictive Value:* $\frac{TN}{TN+FN}$

Accuracy indicates the number of samples correctly classified amongst all the data points. Specificity indicates the number of actual negatives correctly predicted as negatives. Negative Predictive Value is the number of actual negative samples amongst the samples predicted as negatives.

In our field of work it is important to correctly identify all the anomalous frames as anomalous, i.e., misclassify very few anomalous frames as normal frames. The value FP in the confusion matrix indicates the count of such misclassification. Reducing the value of FP indicates the need for a higher value of Specificity metric. Nevertheless, we would also want only a few number of normal frames classified as anomalous. This is indicated by the value of FN in the confusion matrix. We would want to minimize the value of the misclaafication FN, meaning a higher value for the metric Negative Predictive value.

The next section 5.2 will discuss the results achieved using Fuzzy C-Means Clustering model, with the features generated using Resnet50 and VGG16 models.

5.2 Results of Fuzzy C-Means Clustering

Fuzzy C-Means algorithm was used to cluster the frames into normal and anomalous clusters. Dimensionality Reduction was performed on the features extracted using both ResNet50 and VGG16 models using Principal Component Analysis. The dimension of the feature vector reduces from 1000 to 2. The reduced input feature space is given as input to perform clustering.

The results of clustering are discussed in following sections 5.2.1 and 5.2.2.

5.2.1 Clustering using features extracted from ResNet50

Figure 9 includes a plot of the average accuracy of Fuzzy C-Means Clustering using the features extracted from the ResNet50 model. The Average Accuracy across all classes is 0.6009

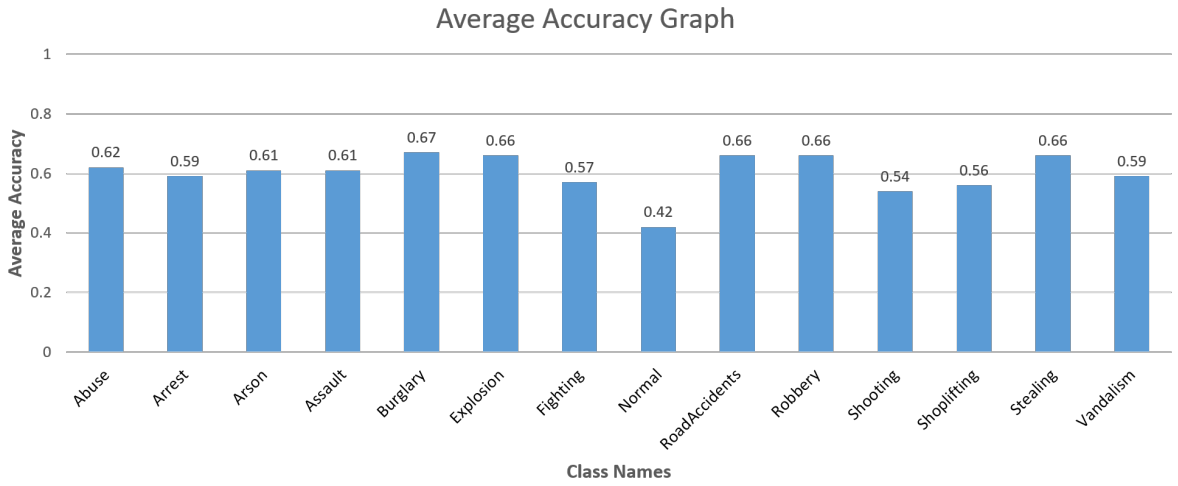

Figure 9: Average Accuracy Plot (FCM using Features from ResNet50)

5.2.2 Clustering using features extracted from VGG16

Figure 10 includes a plot of the average accuracy of Fuzzy C-Means Clustering using the features extracted from the VGG16 model. The Average Accuracy across all classes is 0.6179

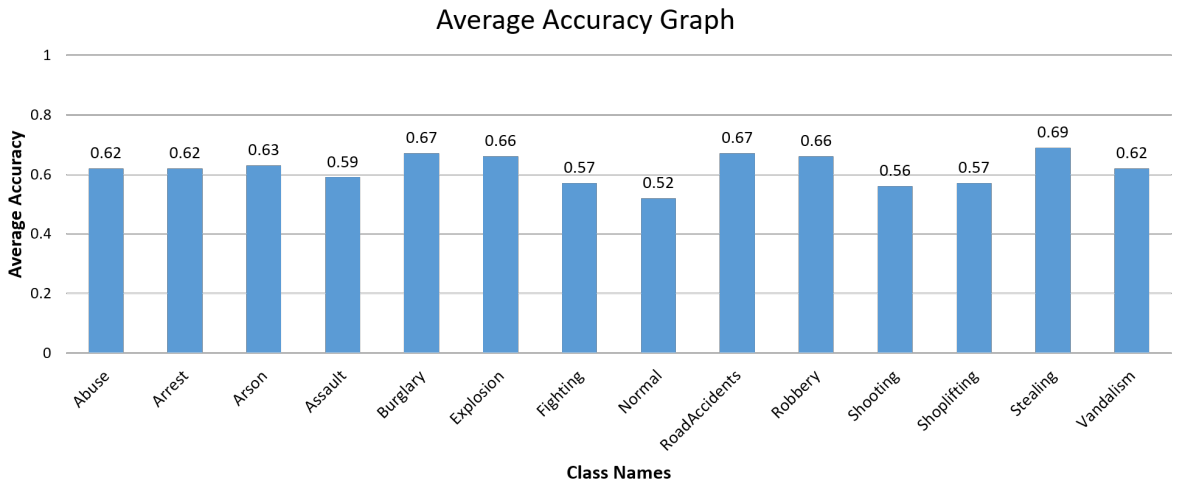

Figure 10: Average Accuracy Plot (FCM using Features from VGG16)

The performance of FCM is compared from both sections 5.2.1 and 5.2.2. The accuracies of both the cases are comparable, and it can be inferred that both the models VGG16 and ResNet50 can be used for feature extraction.

The anomalous cluster obtained after performing Fuzzy C-Means Clustering algorithm is given as input to generate dynamic images. The identification of anomalous frames is also done using Deep Neuro Fuzzy Network. The results of the same are discussed in the upcoming section 5.3

5.3 Results of Deep Neuro Fuzzy Network

The Deep Neuro Fuzzy model was trained and tested on various datasets to understand the effect of varying different hyperparameters like batch size, the number of layers and chunk size.

The Deep Neuro Fuzzy model was further trained on the videos of the Explosion class of the UCF Crime dataset. Considering the huge amount of input data to be trained, it is efficient to make use of the input pipeline to feed the input images in chunks. The chunk size selected is 5000. The input dataset was divided as follows:

- Training split: 75% of the total dataset
- Testing split: 25% of the total dataset
- Validation split: 20% of the training dataset

The batch size selected while training the model is 512. Table 6 summarizes the various datasets experimented with and the results obtained.

Dataset	Description	Training Accuracy	Testing Accuracy	Validation Accuracy
MNIST	Handwritten digits (10 classes)	0.9829	0.9894	0.9915
UCF11	Action classes like cycling, driving (11 classes)	0.8776	0.8961	0.8939
CALTECH101	Object classes like ant, chair, crab; Background clutter class (102 classes)	0.7042	0.5385	0.5454
Explosion	Subset of the UCF Crime dataset with 50 videos (2 classes)	0.9698	0.5401	0.9881

Table 6: Results: Deep Neuro Fuzzy Network

5.4 Performance Comparison

Table 7 compares the performance of the Fuzzy C-Means clustering Algorithm and the Deep Neuro Fuzzy Classification models using the various performance metrics discussed in 5.1.

Performance Metric	Clustering	Classification
Training Accuracy	0.634	0.969
Testing Accuracy	0.641	0.540
Training Negative Predictive Value	0.685	0.972
Testing Negative Predictive Value	0.618	0.714
Testing Specificity	0.541	0.422

Table 7: Performance Comparison Table

It can be observed that the classification model has produced a much greater training accuracy than the clustering model, but couldn't achieve higher testing accuracy. The reason that can be attributed to the same is the possible overfitting of the deep neuro fuzzy model.

5.5 Results of Dynamic Image

The output of clustering and classification models is given as input to generate dynamic images.

5.5.1 Dynamic Image using the output of Clustering

The following are the sample output dynamic images generated using the frames clustered into the anomalous cluster from the video Explosion008_x264.mp4 by the FCM clustering algorithm. Video cube size considered is 32.

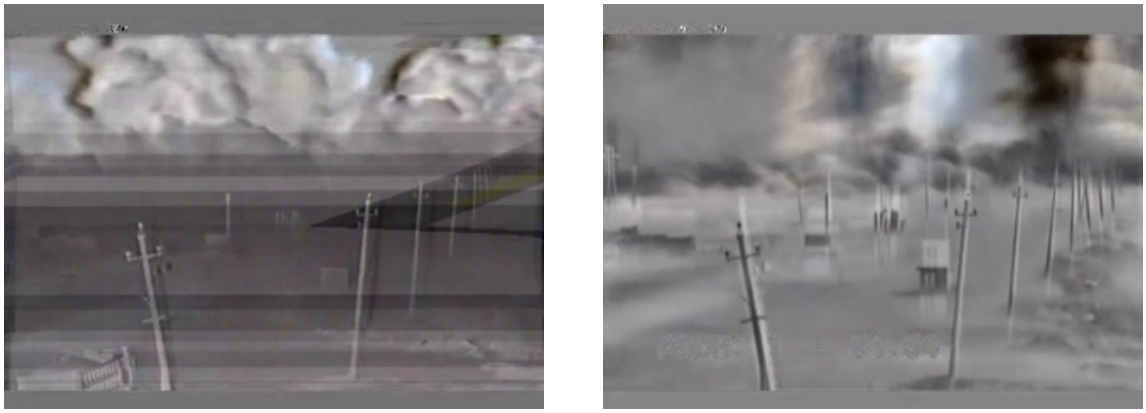

Figure 11: Dynamic Images using the output of FCM Algorithm

5.5.2 Dynamic Image using the output of Classification

The following are the sample output dynamic images generated using the frames classified into the anomalous class from the video Explosion008_x264.mp4 by the Deep Neuro Fuzzy Network. Video cube size considered is 32.

Figure 12: Dynamic Images using the output of classification

From Figure 12 and Figure 11, we can observe huge clouds of smoke that arose due to an explosion activity. The anomalous frames for the video considered would be the frames with the explosion event. These frames were summarized into a single RBG image.

Chapter 6

Conclusion and Future Scope

In the current chapter, we will discuss the conclusion derived from the results obtained. We will also discuss the future scope of the project.

6.1 Conclusion

In this project, we generated a dynamic image summary of the anomalous content in a surveillance video taken from the UCF-Crime dataset. The anomalous content in the video was selected using two methods: Fuzzy C-Means Clustering and Deep Neuro Fuzzy Classification model. The input videos were preprocessed and then given as input to these methods. We observe that the Clustering model performs better for unknown testing data than the Classification model despite the very high training accuracy. It can be treated as a case of model overfitting, and we believe that the classification model could produce better results when re-trained with a better set of hyperparameters.

6.2 Future Scope

The Deep Neuro Fuzzy model can be extended and trained extensively on all the classes of the UCF Crime dataset. Generative Adversarial Networks(GANs) can be used to capture key frames in a video. A GAN model has two blocks within, namely, a Generator block and a Discriminator block. GANs can be used for video summarization in two ways. First, given a video input to the generator block of a GAN model, the generator creates a dynamic image as an output. Second, we can use GAN for the key frame selection. These key frames can be used to generate a dynamic image. Using GANs will make the process of video summarization unsupervised learning. While one block tries to summarize the video, the other block tries to distinguish the summary from the original video. Learning happens to make the video and its summary indistinguishable.

References

[1] Real-world anomaly detection in surveillance videos. https://www.crcv.ucf.edu/projects/real-world/.

[2] Vgg16 architecture. https://iq.opengenus.org/vgg16/.

[3] Kalaba R.A. Zadeh L.A Bellman, R.E. *Abstraction and pattern classification.: J. Math.* Anal. Appl. 13, 1966.

[4] H.R. Berenji and P. Khedkar. Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 3(5):724–740, 1992.

[5] Bezdek. J.C.: Pattern recognition with fuzzy objective function algorithms. *Plenum Press, New York*, 1981.

[6] Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. Action recognition with dynamic image networks. *CoRR*, abs/1612.00738, 2016.

[7] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, jun 2002.

[8] Basura Fernando, Efstratios Gavves, Jose Oramas M, Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. 06 2015.

[9] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[10] F. Guoyao. Optimization methods for fuzzy clustering. *Fuzzy Sets Syst. 93*, pages 301–309, 1998.

[11] S.K. Halgamuge and M. Glesner. Neural networks in designing fuzzy systems for real world applications. *Fuzzy Sets and Systems*, 65(1):1–12, 1994.

[12] Zakaria Jaadi. A Step-by-Step Explanation of Principal Component Analysis (PCA). https://builtin.com/data-science/step-step-explanation-principal-component-analysis, 2021.

[13] J.-S.R. Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685, 1993.

[14] Vivekraj K, Debashis Sen, and Balasubramanian Raman. Vector ordering and regression learning-based ranking for dynamic summarisation of user videos. *IET Image Processing*, 14, 12 2020.

[15] George J. Klir and Bo Yuan, editors. *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh*. World Scientific Publishing Co., Inc., USA, 1996.

[16] Rudolf Kruse and Detlef Nauck. Neuro-fuzzy systems. In Okyay Kaynak, Lotfi A. Zadeh, Burhan Türkşen, and Imre J. Rudas, editors, *Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications*, pages 230–259, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[17] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003, 2009.

[18] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[19] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[20] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572, 1901.

[21] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 540–555, Cham, 2014. Springer International Publishing.

[22] Bellman R. Adaptive control processes:. *A Guided Tour Princeton University Press*, 1961.

[23] Zimmermann H.J. Ravi, V. Fuzzy rule based classification with feature selector and modified threshold accepting. *Eur. J. Oper. Res.*, pages 123, 16–28, 2000.

[24] Ruspini. E.H.: A new approach to clustering. *Inf. Control*, 15(1):22–32, 1969.

[25] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[26] J. Ziang. Knn approach to unbalanced data distributions: a case study involving information extraction. *Proc. Int'l. Conf. Machine Learning1 (ICML'03), Workshop Learning from Imbalanced Data Sets*, 2003.