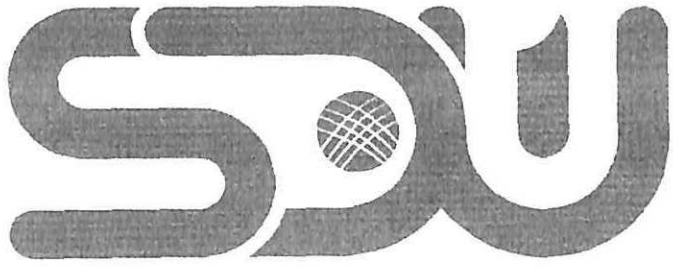
Ministry of Education and Science of the Republic of Kazakhstan
Suleyman Demirel University



Nazerke Manteyeva

Crime type identification from video footage based on 3D convolutional neural network

THESIS

Presented in Partial Fulfillment of the Degree
of Master of Science in Computer Science
(degree code: 7M06102)
Department of Computer Science
Faculty of Engineering and Natural Sciences

Supervisor: Birzhan Moldagaliyev

Kaskelen, 2022

Suleyman Demirel University

Faculty of Engineering and Natural Sciences

Department of Computer Science

Dean of Faculty

Associate Professor, PhD Zhamanov A.

«20» 06 2022

Topic of the thesis:

Crime type identification from video footage based on 3D convolutional neural network

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science", SDU, 2020-2022

Head of Department

Assoc. Prof. Cemil Turan

Academic Supervisor

Birzhan Moldagaliyev

Master's Student

Nazerke Manteyeva

Kaskelen, 2022

Abstract

In this thesis, we investigate the problem of crime classification given video footage from CCTV cameras. To do this, we will use 3D convolutional neural networks that first intended to be used for recognizing various actions. The unique feature of 3D convolutional neural networks lies in the fact that they capture both spatial and temporal dimensions by using 3D convolution, thus processing the motion data and its outlines. The system discussed in this thesis produces distinct channels for incoming data based on the input sketches. The final part includes representation that aggregates data from all channels. It is shown that the discussed architecture results a high accuracy in identifying the type of criminal offence shown in the underlying video footage.

Аңдатпа

Бұл дипломдық жұмыста біз бейнебақылау камераларынан алынған бейнежазбаларды ескере отырып, қылмысты саралау мәселесін зерттейміз. Бұл әрекетті тану үшін бастапқыда әзірленген 3D конволюционды нейрондық желілерді қолданамыз. 3D CNN 3D конвульсиясын орындау және осылайша әртүрлі іргелес контурларда кодталған қозғалыс деректерін түсіру арқылы кеңістіктік және күнделікті өлшемдерден бөлектейді. Жасалған модель кіріс эскиздерінен деректердің әртүрлі арналарын жасайды, ал соңғы қосу көрінісі барлық арналардағы деректерді біріктіреді. Біз бұл желілер бейнематериалдарда басталған қылмыс түрін анықтауда жоғары дәлдікті ұсынатынын көрсетеміз.

Аннотация

В данной диссертации мы исследуем проблему классификации преступлений по видеоматериалам с камер видеонаблюдения. Для этого мы будем использовать 3D сверточные нейронные сети, изначально разработанные для распознавания действий. 3D CNN выделяет как пространственные, так и мирские измерения, выполняя 3D-свертку и, таким образом, захватывая данные о движении, закодированные в различных соседних контурах. Созданная модель генерирует различные каналы данных из входных эскизов, а окончательное включаемое представление объединяет данные из всех каналов. Мы показываем, что эти сети обеспечивают высокую точность определения типа преступления, начатого на видеозаписи.

Acknowledgements

It is my intention to express my immense gratitude to all members of Suleyman Demirel University (SDU). My time at SDU has been truly amazing. I have made many friends, seen inspirational instructors and experienced the joy of learning.

Contents

Chapter 1

Introduction

1.1 Motivation

The total amount of perpetrator spite has increased dramatically. This can be a terrifying attack where one or more people use weapons to prevent an attack by a single person. This has led to the ubiquitous use of surveillance cameras. This has made the difference specialists spot brutal attacks and take the simple steps in order to diminish the unfortunate impact. However, almost every system today need manual human-based verification of the underlying video snippets to discriminate such scenarios. It is clear that the given state of affairs is impractical and far from efficient by any measure. It is in this context that the proposed study becomes relevant. Having such a sensible framework that can review consistent intelligence records and identify people's wild behavior will contribute well and help the law and the establishment. In the given thesis we will look at crime-related activities through CCTV video footage.

Clearly, almost every major city has significant number of CCTV cameras. This number is only growing and projected to grow in the future. The natural problem is getting these cameras to detect and verify crime scenes. The major complication of working with video-based data lies in the huge amount of storage needed and data density. One should remember that a single second of the given CCTV footage can contain roughly 24 frames that must be examined. Thus, a process of crime scene detection involves working with both spatial dimensions of a single frame and temporal differences between consecutive frames. Given this observation, crime scene detection task ought to deal with both space and time based data.

1.2 Aims and Objectives

In this study we try to understand the ways to classify crime video footage by crime classes. For this, we are using 3D Convolutional Neural Networks (CNN) as they have demonstrated considerable success in the problem of action recognition

from video footage. We have provided background for upcoming discussions and we have reviewed the available relevant literature.

1.3 Thesis Outline

The given thesis is intented to follow the given logical order. First chapter is where we provided necessary background for the upcoming discussions. In particular, we go cover the topics of crime detection, artificial neural networks, convolutional neural networks (CNN), 3D convolutional neural networks (3D CNN). In chapter 2 we review relevant literature. We believe that our research should be built upon the earlier research. We cover the most relevant and influential papers in the given domain. In chapter 3, we share our results. In the final fourth chapter, we summarize the thesis and provide conclusions. At the end, we provide references.

Chapter 2

Background

2.1 Crime Detection

The later decades witnessed an immense growth in urban urban populaces, which has driven to the request for a secure, affable and feasible society. With the presence of ever-increasing urban development inundating rural areas and provincial regions, overseeing urbanization is and projected to be a significant issue for almost all government organizations. Major cities are getting to be overpopulated, constraining governments to embrace shrewd city activities that would offer assistance way better oversee framework and address major security, supportability and improvement challenges. Whereas keen city activities have picked up huge force with the guarantee of progressing quality of life, they too have their possess challenging angles. One of the greatest challenges in keen city life is the issue of security. Different think tanks have written about have been conducted to way better get it wrongdoing designs and their relationship to the socio-economic advancement of particular locales, human characteristics, their level of instruction and family ties [29].

Wrongdoing examination organizations have distinguished diverse sorts of violations. The four fundamental categories incorporate slaughtering, badgering, plundering, and strongly assaults. Murdering or kill alludes to the willful kill of one individual by another. Badgering is the sexual manhandle of a lady, man, or child against their will. This wrongdoing is as shocking as assault and has genuine results. Plunder alludes to the act of taking merchandise from a human domain utilizing over the top physical drive or drive. At long last, strongly assaults allude to the illicit encounter of one individual against another in arrange to attain something or essentially hurt the individual [51]. Wrongdoing discovery could be a need in urban life, and machine learning could be a well known strategy for wrongdoing location and anticipation. A few organizations around the world have tested with these strategies.

It has been consistently noticed that violations are regularly unsurprising, requiring as it were the handling of huge quantities of information that would reveal

curiously designs appropriate for law authorization. In numerous cases, violations committed regularly go unreported due to outside weights from all divisions of society. Cleverly frameworks can expeditiously distinguish wrongdoing and offer assistance dispose of such manipulative action by bypassing people and naturally informing the fitting specialists. For case, the study by Borges et al. [29] examined the case consider of San Francisco, USA, and Natal in Brazil, where criminal movement was broad. The different traits of urbanization in these two cities were analyzed and after that sophisticated machine learning models were actualized to distinguish hotspots of criminal action.

Agreeing to [51], they made a relapse demonstrate to anticipate wrongdoing rates in several Indian states. Directed and unsupervised learning procedures have too been utilized to realize moved forward precision in wrongdoing forecast. In [36], the fluffy implies calculation was utilized for wrongdoing information clustering for different recognizable wrongdoings, specifically seizing, kill, forceful theft, burglary and wrongdoings against ladies. Essentially, closest neighbor strategies have been utilized to screen wrongdoing rates, which have made a difference to get it wrongdoing sorts and time and place information of the underlying event.

Considering the different ponders that have been conducted, it is famous that most of the existing work emphasizes the utilize of criminal past history and populace density levels for wrongdoing forecast. However, one could look at other four property era strategies for identifying criminal offenses. The dataset contains different wrongdoing areas in an region where implies clustering is connected, coming about in wrongdoing hotspots. At that point a wrongdoing proportion network is made, which leads to the expectation of wrongdoing likelihood when subjected to a machine learning show. Beneath the proposed technique, wrongdoing reconnaissance is carried out utilizing the taking after strategies:

(i) It is possible to propose the Crime Dynamics Probability that is meant to compute the relation dynamics of one criminal offence to the some other criminal offence.

(ii) One could also propose that Vulnerability Index. It should tell the security level of the underlying area.

It ought to be famous that much existing work employments manufactured insights and machine learning to extricate wrongdoing designs and distinguish and anticipate criminal episodes. Most of the existing works have a few restrictions.

Wrongdoing may be a worldwide issue that emerges day by day and impacts society, negatively impacting society [9]. The continuous ever-increasing populace at the side expanding urbanization has driven to an immense increment in criminal action [18, 37], especially in urban settings [46, 43, 38]. Whereas expanded police nearness has been reported to diminish increments in wrongdoing [1], police must be able to arrange for and respond successfully to criminal occasions, especially those influencing personal or open security. The police and others within

the respectful arrange would advantage specifically from insights to assist battle all shapes of wrongdoing, and insights may be a prerequisite for overseeing and battling "smart" shapes of wrongdoing [28].

Data-based insights requires data or information and a technique for analyzing the information. Law authorization offices are an fabulous source of huge quantities of information. For illustration, the figures of accessible capture records from 2017 to 2019 for the city of Detroit is 81,442, 82,195 and 83,898 are [15], or nearly a quarter of a million records for the given 3-year period.

Manufactured insights and more precise machine learning (ML) give instruments to move forward police information of current and hypothetical wrongdoing and encourage extensive criminal approach choices. Machine learning procedures typically scale pretty decently to exceptionally expansive quantities of information [27]. Neural Network based systems (NNs) are a type of machine learning based on the usefulness of the human brain and its neurocognitive handling. NNs are the foremost effective and precise clustering innovation accessible [27], empowering high-quality data to bolster police intelligence capabilities.

2.2 Machine Learning

The typical objective of machine learning is to find a mapping between input patterns and output values. For example, we have object pictures as input data (represented by pixel intensity values) and proper labels as output values (one for each category of item). The algorithm's goal then becomes to learn this mapping (from samples to output value) in order to anticipate the proper output of a new input sample. In actuality, this straight mapping from input to output values is frequently a complicated and non-linear function that machine learning algorithms that only learn a linear mapping cannot uncover.

This is why researchers use hand-optimized characteristics to describe data so that the algorithm can learn the simplest mapping from these attributes to the output value. Unfortunately, this technique necessitates the creation of new hand-optimized features for each new activity, and the engineering process is complex and time-consuming.

In the case of this work, we also look at algorithms that attempt to learn feature representations from data on their own. In most cases, the so-called training set is utilized to fine-tune the machine learning model's parameters. This training set contains n input vectors $\{x1, x2, \ldots, xn\}$ as well as n target vectors if desired (one for each sample). The parameters of the model, which define the mapping from the input vector x_i to the output value $y(x_i)$, are adjusted during the training phase, also known as the learning phase. Following the training phase, the test set is used to determine the model's quality.

Generalization refers to a model's capacity to properly forecast the target value of fresh (unknown) samples. Generalization is critical in machine learning since

the training set often only includes a tiny portion of all potential input vectors. There are several approaches to dealing with this problem, which is discussed in the subsections below.

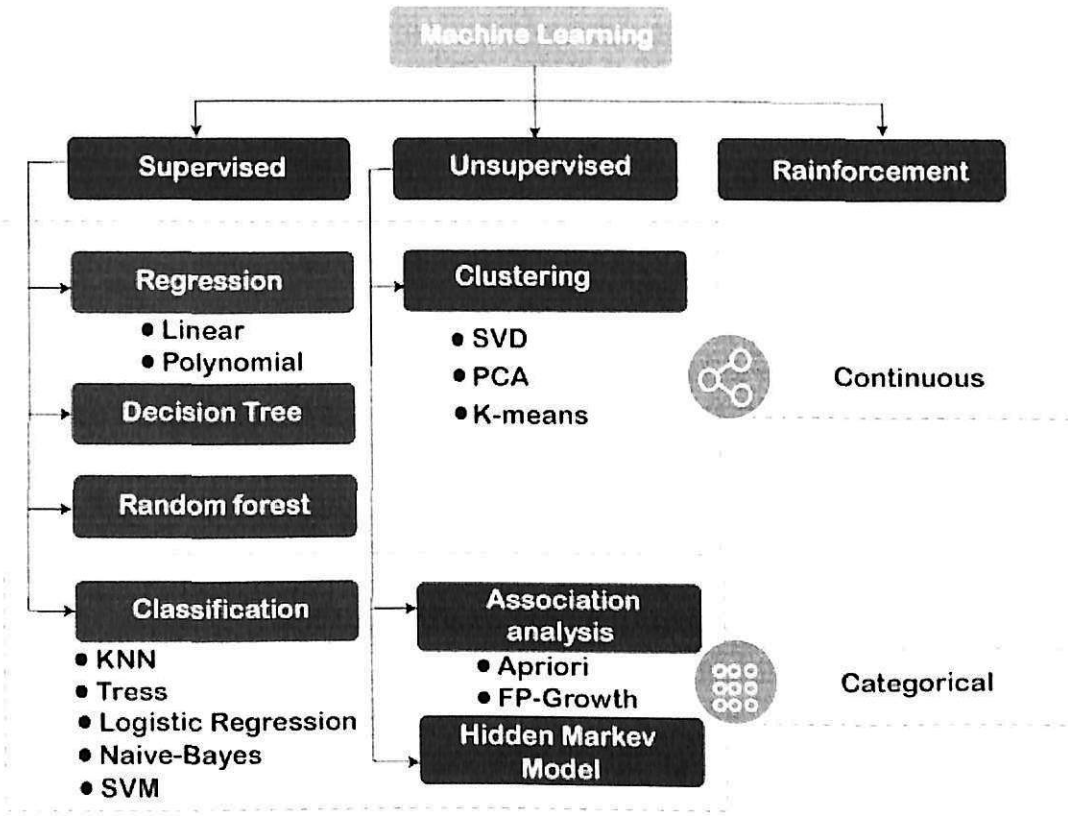


Figure 2.1: Machine Learning Settings

There are several machine learning settings, as seen in the diagram above and detailed further below. The algorithm type is determined by the properties of the data collection and the specific job, as explained below.

Supervised Learning

A classic machine learning issue is the supervised learning problem. The training data consists of tuples (x_i, y_i), where x_i represents the input vector and y_i represents the target vector. A classification problem is one in which the goal value is discrete, such as the digit identification issue, and the pictures are mapped to a finite number of discrete categories. The job is known as a regression issue when the goal value is continuous. The prediction of the price of a house (continuous output value), where the input variables are the number of rooms and the living area, is an example of a regression issue.

Learning Without Supervision

The training data in unsupervised learning issues consists of samples of the input vectors without any matching goal values. Clustering, density estimation, and visualization are some of the aims of unsupervised learning challenges. Clustering is a technique for identifying groupings of comparable items based on measurable

or perceived similarities between them. The goal of density estimation is to figure out how the data is distributed within the input space. The data is projected down from a high-dimensional realm to two or three dimensions for viewing.

Semi-supervised and Self-taught Learning

There are two techniques for supervised learning tasks that use unlabeled data and are therefore midway between supervised and unsupervised learning. These techniques are motivated by the prospect of having algorithms learn from unlabeled data as well as the difficulty of obtaining (enough) labeled data for supervised learning tasks.

The basic concept is to feed the algorithm a large quantity of unlabeled data in order for it to learn a decent feature representation of the input and then feed the algorithm labeled data in order for it to solve the supervised task using the learned feature representation. The training data set in a semi-supervised learning context may be separated into two parts: data samples $X_l = \{x_1, \ldots, x_l\}$ with associated labels $Y_l = \{y_1, \ldots, y_l\}$, and data samples $X_u = \{x_{l+1}, \ldots, x_{l+u}\}$ with unknown labels.

In contrast to the semi-supervised learning environment, the self-taught learning setting is more powerful since it does not presume that the unlabeled data Xu follows the same distribution (or class labels) as the labeled data Xl. We'll use an example to highlight the differences between the two approaches: the goal is to identify between photographs of cats and images of dogs.

Both systems rely on labeled pictures, with each example being either a dog or a cat image. This configuration is dubbed semi-supervised if there are unlabeled data examples that are all photographs of either a cat or a dog (but are not labeled). In self-taught learning, on the other hand, there is an unlabeled data set, which is made up of random pictures (perhaps acquired from the Internet) and so comes from a different distribution than the labeled data set.

Reinforcement Learning

The challenge of how to interact with the environment and pick appropriate actions in a particular scenario in order to maximize the reward is addressed by reinforcement learning. Unlike supervised learning, there are no examples with an optimal output; instead, the algorithm should learn which actions to take in a particular circumstance by interacting with the environment (the process of trial and error).

Hyper-Parameters

Selecting hyper-parameters is thus formally equal to selecting a model, i.e. selecting the best suited value/algorithm from a set of values/algorithms. Hyper-parameters can be either continuous (for example, learning rate) or discrete (for example, the number of neurons in one layer) and can be thought of as an external control button. We'll use the example of training a polynomial function to match a certain function to demonstrate the distinction between hyperparameters and parameters.

It is possible to look at the example of a polynomial function given by the

following equation

$$y(x, w) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

where M is the polynomial order and w_i are the polynomial coefficients $w = \{w_0, w_1, \ldots, w_M\}$. M must be chosen 'by hand' before the polynomial's training can begin. The polynomial coefficients w_i, on the other hand, are adjusted throughout the training operation. As a consequence, M may be thought of as a hyper-parameter, whereas the polynomial coefficients can be thought of as parameters. Hyper-parameters must be changed for all of the learning algorithms covered in this paper. The number of hyper-parameters might be rather high (ten or more) in some circumstances, such as neural networks, which further complicates the situation.

Choice of the Model

In terms of complexity and the quantity of free parameters that may be modified, different machine learning models differ. Aside from the model selection, the values for the model's parameters must be determined. More specifically, the key goal in this context is to set the free parameter values in such a way that the greatest prediction performance on fresh data is obtained.

Due to the problem of overfitting, the training set is not a reliable indication for evaluating the prediction performance of alternative values for the parameters. Instead, if enough data is available, the dataset can be separated into three sections. The initial part of the data and various hyper-parameter settings are used to train the model, and this fraction is referred to as the training set.

The second half of the data isn't utilized for training, thus it's ideal for comparing different hyperparameter values and choosing the one with the best-predicted performance. A validation set is a collection of independent data. The third set of data, referred to as the test set, is required to evaluate the selected model in terms of generalization. Because the validation set was used to choose the final model, it is no longer completely separate from the model (overfitting to the validation data can occur).

Cross-validation is another approach that divides the dataset into two parts: the training set and the test set. This method is particularly useful when the amount of data available is restricted and as much of it as feasible should be used for training. Cross-validation comes in a variety of forms, but the basic concept is the same in all of them.

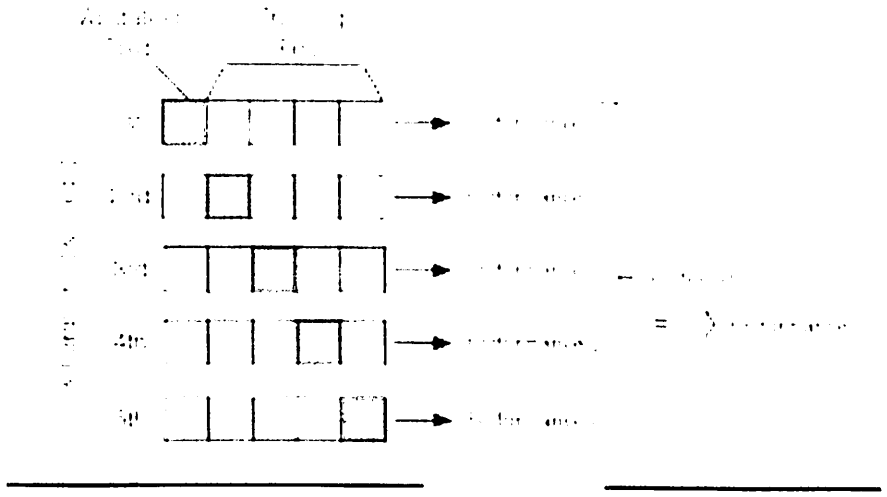


Figure 2.2: Cross Validation Example

A frequent kind is a k-fold cross-validation, which divides the training set into k groups at random. A collection of models has trained in $k-1$ groups and then assessed in the remaining group. This method is repeated for each group, ensuring that each group has been evaluated at least once. The aggregate performance scores of all runs are then used to pick a model. Finally, the test set is utilized to assess the chosen model's performance. Aside from k-fold cross-validation, another alternative is the leave-one-out strategy, in which k equals the total number of samples. Beside k-fold cross-validation, another specific variant is the leave-one-out-technique, where k is chosen to be equal to the total number of samples.

Cross-validation has the disadvantage of increasing the number of training runs (for a single setup) by a factor of k. Computational considerations may be considered in the process of picking hyper-parameters, in addition to the validation error, which is a proxy for the generalization error. Computing resources are limited and, for instance, have an impact on the choice of intervals of values to consider. Coordinate descent, multi-resolution search, grid search, and random search are some of the strategies for optimizing hyper-parameters. Only one hyper-parameter is changed at a time in the coordinate descent technique, and the change is always done from the best hyper-parameter configuration available at the moment.

The grid search is nothing more than an exhaustive search of all potential hyper-parameter combinations. The fundamental disadvantage of this method is that the number of configurations rises exponentially with the number of hyper-parameters (for example, 6 hyper-parameters, each of which may take 5 distinct values, yields $5^6 = 15625$ options). The grid search, on the other hand, is fully parallelizable, which is a benefit of this approach (the ability to run different combinations on different computers).

In contrast to grid search, the random search method involves changing all

parameters at the same time and sampling each hyper-parameter individually from a previous distribution (e.g. uniform distribution, inside the interval of interest). When the number of hyper-parameters exceeds two or three, random sampling can be more efficient than grid search. The qualitative examination of the results, on the other hand, is more difficult. The concept behind a multi-resolution search is to start with large-sized hyper-parameter steps. After a few 'best configurations' have been identified, one may begin by exploring more locally around them to fine-tune the setup (with smaller-sized steps).

Underfitting and Overfitting

Machine learning's goal, as previously stated, is to maximize the predicting accuracy of fresh data rather than training data. The predicted accuracy of fresh data must not be good if the model matches exactly with the training data. Consider the case where we want to fit the function $\sin(2x)$ with a polynomial of order M for a better understanding. To calculate the complexity of this linear model, the order M of the polynomial must be determined.

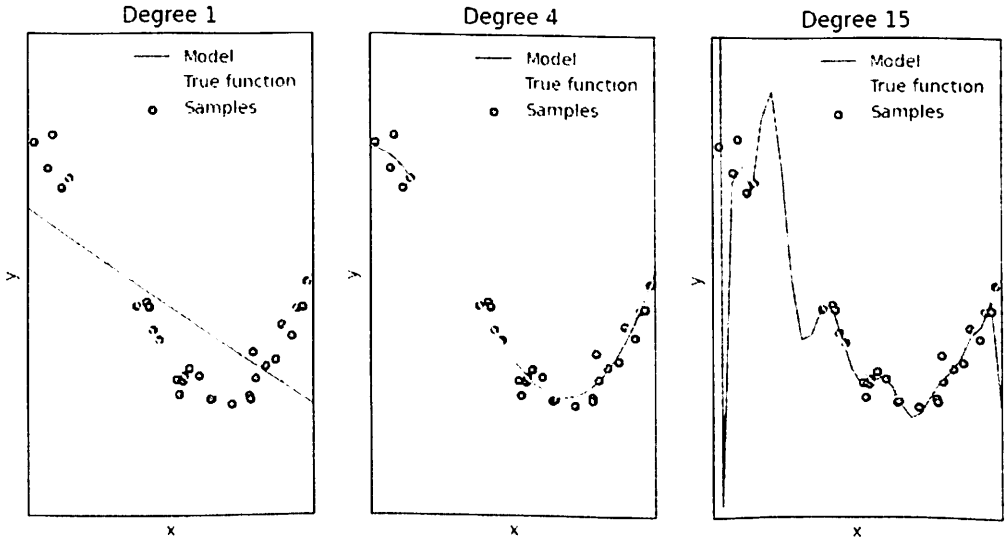


Figure 2.3: Underfitting and Overfitting Examples

The higher-order polynomial ($M = 4$) has a perfect fit to the training data, but oscillates wildly and poorly depicts the 'real' function, as illustrated in the figure. This is referred to as overfitting. Lower order polynomials, on the other hand, are too simple to match the underlying trends in the training data, and hence are poor representations of the 'real' function. This is referred to as underfitting.

Preprocessing's Importance

In machine learning, preprocessing is a usual step. Depending on the nature of input data, numerous alternative approaches, such as PCA and whitening, can be applied. This section explains the fundamental concepts of preprocessing and explains why preprocessing may help machine learning algorithms perform better.

Data Normalization

Computing the mean value (of all dimensions) of an example and removing that mean value from every dimension of that example is a simple preprocessing step in machine learning. This is referred to as brightness normalization in photographs. Another way to normalize data is to calculate the standard deviation of an example and divide it by that value. This has the property of contrast normalization in photos.

Dimensionality Reduction Using PCA

The principal component analysis approach can be used to reduce dimensionality. This can help minimize the time it takes for machine learning algorithms to execute, as the quantity of input characteristics might affect the time it takes. PCA's major goal is to create a more compact representation of the data based on the premise that the input variables are linked and hence redundant.

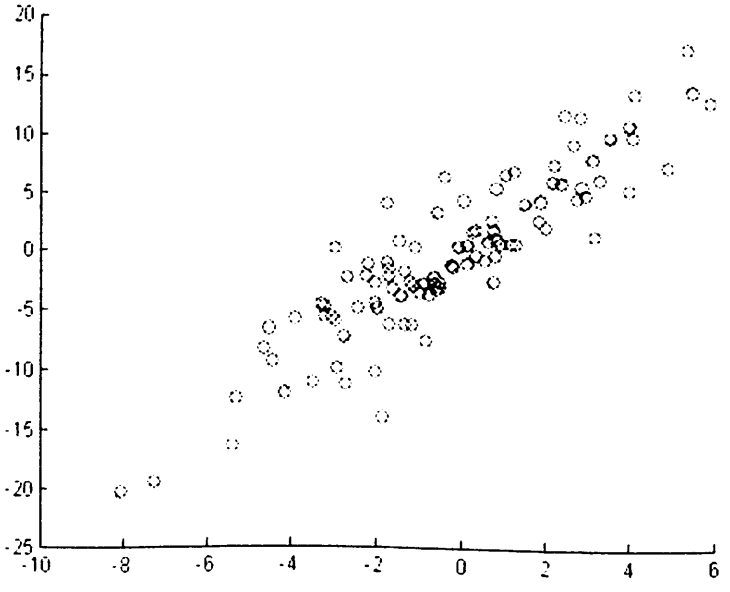


Figure 2.4: PCA Example

The goal is to use a lower-dimensional subspace to explain the original high-dimensional data as accurately as feasible. This may be seen geometrically as a linear projection of the original data example onto a (lower-dimensional) coordinate system, with the goal of retaining as much information as possible. Principal components are the major axes of this new coordinate system. The covariance matrix of the provided dataset is calculated in PCA to accomplish this. The statistical relationship between the pairs of variables is determined by the components of the covariance matrix. This covariance matrix's eigenvectors create an orthonormal basis.

The eigenvectors of this covariance matrix point in the direction of the data's maximum variation and are identical to the principal components. If e_1 and e_2

are the eigenvectors and x_i is an input-example, then under the new basis vectors their coordinates would be

$$\hat{x}_i = (e_1^T x_i, e_2^T x_i)$$

The data coefficients are uncorrelated with regard to this new basis, which is the desired attribute.

The variance of the data projected onto the associated eigenvectors is represented by the eigenvalues. The eigenvalue specifies how much of the overall variance is covered by the eigenvector that corresponds to it. The k components with the 11 greatest eigenvalues are kept, while the others are discarded, resulting in a dimensionality reduction from n to k dimensions. As a result, the resultant representation retains as much volatility as feasible. In actuality, the feature dimension is usually significantly higher, making the selection of k more difficult.

2.3 Neural Networks

2.3.1 Introduction

A neural network (NN) comprises of hubs (known as neurons) and coordinated edges between hubs, which are drawn below for the sake of example.

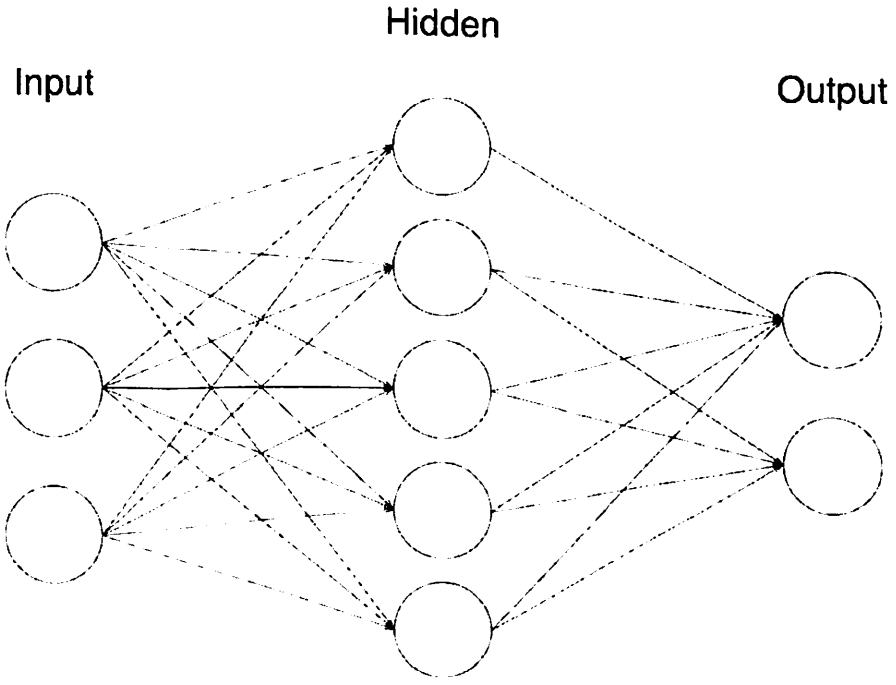


Figure 2.5: Typical Neural Network

Neurons execute a linear or non-linear functional transformation of a weighted sum of inputs values onto the real number space. The underlying result is at that point utilized as an input for subsequent neurons. Neurons are organized in several

layers, with the previous layers giving inputs to the subsequent layers. A typical neural network gets the starting information by means of an incoming layer. The data is flown through the neural network, with each layer acting on it until the costs are repaid within the final layer. In most cases neural networks are utilized to model regression-based and/or classification-based problems.

The inherent property of the particular problem decides the inputs, yields, and engineering of the arrange. For an illustration, in an picture classification assignment, the inputs are ordinarily the grayscale pixel quantities for each pixel, that takes a quantity between 0 and 255, and the yield of the modeling is usually a 1-of-K kind of classification. The architecture of the neural network and the mapping property of each neuron (alluded to as the enactment work) are as a rule characterized at the starting of the neural network development.

The neural network is at that point "instructed" how to process the incoming data by altering the weights of the associations (moreover called free parameters) to progress an mistake esteem computed from a predefined error functional form. Weight upgrades are educated by the slope of the error functional forms and scored employing a set of labeled "preparing cases". Within the following subsections, the neuron and the actuation work are talked about, in expansion to the regular actuation capacities for the covered up and yield neurons of the neural network. Much of the work and ideas displayed here are from [6]. When portraying the number of layers, the number of layers of the trainable weights is additionally numbered.

2.3.2 Function Based Form

The artificial neuron. An artificial neuron computes an yield quantity given a collection of incoming data. More particularly, a neuron comprises of an actuation work that works on a direct combination of the neuron's input values $x = (x_1, x_2, ldots, x_n)$, and weights $w = (w_1, w_2, ldots, w_n)$ related with the input values, and an discretionary predisposition term w_0. For case, the yield of a neuron within the covered up layer would be computed as

$$h = \sum w_i x_i$$

The weights are considered free parameters and are balanced in arrange to accurately classify the incoming data within the learning procedure. The direct combination of inputs to a neuron is commonly alluded to as a neuron's action. Enactment capacities are at that point connected to straight wholes. A number of enactment capacities that are utilized as cases are further discussed afterwards. The choice of enactment work utilized is affected by the sort of arrange and the learning calculation utilized.

Linear neuron, or $f(a) = a$. The straight neuron is frequently utilized as an yield unit in relapse issues, but not as a covered up unit, but as a stack of

direct combinations does not give the organize with extra separation. This can be since a straight combination of a direct work can itself be characterized as a direct work. Here is the graphical representation of a direct neuron.

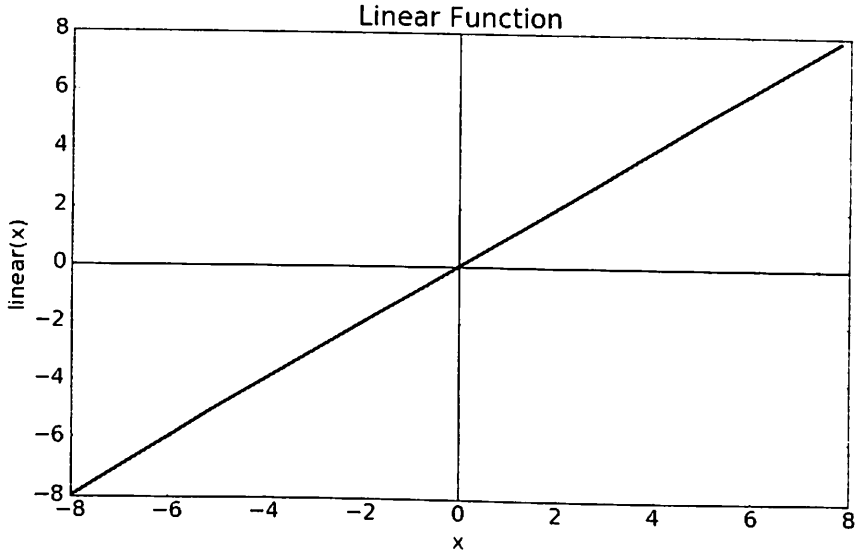


Figure 2.6: Linear Neuron

Sigmoidal neuron is given by the equation $f(a) = frac11 + e^{-a}$. It has generally been utilized to smooth the mapping to the |0,1| space, making it simpler to translate the yield as likelihood, which is valuable in a classification setting. Here is the graphical representation of a sigmoidal neuron.

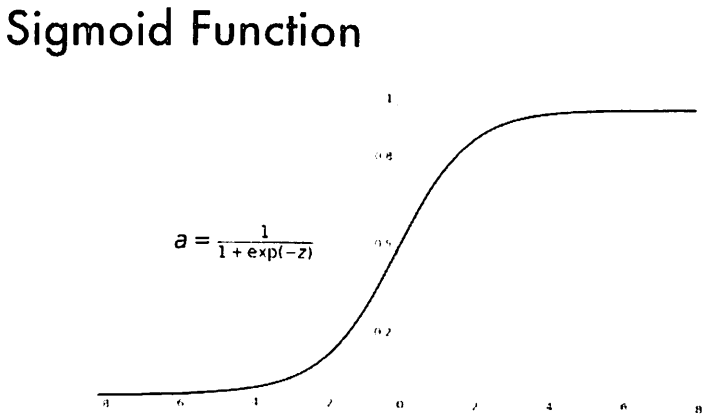


Figure 2.7: Sigmoidal neural

Hyperbolic Tangent and Soft Sign is given by the equation $f(a) = frac1 - e^{-2a}1 + e^{-2a}$. More as of late and as often as possible, the hyperbolic digression |50| and softsign |49| capacities are utilized, coming about in a so also molded work. The sigmoid unit has an asymptote at zero. When the input tends to zero, the angle of the sigmoid unit too tends to zero. This immersion can anticipate learning in going before layers |49|, given the hyperbolic digression and

delicate sign work are favored to the calculated sigmoid work. Here is the graphical representation of a hyperbolic digression and softsign actuation capacities.

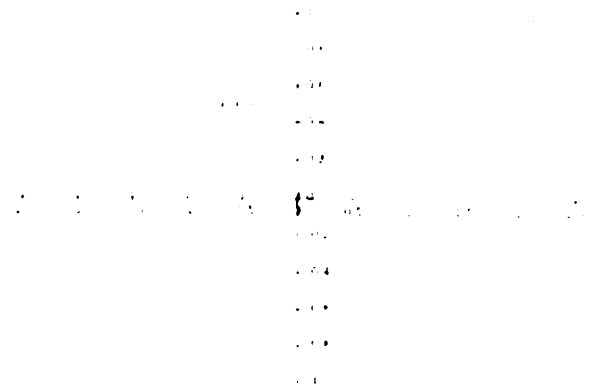


Figure 2.8: Hyperbolic Tangent

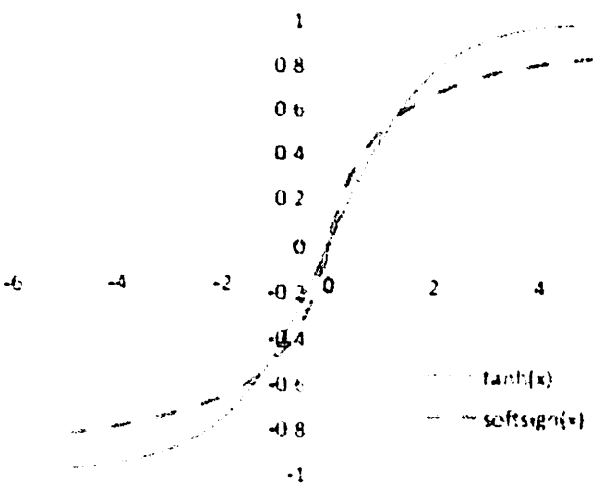


Figure 2.9: Softsign Activation

Rectified Straight Unit (ReLU) neuron is given by the equation $f(a) = a$ in case $a > 0$, and $f(a) = 0$ something else. As of late, the amended direct unit has experimentally accomplished triumphs [3]. Essentially, the subordinate is speedier to calculate and was too appeared to extend learning speed [3]. Here is the graphical representation of ReLU enactment work.

2.3.3 Parameter Learning

In arrange to adjust a neural network to an assignment, a strategy for modification of the free parameters must be embraced. A slope plunge adjusted to the NN setting is more often than not utilized. This requires a set of training data as well as error functional form. Given the labeled preparing information, the NN can be

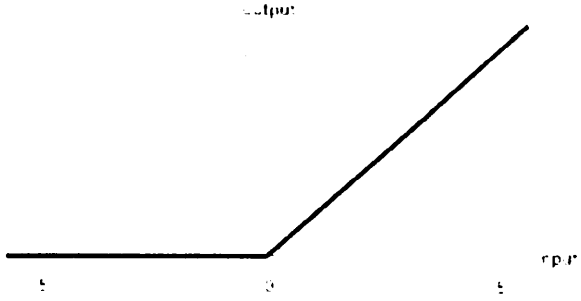


Figure 2.10: ReLU

prepared in a administered way by minimizing the mistake of the organize yields compared to the actual output classes. On the other hand, neural networks can be generally trained in unsupervised fashion with unlabeled information.

In spite of the fact that broad investigate has been conducted on unsupervised preparing [16], it is exterior the scope of this inquire about. Preparing administrations utilize labeled and unlabeled information in a semi-supervised way, or utilize labeled information from comparative datasets to pretrain the organize, expecting there are commonalities within the characterizing characteristics within the information [13]. These administrations are moreover not the subject of the think about. Different blunder capacities are examined within the taking after subsection. After that, slope plunge, i.e. gradient descent, together with backpropagation is the go-to the strategy for getting data almost weight upgrades in covered up layers.

Error Capacities. As specified over, in arrange to overhaul the parameters in an educated way, an error functional form has to be preset. In this subsection, the error funcitonal form is displayed employing a probabilistic elucidation of the output of the neural network within the context of maximizing the likelihood function [6]. Ordinarily, this process if referred to as mininization of the error functional form. The yield of the neural network gives hints on the error functional form in the operation.

In parameter optimization weights have to be modified so that the error functional form reaches a low point, or its minima. This problem is made much more difficult due to the non-convex error functional forms arising from the nonlinear relationships between variable weights and the underlying error functional form. In this case the functional form could have several local and global low points. The reason behind this could be due to the issue of weight symmetries. In turn, this leads to essentially different results. One should also note that we can't say for sure if the low point is a local or a global low point. Repeated weight renewals that use gradient-based data to slowly relocate the set of weights toward significantly lower error functional form value are commonly used in these circumstances.

In a general context of using neural networks, straightforward slope plunge (gradient descent) could be the leading strategy for error functional form mini-

mization. Essentially, gradient descent's basic idea includes slowly moving in a course inverse to the slope of the error functional form within the weight parameters' space utilizing tiny steps relative to the current error slope with regard to the weight. Weights in covered up layers are not specifically related to the mistake work, and as a result an usage of the chain run the show called backpropagation is utilized to assess their error functional form value. Within the figure underneath a sample error functional from is displayed.

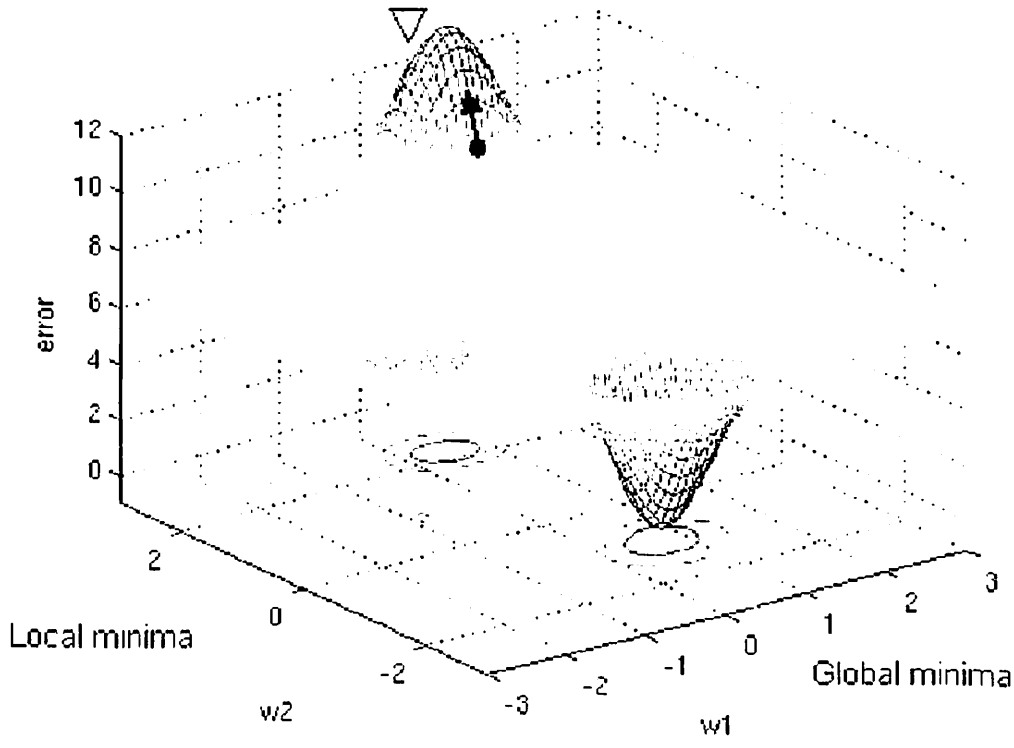


Figure 2.11: Sample Error Function

Gradient Descent. The gradient descent computation steadily upgrades the weight parameters by taking little steps within the negative course of the gradeint of the error functional form within the weight space. ie:

$$\Theta(t+1) = \Theta(t) + \Delta\Theta(t)$$

$$\Delta\Theta(t) = \alpha \nabla E(\Theta(t))$$

where $\nabla E(\Theta(t))$ is the gradeint of the error functional form in terms of W, α is called the learning rate and t is the weight overhaul record. Learning stops at a neighborhood least, i.e. $\nabla E(\Theta(t)) = 0$.

Since it isn't conceivable to decide whether a found minima is worldwide or neighborhood, different restarts with irregular beginning weights are commonly used to discover the finest weight combination. The learning parameter α may

be a predefined hyperparameter. Care ought to be taken when characterizing the learning rate because it influences the speed at which the organize focalizes. On the off chance that the learning parameter is characterized as exceptionally little, the organize will merge exceptionally gradually and may make preparing to merging illogical. Then again, in the event that the learning parameter is characterized to be exceptionally huge, progressive weight overhauls from negligible shells can sway in the event that they are around quadratic.

The learning rate may then again be versatile instead of steady. For example, if a organize contains a unique trainable parameter, at that point gradient descent is ideal when the error functional form surface may be a quadratic shell-like form. For a given set of preparing information, distinctive approaches can be taken as to the number of cases to assess some time recently overhauling the weights. Either full clump learning can be connected, where the normal mistake for the complete information set can be utilized to educate the another weight upgrade, or online learning can be connected, where each weight overhaul is educated by the assessed mistake subordinate inferred from a person irregular is calculated preparing case.

Online learning includes stochastics as the gradient is educated by an mistake surface evaluated from a unique arbitrary case. Stochastic gradient descent (SGD) permits the demonstrate to dodge nearby minima, whereas full clump learning merges to the nearby least closest to the introductory weights. In any case, due to the loud upgrades, SGD will not completely focalize whereas full group learning ought to. In this setting, learning rate treating plans are prescribed to decrease the variances of the stochastic slope plunge. In hone, it is common to apply mini-batch learning, where weights are overhauled concurring to the normal mistake angle of a irregular subset of the preparing cases. This decreases the commotion of online learning upgrades whereas still permitting the arrange to bypass neighborhood minima.

Moreover, within the setting of profoundly excess information, SGD quickens learning since a subset of cases can be agent of the fundamental components within the information and is quicker to compute between weight overhauls. As an extraordinary case of this, consider a preparing information set that contains numerous copies. In case gradient slope is found the middle value of some time recently weight upgrades, the copies include no extra data around a agent subset of cases [4]. The number of cases contained in a mini-stack could be a predefined hyperparameter. Ordinarily, it is equipment obliged instead of optimized based on the particular information through the utilize of a approval information set. Within the taking after chart there's a chart on how gradient descent works.

Backpropagation. A gradient descent is specifically appropriate to weights within the final layer of a organize, be that as it may approaching weights to covered up neurons are not specifically related to the error functional form term. For such instances, it is possible to use the chain rule. It is utilized to calculate the halfway value for corresponding gradients of the error functional form with respect to the weight. Within the setting of neural networks, this application of

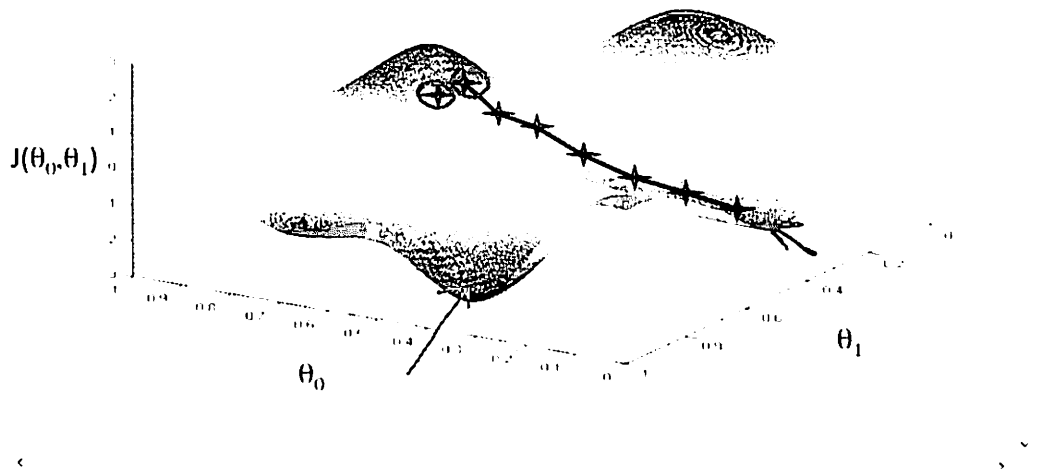


Figure 2.12: Sample Gradient Descent

the chain run the show is called back proliferation and can be depicted within the following steps.

1. Initially, one does a straight pass. At this point, one computes error functional form values and yields at each layer.

2. As a second step, a step-wise gradient computation is performed.

3. At last, the error derivation on computation for the case of hidden units is determined given the error functional form derivation on activities in the subsequent layer.

The following diagram shows the working principles of backpropagation.

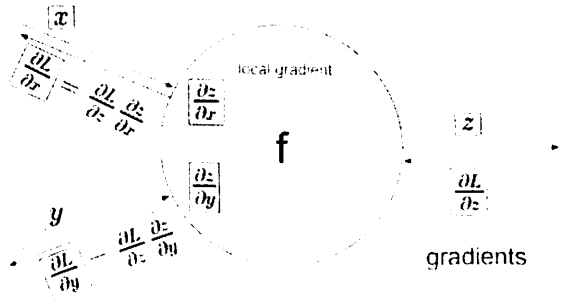


Figure 2.13: Backpropagation

2.4 Computer Vision

Computer vision is a branch of machine learning (ML) that allows computers and systems to extract useful information from digital photos, videos, and other visual inputs, as well as to conduct actions or make suggestions based on that data. If machine learning allows computers to think, computer vision allows them to perceive, observe, and interpret the world around them. Human vision is similar to computer vision, with the exception that people have a head start. Human vision benefits from lifetimes of context to learn how to distinguish objects apart, how far away they are if they are moving, and whether something is incorrect with a picture. Computer vision teaches computers to execute these tasks, but using cameras, data, and algorithms rather than retinas, optic nerves, and the visual brain, it must do it in a fraction of the time. Because a system trained to check items or monitor a production asset may evaluate hundreds of products or processes per minute, detecting faults or issues that are invisible to humans, it can swiftly outperform humans. A lot of data is required for computer vision. It repeats data analysis until it detects distinctions and, eventually, identifies pictures. To teach a computer to recognize automotive tires, for example, it must be fed a large number of tire photos and tire-related materials in order for it to understand the distinctions and recognize a tire, particularly one with no faults. Machine learning is a technique that allows a computer to train itself about the context of visual input using algorithmic models. If enough data is supplied into the model, the computer will "look" at the data and learn to distinguish between images. Instead of someone training the computer to recognize an image, algorithms allow it to learn on its own.

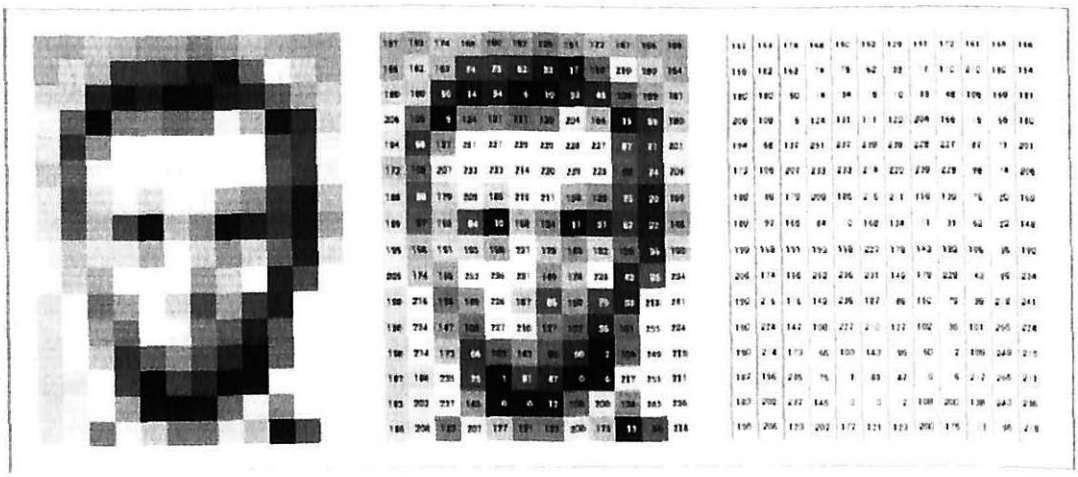


Figure 2.14: Breaking Picture into Pixels

By breaking pictures down into pixels that are given tags or labels, a CNN aids a machine learning or deep learning model in "seeing." It creates predictions about what it's "seeing" by using the labels to do convolutions (a mathematical operation on two functions to produce a third function). In a series of iterations,

the neural network executes convolutions and assesses the accuracy of its predictions until the predictions start to come true. It then recognizes or sees pictures in a human-like manner. A CNN, like a person recognizing a picture from a distance, detects hard edges and simple forms initially, then fills in the details as it runs iterations of its predictions. To comprehend single pictures, a CNN is employed. In video applications, a recurrent neural network (RNN) is used in a similar way to assist computers to grasp how visuals in a sequence of frames are connected to each other.

The development of computer vision

For almost 60 years, scientists and engineers have been attempting to find ways for robots to comprehend and analyze visual input. In 1959, neurophysiologists presented a cat a series of pictures in order to see whether it could correlate a reaction in its brain. They noticed that it responded to hard edges or lines initially, implying that picture processing begins with basic forms such as straight edges.

The first computer image scanning technology was created about the same time, allowing computers to digitize and capture images. In 1963, computers were able to convert two-dimensional pictures into three-dimensional shapes, marking yet another milestone. AI became an academic topic of research in the 1960s, and it was also the start of AI's attempt to address the human vision issue. Optical character recognition (OCR) technology was introduced in 1974, and it could recognize text printed in any font or typeface. Similarly, neural networks might be used to decode handwritten writing using intelligent character recognition (ICR). OCR and ICR have since made their way into document and invoice processing, car plate recognition, mobile payments, machine translation, and a variety of other applications.

David Marr, a neurologist, discovered that vision is hierarchical in 1982 and developed methods enabling robots to recognize edges, corners, curves, and other fundamental structures. Kunihiko Fukushima, a computer scientist, devised a network of cells that could identify patterns at the same time. Convolutional layers in a neural network were used in the Neocognitron network. The study's focus shifted to object identification by 2000, and the first real-time facial recognition applications debuted in 2001. Throughout the 2000s, standardization of how visual data sets are labeled and annotated arose. The ImageNet data collection was released in 2010. It comprised millions of annotated photos from a thousand different object classes and served as the basis for today's CNNs and deep learning models. In 2012, a team from the University of Toronto competed in an image recognition competition with a CNN. The model, dubbed AlexNet, drastically lowered picture identification error rates. Error rates have dropped to only a few percent since this breakthrough.

Image Processing

Computer vision preprocesses pictures and transforms them into more relevant data for future analysis using image processing principles or techniques. In most

computer vision systems, image processing is the initial step. The majority of computer vision applications rely on image processing methods.

Using a Gaussian Filter to Blur

The most popular method for blurring or smoothing an image is this. As pixels travel away from the center, this filter enhances the resultant pixel and gradually reduces the impacts. This filter may also be used to reduce picture noise. For example, on a shot with a hard-edged item, the box filter does not provide a smooth blur, however, the Gaussian filter can address this problem by smoothing the edges surrounding the object.

Non-linear filtering

Linear filtering is simple to use and put into practice. In certain circumstances, this procedure is sufficient to provide the desired result. Non-linear filtering, on the other hand, can be used to improve performance. When confronted with a more complicated computer vision problem, we may gain greater control and produce better outcomes by using non-linear filtering.

Median Filtering

A non-linear filtering technique such as the median filter is an example. This method is widely used to reduce the amount of noise in an image. It works by pixel-by-pixel examining the image and replacing each pixel's value with the value of the nearby pixel median.

Detecting and matching features

When it comes to computer vision ideas, feature detection and matching are two of the most important strategies in many applications. Recognition, 3D reconstruction, structure-from-motion, image retrieval, object identification, and many more tasks are included in this approach. Detection, description, and matching are the three activities that are commonly grouped into this approach. In the detection task, points in each image that are simple to notice or match are detected. The aspect that surrounds each feature point is described in the description task in such a manner that it remains intact during actions such as lighting, translation, scaling, and in-plane rotation. Descriptors are compared across photos in matching for comparable characteristics to be categorized. The following are some ways for identifying and matching features:

- Lucas-Kanade
- Harris
- Shi-Tomasi
- SUSAN (smallest univalue segment assimilating nucleus)
- MSER (maximally stable extremal regions)
- SIFT (scale invariant feature transform)

- HOG (histogram of oriented gradients)
- FAST (features from accelerated segment test)
- SURF (speeded-up robust features)

Let's look at some of the most common feature detection and matching methods.

Scale Invariant Feature Transform (SIFT)

Feature Transform with Scale Invariance (SIFT) Even if an object is resized, SIFT overcomes the difficulty of recognizing its corners. Implementation steps for this method are as follows: Scale-space extrema detection -- This stage will find the locations and scales of the same object in a picture that can still be identified from multiple perspectives or viewpoints. Localization of important points - Once key points have been discovered, they will be improved to provide precise findings. This would result in the deletion of low-contrast spots or points with poorly localized edges. In this stage, a consistent orientation is assigned to each key point in order to achieve invariance when the picture is rotated. Keypoint matching - The key points between pictures are now connected to recognizing their closest neighbors in this stage. The output, as seen in the figure below, displays where the inputted image's important spots are placed.

SURF

SURF (Sped-Up Robust Features) was presented as a faster variant of SIFT. SIFT is a slow technique that can recognize and characterize important points of an object in an image. The white blobs of the Full-Scale logo are detected using SURF, which functions like a blob detector. This approach can aid in the detection of picture imperfections. FAST oriented and rotated BRIEF (ORB) This method is a viable alternative to SIFT and SURF, owing to its superior calculation and matching capabilities. It's a hybrid of a quick keypoint detector and a short description with several tweaks to boost speed. It's also a wonderful cost-effective option because the SIFT and SURF algorithms are patented, which means you'll have to pay them to utilize them. Segmentation In computer vision, segmentation is the process of extracting pixels in an image that is related. Segmentation algorithms usually take an image and produce a group of contours (the boundary of an object that has well-defined edges in an image) or a mask where a set of related pixels are assigned to a unique color value to identify it.

Image segmentation algorithms that are widely used:

- Active contours
- Level sets
- Graph-based merging
- Mean Shift

- Texture and intervening contour-based normalized cuts

Semantic Segmentation

Semantic segmentation is the process of assigning a class or label to each pixel in a picture. We can know what a pixel's class is by merely glancing at its color in semantic segmentation, but one drawback is that we can't tell if two colored masks belong to the same item.

Segmentation of Instances

The only thing that counts in semantic segmentation is the pixel's class. This would cause a difficulty in that we wouldn't be able to tell if that class belonged to the same object or not. Semantic segmentation can't tell if two things in a picture are one and the same. In order to address this issue, instance segmentation was developed. This segmentation may distinguish between two items belonging to the same class. If a picture contains two sheep, for example, the sheep will be recognized and masked with different colors to distinguish which class they belong to.

Panoptic Segmentation

Semantic and instance segmentation are combined in panoptic segmentation. Every pixel is categorised by a specific class in panoptic segmentation, and pixels with multiple occurrences of a class are also identified. If a picture has two vehicles, for example, the cars will be disguised with different colors. These colors all indicate the same class — automobile — yet they represent various examples of that class.

Recognition

One of the most difficult topics in computer vision is recognition. Why is it so difficult to get recognized? Recognizing an object's traits or properties would be simple for human sight. Multiple items can be recognized by humans with little effort. This does not apply to a machine, though. Because items vary, it would be difficult for a machine to distinguish or detect them. They differ in terms of perspectives, sizes, and scales. Though most computer vision systems still confront similar issues, they are making progress or developing new techniques to tackle these difficult jobs.

Object Recognition

Object identification is a technique for identifying a specific object in a photograph or video. This is the result of deep learning and machine learning algorithms. Object identification attempts to replicate this intrinsic human capacity to recognize certain characteristics or visual details in a picture. Deep learning may be used to recognize objects either by training models or by using pre-trained deep learning models. The first step in building a model from scratch is to gather a large number of datasets. Then you must create a specific architecture that will be utilized to build the model. Deep learning for object identification can yield thorough and precise results, but it's a time-consuming process that necessitates the collection of a vast amount of data. Object identification using machine learning, like

deep learning, has a range of methodologies. The following are some examples of common machine learning techniques:

- HOG feature extraction
- Bag of words model
- Viola-Jones algorithm

Object Detection
The capacity of machines to determine the position of an item in an image or video is referred to as object detection in computer vision. Object detection methods are used by several firms in their systems. It's used for facial recognition, online photos, and security.

Figure 2.15: Object Detection Example

What's the difference between recognition and detection of objects? Object recognition is the process of producing an image, whereas object detection is the process of determining where an object in an image is located. Object detection classifies an object's class based on its characteristics. When searching for circles in a picture, for example, the system will find any round item. This method employs learning techniques and picture attributes to recognize any occurrences of an object in a class.

2.5 Covolutional Neural Networks

A Convolutional Neural Architecture (ConvNet/CNN) may be a profound learning computation system that can take an input picture, dole out significance

(learnable weights and inclinations) to different aspects objects within the picture, and recognize one from another. The amount of initial processing required in CNN is much limited when we try to compare it to other deep learning based computation systems. Whereas primitive strategies include developing filters by hand, with adequate preparing, CNNs have the capacity to generate efficient filters by their own.

The design of a CNN is closely resembling to the network design of neurons within the human brain and was motivated by the organization of the human visual cortex. Each neuron as it were react to jolts in a constrained zone of the visual field known as the open field. A collection of such fields combine in order to account for the complete visual area.

In cases of greatly basic double pictures, the strategy might appear an normal exactness score whereas performing course forecast, but it would have small to no precision when managing with complex pictures that have pixel conditions throughout.

A CNN is able to effectively absorb the spatial and timely conditions in an picture by applying pertinent channels. The engineering adjusts superior to the picture information set since the number of parameters included is diminished and the trainable weights could be used over and over again. Putting in other words, the arrange can be prepared to superior get it the advancement of the image.

Input Picture. One can envision how computationally seriously things would get once the pictures come to measurements, e.g. 8K (7680×4320). The part of the CNN is to put the pictures into a shape that's less demanding to handle without losing highlights that are vital for a good forecast. Typically critical on the off chance that we are to plan an engineering that's not as it were great at learning highlights, but too spreads to colossal datasets.

Convolutional Layer - The Kernel. The overall objective of the convolution operation is to extricate the higher-level highlights, such as lines and circles, from the input picture. ConvNets do not got to be constrained to fair one convolutional layer. Customarily, the primary CNN is capable for capturing the low-level highlights such as lines, circles, shapes, color, angle arrangement, and so on. As more layers are included, the design too adjusts to the high-level highlights, giving us a organize that has the helpful understanding of pictures within the underlying data similar to what humans are capable of.

There are two sorts of comes about for the operation - one in which the size complexity of the convolved highlight is diminished compared to the input, and the other in which the size complexity is expanded or keeps intact. One could observe it is often done by applying substantial cushioning within the case of the previous, or same cushioning within the case of the latter.

Same cushioning: A 6x6x1 picture is cushioned with zeros to form a 7x7x1 image If we extend the 6 x 6 x 1 picture into a 7 x 7 x 1 picture and after that apply the 2 x 2 x 1 bit to it, we discover that the convolved network has measurements 6 x 6 x 1. Consequently the title - same Cushioning. Here is the graph of applying

convolution parts within the input picture.

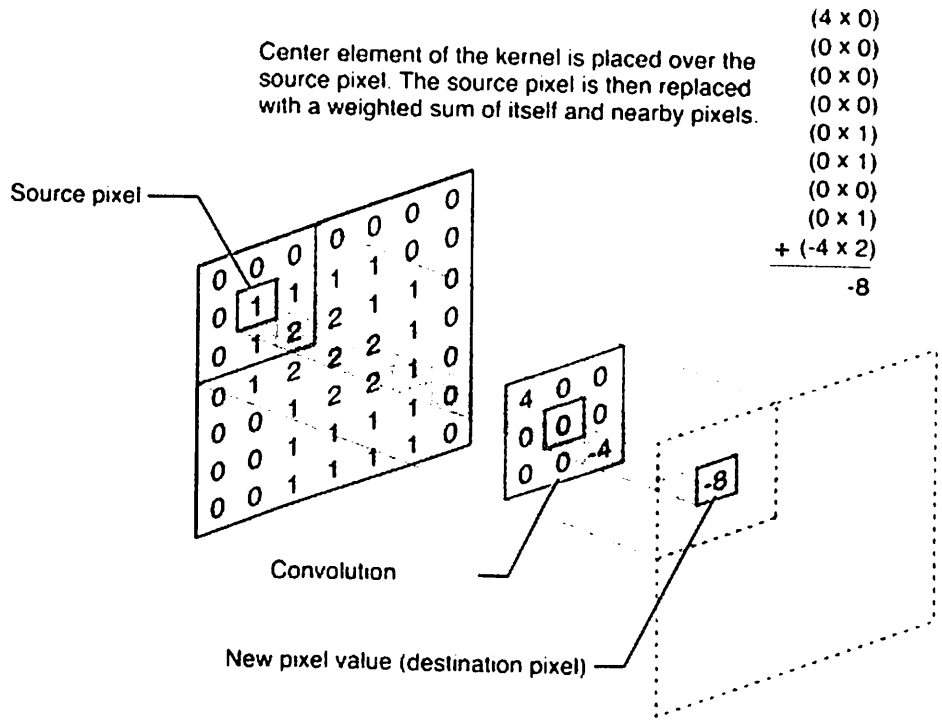


Figure 2.16: Convolution in Action

Pooling. Comparable to the convolutional layer, the pooling layer is mindful for diminishing the spatial estimate of the convolved include. This serves to diminish the computing control required to handle the information through measurement lessening. In expansion, it is valuable to extricate prevailing highlights that are rotationally and positionally invariant, subsequently keeping up the method of viably preparing the model.

One could observe two sorts of pooling: Max Pooling and Mean Pooling. When it comes to Max Pooling, it yields the greatest esteem from the portion of the picture secured by the part. On the other hand, Mean Pooling returns the mean of all values from the portion of the picture secured by the part.

Max pooling moreover works as a commotion canceling system. It disposes of the loud actuations inside and out additionally performs denoising beside measurement lessening. On the other hand, Mean Pooling essentially performs dimensionality decrease as a commotion lessening component. Subsequently, one could conclude that that Max Pooling tends to work somewhat better than Mean Pooling.

Types of pooling. The concatenation of convolutional and pooling layer shape the k-th layer of a CNN. Depending on the complexity of the pictures, the number of such layers can be expanded to capture indeed more low-level points of interest, but at the cost of more computational control. Here is the chart of a pooling layer.

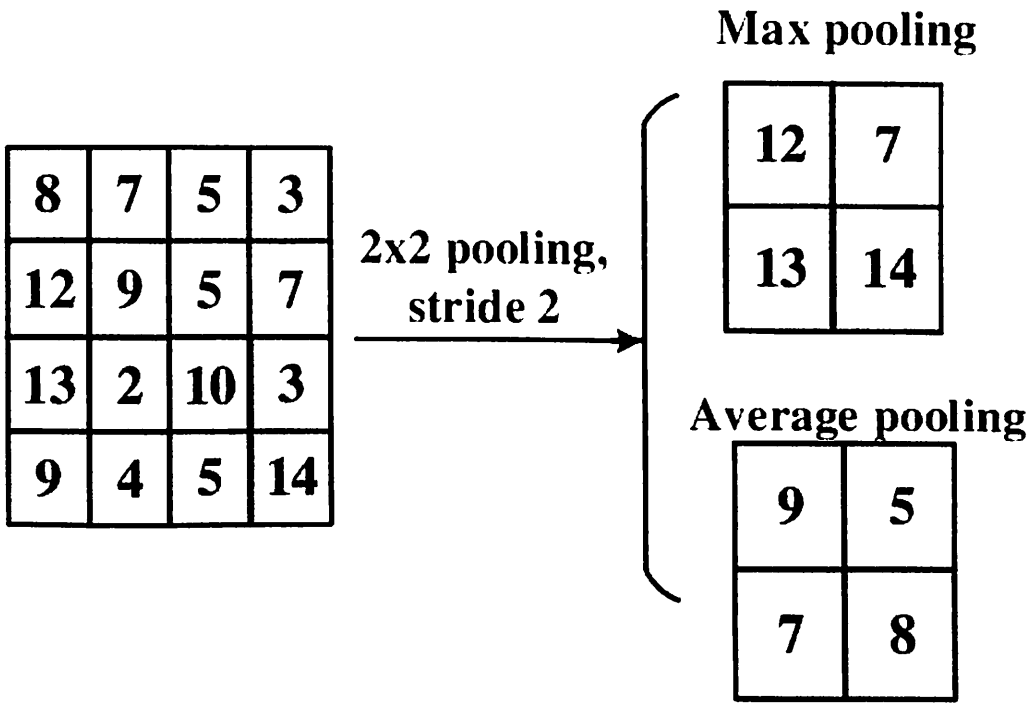


Figure 2.17: Pooling in Action

Classification – Completely Associated Layer (FC Layer). Including a completely associated layer may be a (as a rule) cheap way to memorize nonlinear combinations of the high-level highlights as produced by the convolutional layer yield. The completely associated layer learns a conceivably non-linear work in this domain.

After changing over our input picture into a appropriate shape for our multi-stage perceptron, we'll smooth the picture into a column vector. The smoothed yield is encouraged into a feed-forward neural arrange, and backpropagation is connected to each emphasis of the preparing. For an extend of epochs, the demonstrate is able to recognize between overwhelming and certain subordinate highlights in pictures and perform classification of them utilizing the softmax classification procedure. Here is the chart of completely associated layer and softmax layer.

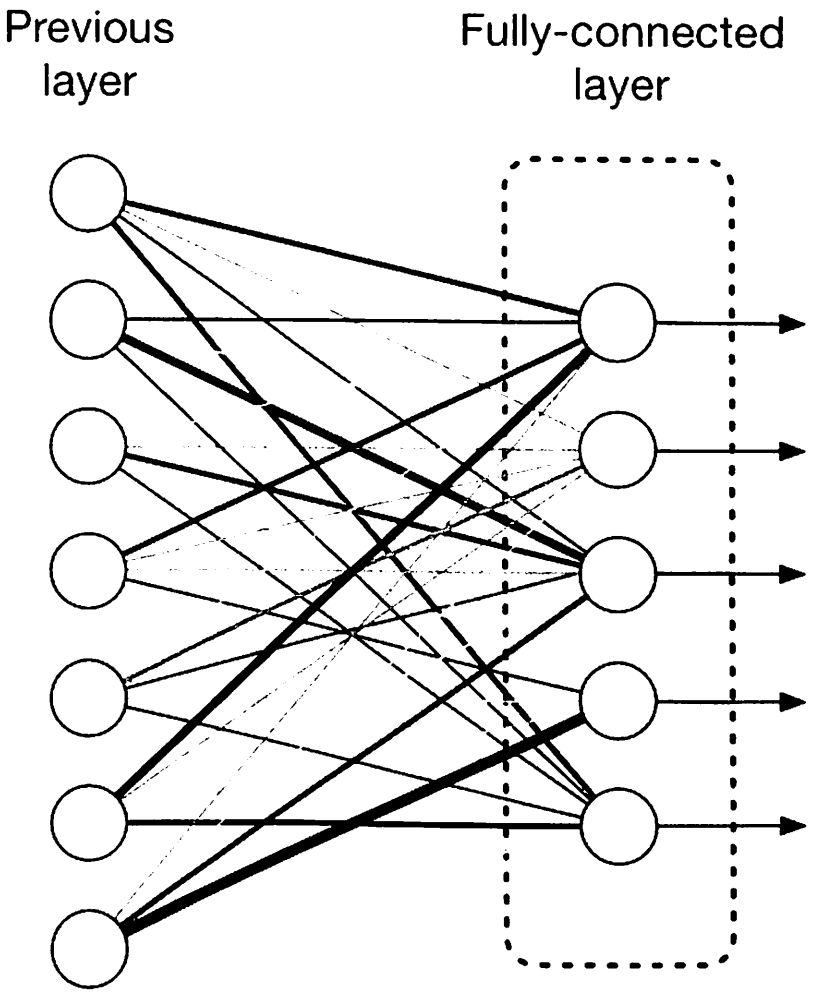


Figure 2.18: Fully Connected Layer

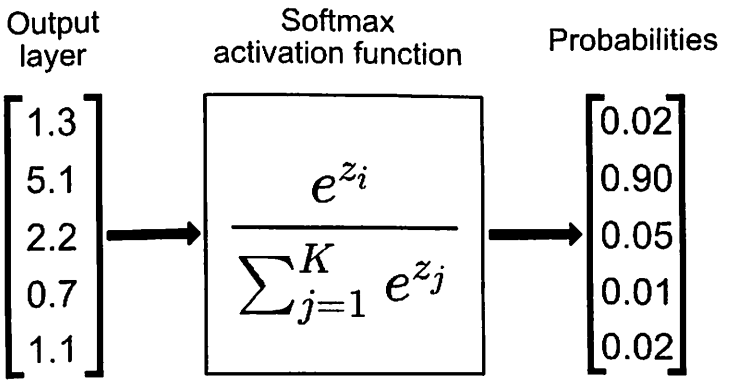


Figure 2.19: Softmax Layer

There are different structures of ConvNets that played a key part in building computations that control and will control AI as a entire for the predictable future. A few of them are recorded underneath:

- LeNet

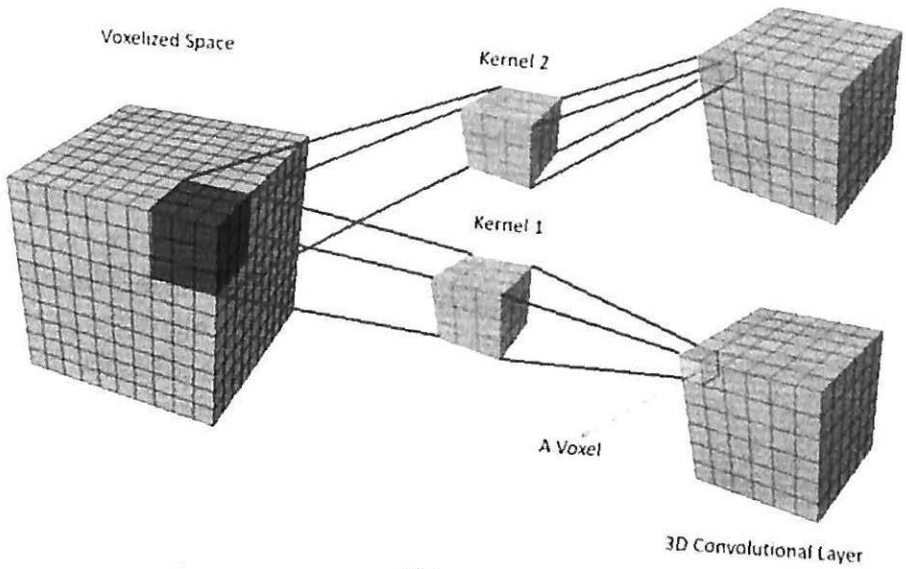
- AlexNet
- VGGNet
- GoogLeNet
- ResNet
- ZFNet

2.6 3D Convolutional Neural Networks

3-dimensional CNN (3D CNNs) to begin with were proposed by [35] in order to amplify the two-dimensional (2D) picture classification CNNs to video classification basically by including an extra transient measurement. Video information is anticipated to make strides the classification of activities as a few recognizing highlights develop from the movement. Whereas great precision can be accomplished with still outlines, expanding the number of outlines in a clip when classifying an activity has been found to progress the precision of the underlying classifier, but with quickly declining returns [32, 42].

The proximity between video-based and image-based data sources provides the motivation behind the limitations imposed on the architecture for regular 2D CNNs that apply to 3D CNNs. In particular, features in both spatial and temporal dimensions are seen to be localized to nearby located regions, and variances in video data are seen to include the ones present in image-based data. To add a note, the addition of the time-based complexity size might significantly increase the complexity of the input. If so, the usage of classical (2D) convolutions to network architecture could be continued. As with classical CNNs (2D CNN), the step of downsampling is rather encourages to diminish the complexity of the incoming data enough so that the underlying neural network keeps being learnable.

When it comes to video-based classification, it is ordinarily performed with specialized include extractors that are strong to varieties within the information and a regular classification strategy [12]. There's a wide run of diverse include extractors and classification methods, and distinctive strategies may be applicable depending on the video information. Within the same way as with picture acknowledgment, CNNs have an advantage over them since they perform include extraction and classification at the same time and don't require critical alteration to be connected to unused information. Be that as it may, there has been constrained advancement of 3D CNNs [34, 42, 47] and they are regularly extricated by the previously mentioned specialized include extractor in combination with a standard classification method outperformed. Within the following graph we display a sample 3D CNN kernel.



(b)

Figure 2.20: 3D CNN

textbfConvolution and Subsampling. The responsive field domain of each convolutional neuron is characterized over two spatial measurements and one transient measurement. On the other hand, the parameterization is the same as for 2D CNN's Convolutional Neuron. Downsampling between convolutional layers is as a rule as it were actualized over the spatial measurements, since the time-based measurement is regularly little sufficient that convolution over it diminishes its estimate adequately [42].

Example of a 3D CNN Engineering The work of [42] created a 3D-CNN for classifying human activities in recordings. The arrange was connected to the TRECVID [17] and KTH [8] datasets. The previous comprises of captioned security film of London's Gatwick Air terminal, comprising of numerous individuals frequently performing activities at the same time. The KTH dataset comprises of postured recordings of people performing activities against a homogeneous foundation. The 3D CNN inspired model outmatches the current video classification strategies as well as a 2D CNN when connected to the TRECVID information and conveys a decent error rate on the given KTH based dataset.

The distinctive channels appear in totally distinct colors, with outlines appeared as squares. One should remember that the quantity of squares fails to reflect the genuine quantity within the channel. Each component demonstrates a convolution, pooling, and yield layer. The width, tallness, and time measurements of a layer's include outline yield are included underneath the layer. By [42] tended to the TRECVID dataset, be that as it may, given the pertinence of the work done on the KTH dataset, the engineering of the 3D-CNN connected to the KTH dataset is mentioned here. The design utilized for each dataset was

comparative, with alterations made to the part measure to account for the diverse dimensionality the information.

The design in [42] comprised of one fixed component and six variable components. The fixed layer part isthen split into five distinctive channels concurring to the distinctive characteristics from the earlier information that [42] found that it progresses execution. The given five components included a flat and vertical slope, a level and vertical optical stream, and the first video as is. The dynamic component is comprised of two convolutional and pooling components, taken after by a convolutional component working precisely like a completely associated layer due to the measure of the getting field and the underlying incoming data, and an yield component that employs a softmax yield component.

Within the to begin with convolution layer, bits measured $9 \times 7 \times 3$ in terms of width, stature, and time were convolved onto the $80 \times 60 \times 9$ input. Two highlight maps were prepared for each channel, yielding a add up to of 1,900 dynamic parameters. The consequent pooling component connected a common dynamic parameter and inclination to the basic normal of non-overlapping 3×3 neighborhood spatial locales. This provided an opportunity for dynamic parameters to differ distinctive outlines, coming about in 132 trainable parameters. The kernel size of the given convolution layer was $8 \times 8 \times 3$, and three include maps were prepared per include from the wage layer, coming about in 5,340 dynamic parameters.

When it comes to the second pooling component, it utilized the same structure as the primary component of pooling but as a result of the expanded property maps had 226 dynamic parameters. Finally, the third convolutional component works in a similar fashion to a completely associated layer with 148 neurons. This is clear since its part, connected as it were to the spatial measurements, was 5×4, which is the same estimate as the input maps, also each neuron is connected to each highlight map and worldly outline. At the end, this comes about in 491,400 dynamic parameters. The final layer could be a softmax layer that's completely associated to the past layer and has 738 variable parameters. For every computational unit, the hyperbolic tangent unit is utilized as the enactment work and the learning rate is balanced utilizing the SDLM.

3D CNNs, in essence, are a normal expansion of 2D CNNs, and whereas 2D CNNs are well recognized in picture classification, 3D CNNs have not been broadly utilized for video classification. This is often likely due in portion to the littler video dataset sizes [42] and the higher dimensionality of recordings, making CNNs less viable compared to more productive, but expert-coded, express highlight extraction and classification models. When used with respect to the KTH dataset, the [42] accomplished an average error rate of 9.8% with a 5-fold sum of information. The result, whereas not prevalent to current video classification procedures, yet it is decent.

When it comes to the TRECVID, [42] connected two expansions to the fundamental demonstrate portrayed over. To begin with, to control the arrange, [42]

prepared the arrange on assistant yields computed from a arrangement of outlines that were bigger but contained the input frames. Auxiliary yields were calculated employing a set of word highlights from a Filter highlight extricated from the crude and moving edge history pictures.

To decrease the amount of the input, the KTH data was pre-processed to generate a compact bounding box having the underlying subject. In addition, after each of the first two layers of convolution, the architecture featured a correction layer that returned the absolute value of the input. [47], on the other hand, shows how to execute unsupervised pretraining on a 3D convolutional network structure using the Independent Subspace Algorithm (ISA). The authors attained a 6.1% error rate using the KTH dataset. Because it did not require localization as a pre-processing component like [34], which is an issue in and of itself. As for the [41] utilized to make initial design judgments. Furthermore, the architecture had outperformed its immediate competition.

The findings were shown to be improved by the regularization component, though not dramatically. Several models, each with its own architecture, were united into one ensemble in the second expansion. The outcomes were greatly enhanced as a result of this extension. Furthermore, the 3D CNN outperformed a 2D CNN, implying that key distinguishing traits were discovered across the time dimension.

There are various more 3D CNNs and models that are similar to 3D CNNs that have been applied to the KTH dataset. [34] study employed the yield of a short-sequence 3D CNN as input to a recurrent neural network (RNN), which are neural networks that have been tuned to work with sequential data. [34] obtained an average error rate of 8.97% over 4 epochs the data without RNN. The RNN architecture consisted of two components of convolution and pooling, a convolution component, a fully connected component, and finally an yield component. The error rate was lowered to 5.6% by applying an RNN to the set of yields of the 3D CNNs for the set of sequences that make up a movie.

2.7 Video Processing

It is preferable to use 3D convolution in the convolutional layers of CNNs to efficiently combine motion data in video analysis so that differentiating characteristics are collected along both spatial and transient measures. A variety of 3D CNN architectures may be developed to evaluate video data using the suggested 3D convolution.

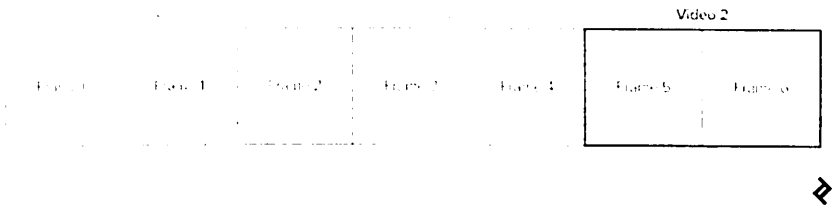


Figure 2.21: Video and Frames

The modeling of temporal information is required for the shift from image to video comprehension. This complexity size, which cannot be viewed as a spatial dimension, is critical for extracting information like motion that can't be predicted from static pictures. Predicting the activity in a video is a typical and intriguing challenge when working with video footage. Because actions are (primarily) conducted by humans, modeling human behavior as a first step before extracting semantic content is of considerable importance.

Early action detection research [11, 2] uses human body parts or various human features to forecast what will happen in the video. Such detectors, on the other hand, are not always feasible and/or can be hard to compute. To solve this problem, unconstrained approaches have been developed [23, 22]. The emergence of end-to-end approaches [30] employing deep learning methods has altered the area of action detection for both unconstrained and human-centric methods, similar to image understanding frameworks.

Methods based on human models that have been articulated - When modeling movements in films from the perspective of the human body, knowing what information is required for motion detection is critical. [19] shows how visual interpretation of several moving light indicators linked to the human body can be adequate to identify a person's behavior. This groundbreaking study inspired techniques [40, 48, 5] to detect human activity utilizing trajectories of joint locations, landmarks, or body components based on 3D or 2D human-body models. Although body component localization produces outstanding results, it remains a tough challenge, particularly for unrestricted movies, limiting its usefulness.

Methods for Human Global Dynamics - scientists have proposed a less restrictive way to solving this noisy body part localization problem, which entails understanding global human-body dynamics given a human body-focused region of interest.

Modeling of human dynamics may be separated into two types. The first makes use of silhouette information and shape masks. Some advocate employing the foreground-to-background ratio in a grid over the silhouette in the first technique, which is based on silhouette photographs. Some people devise a technique based on form masks. They suggest utilizing momentum energy pictures and motion history images to recognize human activities, making them the first to offer a time-based sample for human action detection. One might alternatively propose

that silhouette information calculated using background removal be used to generate space-time forms. Local focus and form structure are among the qualities they extract.

The other makes use of optical flow and shape data. The usage of spatio-temporal lattices of optical flow magnitudes is a possibility. A two-step technique might be proposed by initial geo tracking football players in movies and then computing a descriptor on the somewhat stable tracks using fuzzy version of the mentioned optical flow. A similar method might be used to create mid-level descriptors given a somewhat lower-level optical flow data. [23, 24] proposes employing HOG, motion functions, and a pre-filter operation with a human features to identify drinking activities in movies.

One could look at man-made unrestricted local features that rely on people detection and/or trackable system propagates the first-stage inaccuracy. Several studies suggest handmade unrestricted approaches to handle this problem, comparable to the way employed in object detection. They adjust the local features to the underlying data format since the input signal might be both space and time dependent.

The modeling of low-level representations is one of the initial areas of research. By extending the idea of space-based outlook to the both space and time based region. [25] initiates the research in both space and time based interest points (STIP). They find local spatio-temporal structures with considerable structure using Harris and Förstner's point-of-interest operator. Difference between [26] (spatiotemporal points of interest) and [29] (purely spatial sites of interest). By matching local HOG features taken from two pictures, [30] comes up with Motion Boundary Histograms (MBH).

Chapter 3

Literature Review

3.1 Crime Detection

Earlier studies focused on crime detection from video footage were based on interrelation of convolutional neural networks (CNN) and recurrent neural networks (RNN, LSTM). One of the duties of the CNN was to retrieve vector representations of every frame, then feed these vector representations into the LSTM network, which classified them. Manifestations of this method are [45, 44]. Following is the block diagram of this approach:

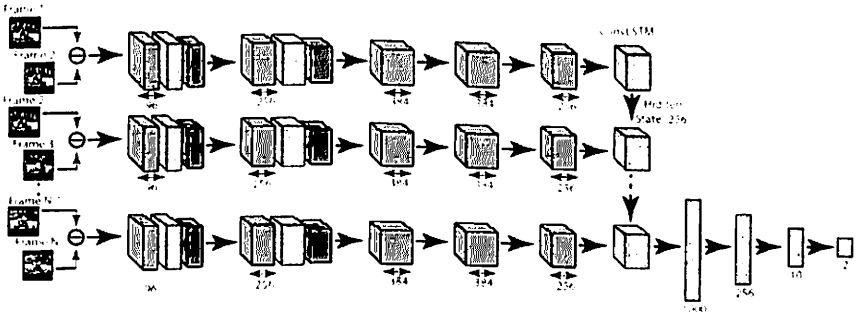


Figure 3.1: CNN LSTM

Convolution (red), normalization (grey), and pooling (blue) layers alternate throughout the model. It's important to note that the crime detection problem is linked to the action detection problem because crimes frequently entail violent scenes such as shoving, striking, kicking and so on. 3D convolutional neural networks are used as building blocks in modern and cutting-edge solutions to action recognition challenges. [7], for example, employs 3x3x3 convolution blocks as the proposed solution's building component. The following is a rudimentary diagram of stated.

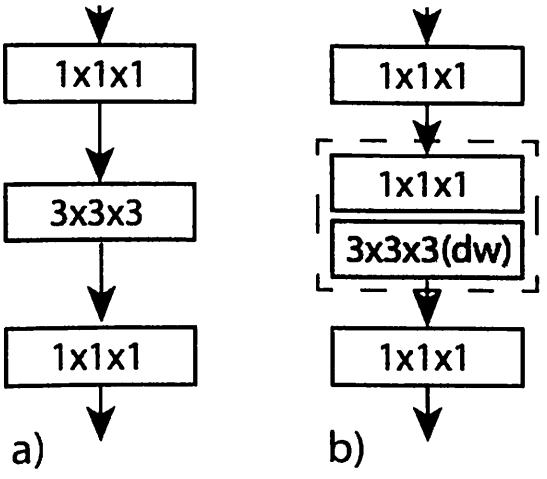


Figure 3.2: 3D Convolutional Blocks

There is a considerable corpus of research on using 2D and 3D CNNs to detect crime scenes from video footage in general. We chose four highly cited academic publications that employ 3D CNN to uncover crime in a variety of methods for our study. Simple 3D convolution layers were utilized by the researchers in [10] to anticipate crime situations. [7] combines 3D convolutional layers with a sequential LSTM architecture. Researchers employed the DarkNet neural network as part of a pre-training process in [8]. The authors of [39] employed a frame-by-frame crime detection approach to collect a new crime detection data set.

The 3D CNN architecture provided by [21] is used in reference [33] , as it appeared to be a good method in the context of behavioral detection in surveillance videos. Four 3D convolution layers, two max-pooling layers, and two completely connected layers make up the design. The Adam optimizer was utilized as the optimizer, and cross entropy was used for loss computations. The model generates a binary output format, with 1 indicating suspicious behavior and 0 indicating normal behavior. For model training, the Google Colaboratory [20] machine was used.

In [21] proposes a model that detects suspicious behavior rather than criminal activity. Pre-Crime Behavior Analysis, or PCB for short, is a new method of analyzing and extracting suspicious samples from video streams that it offers for this purpose. This technique is used to extract suspicious or criminal video segments and feed them to a deep learning model for training purposes, so that it may self-identify similar moments in the future. Each video fragment, according to the notion, has its own meaning. As a result, four categories of notions have been presented:

- Strict Crime Moment (SCM)
- Comprehensive Crime Moment (CCM)

- Criminal Case (CL)
- Precrime Behavior (PCB)

Reference [21] employs a novel method for assessing the suggested model's performance. The technique entails setting up two experiments with the goal of establishing the best model parameters and determining the best model performance by putting them to the test on the datasets provided. The first experiment, dubbed "Configuration Exploration," had the primary purpose of identifying improved data formats and model parameters. The datasets were split into two categories: symmetrical and asymmetrical. A balanced dataset is one in which each class has the same number of data samples. Because the data from an unbalanced dataset is returned in a 1:2 ratio, there are two normal tests for every one suspicious behavior test. To choose the optimal alternative, the following dataset parameters were used

- In each record class, the percentage of samples. There are 60, 120, and 240 samples in the balanced dataset. There are 90 suspicious behavior tests and 180 normal behavior assessments in Imbalance.
- The following proportions make up the percentage of test records: Approximately 20%, 30%, and 40% of the overall value
- The number of frames utilized for 3D convolution is specified by the Depth option. The numbers are 10, 30, and 90 respectively.
- The image sizes were set to 160120, 8060, 4030, or 3224 pixels.

A balanced dataset of 240 samples and an imbalanced dataset of 180 samples were constructed after different parameters in the datasets were selected. Each dataset has three depth and four resolution parameters, resulting in a total of 24 datasets. The final results suggest that high-resolution photos on unbalanced datasets of 10 to 30 depth produce the best outcomes. With an accuracy of 80.2%, these tests are the most accurate. "Statistical Validation" was the title of the second experiment. The plan was to use cross-validation to run the model 30 times over the largest data sets already prepared, starting with the parameters that demonstrated the best accuracy in the first trial. Again, there are two types of data sets: balanced and imbalanced. The 80x60 resolution produced the greatest results in the suspicious behavior detection task. With a depth of ten frames, it achieved accuracy rates of over 85% for both dataset types. The best accuracy was attained on the thirty-first run, with a 92.50% accuracy.

In addition, reference [10] introduces 3D-CNN using a technique termed unidirectional LSTM and its bidirectional modification. The 3D CNN extracts features from the frames and returns an action classification as an output, which is provided as an input to the LSTM to determine if the action on the frame is normal or

abnormal, and therefore exits the prediction task. When utilizing unidirectional LSTM, the following approach is employed for 3D-CNN, according to [10].

1. Get a sample video record (input from user)
2. Load the 3D CNN model
3. Spatial-temporal feature extraction
4. make prediction (LSTM model)
5. accuracy

In most circumstances, the bidirectional technique outperforms the unidirectional approach. Bidirectional LSTM accepts input in two directions, from past to future and from future to past, whereas unidirectional LSTM stores information from past inputs concurrently. This is the most important distinction between these two architectures. The technique for using bidirectional LSTM for 3D-CNN is similar to the previous one, with the exception that we develop a prediction section where we utilize a bidirectional model.

We often utilize LSTM in Recurrent Neural Nets for sequence problem tasks, but because video is a series of frames, or in other words, a series of images, each frame has its own set of attributes that must be extracted as priority or importance. FPS [45] is a word that describes the number of frames per second captured. Analyzing frame sequences to forecast the suspect's action is one of the most important challenges in surveillance video analysis. The output of the 3D- CNN is used as an input to the LSTM in the proposed model. Unidirectional LSTM was employed at first, however an extension termed Bidirectional LSTM was applied for better performance outcomes.

Unidirectional and bidirectional LSTM are employed in [10] because they can forecast the frame based on previous information. The quality of the frame prediction was assessed using the mean square error (MSE). The epochs were used to calculate the MSE of the train and test datasets. When comparing the end findings, we can see that 3D CNN with unidirectional LSTM has 64% and 70% training and testing accuracy, respectively. However, its modified variant, known as Bi-directional LSTM, has a training data accuracy of 72% and a test data accuracy of 68%.

According to reference [7] , a pre-trained algorithm named Dark-net19 should be used. It's worth mentioning that Darknet19 was pre-trained using 1 million photos from the ImageNet collection, according to [7] . As a result, Darknet can identify photographs into 1000 different categories, such as keyboard, mouse, and a variety of animals. For the purpose of avoiding degradation problems, the model has 19 layers of convolutions in addition to residual layers. If we feed our output from 3DCNN to LSTM as described in [10], the output from Darknet19 is fed

first to following residual layers, and then this output is considered as input to the LSTM layer, where the final choice is made into F and NF labels.

In terms of model architecture, Reference [33] has its own quirks. To circumvent degradation difficulties, a pre-trained system called Darknet19 [30] was used, which was fed image frames retrieved from the CCTV generated video stream and then fed the output frames to CNN. The ultimate conclusion is drawn through two categories labeled F and NF after the output from CNN is supplied as input to LSTM. If there is evidence of violence, the appropriate person is alerted by phone. They chose Darknet19 because it performed well and provided accurate results on Imagenet [20].

The implementation of the architecture without residual layers [7] does not produce totally correct results, according to [14]. However, when residual layers are used, the findings become fairly precise. It's worth mentioning that a basic CNN model without LSTM also outperforms the CNN + LSTM model, but the frame accuracy of such a model will be lower anyway. Finally, the Darknet19 [31] + Residual Layer [14] – LSTM architecture achieves a 98.5% accuracy result.

The reference [39] adds fine-tuning layers to ResNet50 [14], as well as a new database collection. Google gathered the data and divided it into three categories: B. Shoplifting 960 photos, robbery 2073 images, and burglary 1136 images. Because each image has its own resolution and format, all of the images have been compressed to 224x224 pixels and converted to RGB. Data augmentation was used on all ages to eliminate over- and under-fitting concerns. The data is fed into the Convolutional Neural Network after it has been pre-processed and manipulated (CNN). The model's performance was assessed in two stages. The first phase implies that performance will be measured against a test set of Google data. The model should then be fed real-time videos in the second phase. Frames are retrieved from the video and pre-processed, which includes scaling and sorting by RGB channel. The model then takes each frame as input and predicts the class label by returning the label with the highest probability value. Following [39], the following is an overview of the proposed method:

1. Get image from live video
2. Passes extracted frames to the CNN model
3. Shift the predicted label to Q for each frame.
4. Repeat step 3 for 'k' frames.
5. Take the mean of the last 'k' frames and then choose the label with the highest probability.

The dataset was split 80:20 for testing and training purposes. The model had an accuracy of 89 percent after testing. Paper parameters such as precision, memory, f1 value, and support, in addition to accuracy, give. In terms of the results,

the overall accuracy is 91.66%, which indicates that 91.66% of the anticipated positive values are truly positive. The average recall is 86%, which is the percentage of positives among all positives. The average f1 value is 88.33%, indicating a good mix of precision and memory. Based on the results, we can conclude that the model is highly capable of distinguishing between multiple classes. Because all of the photographs in this trial were collected by Google, there should be a strong data set for future advancements. That is, they differ in terms of size, color, resolution, and other factors.

Chapter 4

Methods and Results

4.1 Methods

Our plan is to identify crime scenes using 3D convolution blocks, which have demonstrated a good performance in many action detection problems.

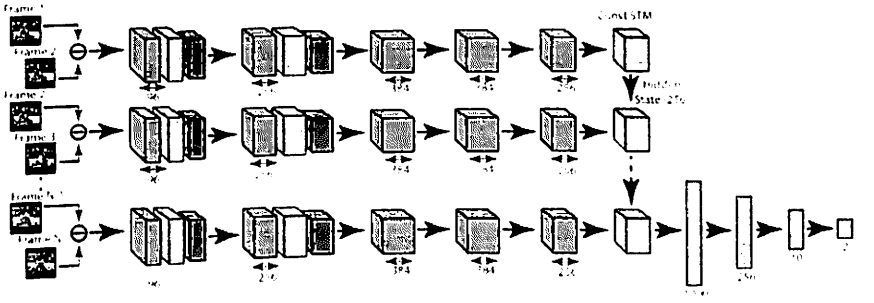


Figure 4.1: CNN LSTM

The organization utilized to identify criminal records is depicted in the diagram above. The convolutional layers are prepared to extract continuous highlights from the video frames, and the ConvLSTM layers are used to aggregate them at this stage. The following are the design options: The model is gradually coupled to the contours of the seen video. As soon as every outline is linked, the representation of the connected input video outlines is stored in the covered state of the ConvLSTM layer in this last time step.

This video representation is then applied to an array of fully connected layers for classification in the convLSTM occluded state. To extract outline-level features in the proposed show, we used the AlexNet show, which was pre-trained in the ImageNet database, as CNN illustrates. The following is a block diagram of our strategy.

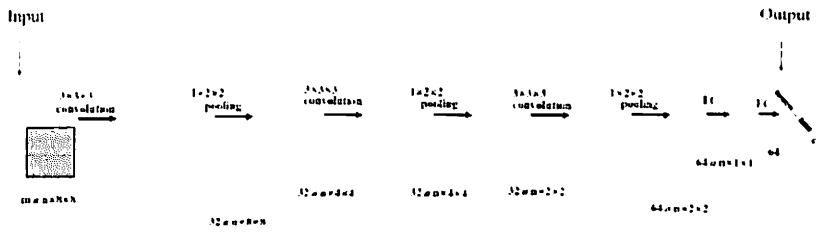


Figure 4.2: Chosen Architecture

The suggested technique is depicted as a block diagram.

4.1.1 Neural Network Architecture Design

We employ 3x3x3 convolution-based blocks in the first layer. After that, we use 1x2x2 pooling-based blocks. For each CNN, pooling is a crucial notion. It reduces the number of mappings between convolution blocks, which reduces the computational load. We'll go through some subsequent pooling algorithms utilized in CNNs in this section. This pattern is repeated two more times, resulting in two components. A softmax layer with the same quantity of elements as the quantity of crime type classes appears in the output. We have used a cross-entropy loss as our objective function. Here is the sample from PyTorch code we have written.

```
NUM_CLASSES = 4

class C3D(nn.Module):
    def __init__(self,
                 sample_size,
                 sample_duration,
                 num_classes=NUM_CLASSES):

        super(C3D, self).__init__()
        self.group1 = nn.Sequential(
            nn.Conv3d(3, 64, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool3d(kernel_size=(1, 2, 2)))
        self.group2 = nn.Sequential(
            nn.Conv3d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool3d(kernel_size=(2, 2, 2)))
        self.group3 = nn.Sequential(
            nn.Conv3d(128, 256, kernel_size=3, padding=1),
            nn.ReLU(),
            nn.MaxPool3d(kernel_size=(2, 2, 2)))
        last_duration = int(math.floor(sample_duration / 8))
        last_size = int(math.ceil(sample_size / 32))
        self.fc1 = nn.Sequential(
            nn.Linear(256 * last_duration * last_size * last_size, 4096),
            nn.ReLU(),
            nn.Dropout(0.5))
        self.fc2 = nn.Sequential(
            nn.Linear(4096, 4096),
            nn.ReLU(),
            nn.Dropout(0.5))
        self.fc = nn.Sequential(
            nn.Linear(4096, num_classes))
```

Figure 4.3: Neural Network Code Sample

4.1.2 Dataset

The Violent-Flows Crown Violence dataset was used to train our model. This dataset contains 246 films depicting both violent and nonviolent crowd behavior during sporting events. We used random cropping and horizontal flipping across frames to extend the dataset.

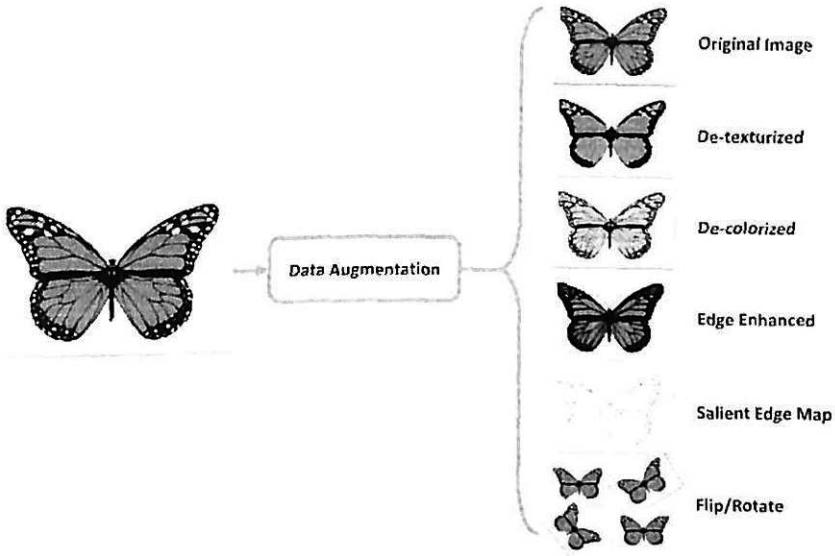


Figure 4.4: Data Augmentation

Street fighting, assault, robbery, and police brutality were the four categories in which the films were categorized. Oversampling and video augmentation techniques were used to create classes of the same size. Since training dataset size was too large for a RAM storage, we needed to desing a dataset abstraction in Pytorch. Dataset abstraction helped us to perform batch optimization using Stochastic Gradient Descent (SGD). We have also written a data preprocessing/transformation code for preprocessing video snippets into corresponding tensors. In order to get original video snippets we have written a denormalization code. Here is the sample code we have written.

```
preprocess = transforms.Compose([
    TrgListToTensor(),  # list of PIL Images to [NUM_SEGMENTS*FRAMES_PER_SEGMENT, CHANNELS, HEIGHT, WIDTH] tensor
    transforms.Resize(),  # Image patch, resize smaller side to
    transforms.CenterCrop(),  # Image patch, center crop to square
    transforms.Normalize(mean=[, , ], std=[, , ]),
])

dataset = VideoFrameDataset(
    root_path=videos_root,
    annotationfile_path=annotation_file,
    num_segments=,
    frames_per_segment=,
    imagefile_template='img_{:05d}.jpg',
    transform=preprocess,
    test_mode=False
)

sample = dataset[]
frame_tensor = sample[]  # tensor of size [NUM_SEGMENTS*FRAMES_PER_SEGMENT, CHANNELS, HEIGHT, WIDTH]
label = sample[]  # integer label

print('Video Tensor Size:', frame_tensor.size())

def denormalize(video_tensor):
    """
    Undoes mean/standard deviation normalization, zero to one scaling,
    and channel rearrangement for a batch of images.
    args:
        video_tensor: a [FRAMES x CHANNELS x HEIGHT x WIDTH] tensor
    """
    inverse_normalize = transforms.Normalize(
        mean=[, , ],
        std=[, , ])
    return (inverse_normalize(video_tensor) * ).type(torch.uint8).permute(, , , ).numpy()
```

Figure 4.5: Dataset Abstraction in PyTorch

Two Nvidia Titan RTX graphical processing units (GPUs) were used for the training. We utilized a stack size of 32 frames and a learning rate of 0.001 as a starting point. The training took place over the course of two weeks.

4.2 Results

The trained model predicted four different crime classifications with an accuracy of 87% for the given four classes. The ROC curve (one-versus-all for the first class) for the test data set may be seen below.

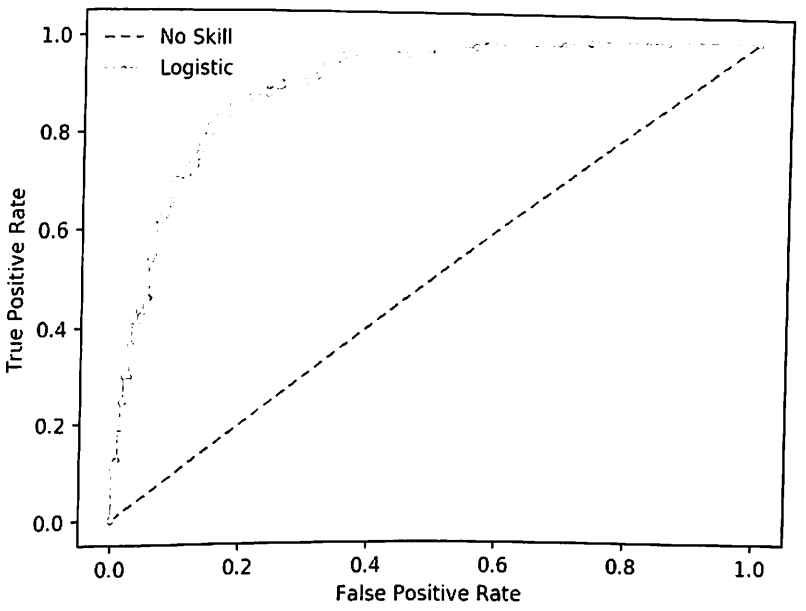


Figure 4.6: ROC curve

The resulting model's accuracy is higher than the accuracy given in [45] (59%) and [44] (64%). This illustrates how 3D convolution blocks may be used to forecast crime kinds from video data. The downside of the current model could be its relative slowness of the inference phase. It takes roughly 8 minutes to infer a video with a runtime of 5 minutes, i.e. we require about 60% more time than the film's duration. This translates into difficulty of using the current approach for real-time implementation.

Chapter 5

Conclusion

Street crime is on the rise around the world. Although the number of CCTV cameras is also rising, it takes a lot of manpower to analyze every video footage. As a part of addressing this issue, we have tried to build a system for crime type classification from video footage. The system works with limited number of crime types, however it has shown some promising results. This thesis used video footage to provide a fresh solution to the problem of crime categorization. The method shown is superior to existing state-of-the-art techniques.

In the real life, it is preferable to have a system that provides a real-life predictions for the coming video footage. Moreover, one might try to incorporate the algorithm into hardware itself in order to avoid internet connection risks. For this reason, we intend to improve the existing approach so that it may be employed in a real-time setting. Also, we wish to investigate to run the future algorithms in edge devices such as cameras.

References

[1] Chalfin A. and McCrary J. "Are US cities underpoliced?" In: *Rev. Econ. Stat.* 100 (2018), pp. 167–186.

[2] Efros A. A. et al. "Recognizing action at a distance". In: *ICCV* (2003), pp. 40–41.

[3] Krizhevsky A., Sutskever I., and Hinton G. E. "ImageNet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems* 25 (2012), pp. 1097–1105.

[4] Le Cun Y. A. et al. "Neural networks: Tricks of the trade". In: *Lecture Notes in Computer Science,* 7700 (2012), pp. 9–48.

[5] Yilmaz A. and M. Shah. "Recognizing human actions in videos acquired by uncalibrated moving cameras". In: *ICCV* (2005), pp. 41–47.

[6] Bishop C. M. et al. *Pattern recognition and machine learning,* Springer, 2006.

[7] B. U. Anu Barathi et al. "Suspicious action and behavior detection using CNN". In: *IJCSMC* 9 (2020), pp. 51–59.

[8] Schuldt C., Laptev I., and Caputo B. "Recognizing human actions: a local SVM approach". In: *Proceedings of the 17th International Conference on Pattern Recognition* 3 (2004), pp. 32–36.

[9] A. M. Costa. "The economics of crime: a discipline to be invented and a Nobel prize to be awarded". In: *J. Policy Model* 32 (2010), pp. 648–661.

[10] Mrs. Manju D., Dr. Seetha M., and Dr. Sammulal P. "Early action prediction using 3DCNN with LSTM and bidirectional LSTM". In: *Turkish Journal of Computer and Mathmatics Education* 12 (2021), pp. 2275–2281.

[11] Ramanan D., D.A. Forsyth, and A. Zisserman. "Tracking people by learning their appearance". In: *IEEE T-PAMI* (2007), pp. 40–41.

[12] Weinland D., Ronfard R., and E. Boyer. "A survey of vision-based methods for action representation, segmentation and recognition". In: *Computer Vision and Image Understanding* 115 (2011), pp. 224–241.

[13] Zeiler M. D. and Fergus R. "Visualizing and understanding convolutional networks." In: *Computer Vision - ECCV* 8689 (2014), pp. 818–833.

[14] "Deep Residual Learning for Image Recognition". In: *arXiv:1512.03385* ().

[15] City of Detroit. *Crime Incidents*. 2020. URL: `https://data.detroitmi.gov/datasets/rms-crime-incidents`. Accessed June 2, 2020.

[16] Hinton G. E. and Salakhutdinov R. R. "Reducing the dimensionality of data with neural networks". In: *Science* 313 (2006), pp. 504–507.

[17] Smeaton A. F., Over P., and Kraaij W. "Evaluation campaigns and TRECVid". In: *Proceedings of the 8th ACM international workshop on Multimedia information retrieval* (2006), pp. 321–330.

[18] P. Fajnzylber, D. Lederman, and N. Loayza. "What causes violent crime?" In: *Eur. Econ. Rev* 46 (2002), pp. 1323–1357.

[19] Johansson G. "Visual perception of biological motion and a model for its analysis". In: *Perception and Psychophysics* (1973), pp. 4–40.

[20] Google. "Google colaboratory". In: (2017).

[21] Martínez Mascorro Guillermo et al. "Suspicious Behavior Detection on Shoplifting Cases for Crime Prevention by Using 3D Convolutional Neural Networks". In: ().

[22] Wang H., A. Kläser, and C. Schmid. "Dense trajectories and motion boundary descriptors for action recognition". In: *IJCV* (2013), pp. 42–44.

[23] Laptev I. and T. Lindeberg. "On space-time interest points". In: *ICCV* (2003), pp. 40–41.

[24] Laptev I. and T. Lindeberg. "On space-time interest points". In: *International journal of computer vision* (2005), pp. 107–123.

[25] Laptev I. and P. Perez. "Retrieving actions in movies". In: *ICCV* (2007), pp. 38–45.

[26] Laptev Ivan. "Modeling and visual recognition of human actions and interactions". In: *Habilitation à diriger des recherches Ecole Normale Suprieure* (2013), pp. 4–13.

[27] Mena J. *Machine Learning Forensics for Law Enforcement, Security, and Intelligence*. CRC Press, 2011.

[28] Tastle W. J. "Introduction to artificial networks and law enforcement analytics". In: *Intelligent Data Mining in Law Enforcement Analytics* (2013), pp. 1–9.

[29] M. Beigl et al. J. Borges D. Ziehr. "Feature engineering for crime hotspot detection". In: *IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computed, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovation* (2017), pp. 1–8.

[30] Carreira Joao and Andrew Zisserman. ""Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset". In: *CVPR* (2017), pp. 110–112.

[31] Redmon Joseph and Farhadi Ali. "YOLO9000: Better, Faster, Stronger". In: *CVPR* (2017), pp. 6517–6525.

[32] Schindler K. and Van Gool L. "Action snippets: How many frames does human action recognition require?" In: *IEEE Conference on Computer Vision and Pattern Recognition* (2008), pp. 1–8.

[33] Fujimoto Lab. "3D convolutional neural network for video classification". In: (2017).

[34] Baccouche M. et al. "Sequential deep learning for human action recognition". In: *Lecture Notes in Computer Science* 7065 (2011), pp. 29–39.

[35] Yang M. et al. "Detecting human actions in surveillance videos". In: *Proceedings of the TrecVID Video Evaluation Workshop* (2009).

[36] U. V. Navalgund and K. Priyadharshini. "Crime intention detection system using deep learning". In: *International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)* (2018), pp. 1–6.

[37] Zhang X. Q. "The trends, promises and challenges of urbanisation in the world. Habitat Int." In: *J. Policy Model* 54 (2016), pp. 241–252.

[38] Zhu Q. et al. "An anticrime information support system design: application of K-means-VMD-BiGRU in the city of Chicago". In: *Inf. Manag.* (2019).

[39] Om M. Rajpurkar et al. "Alert generation on detection of suspicious activity using transfer learning". In: *11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (2020).

[40] Ali S., A. Basharat, and M. Shah. "Chaotic invariants for human action recognition". In: *ICCV* (2007), pp. 40–44.

[41] Ji S. et al. "3D convolutional neural networks for human action recognition". In: *Proceedings of the 27th International Conference on Machine Learning* (2010), pp. 495–502.

[42] Ji S. et al. "3D convolutional neural networks for human action recognition." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), pp. 221–231.

[43] Stebbins S. *The Midwest is Home to Many of America's Most Dangerous Cities. USA Today.* URL: `https://www.usatoday.com/story/money/2019/10/26/crime-rate-higher-us-dangerous-cities/40406541/`. Accessed July 15, 2020.

[44] Sudhakaran S. and Lanz O. "Learning to detect violent videos using convolutional long short-term memory". In: *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference* (2017), pp. 1–6.

[45] Xingjian S.H.I. et al. "Advances in neural information processing systems". In: (2015), pp. 802–810.

[46] Tumulak J. A. U. and Espinosa K. J. P. "Crime modelling and prediction using neural networks". In: *Theory and Practice of Computation: Proceedings of Workshop on Computation: Theory and Practice* (2017), pp. 218–228.

[47] Le Q. V. et al. "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2011), pp. 3361–3368.

[48] Parameswaran V. and R. Chellappa. "View invariance for human action recognition". In: *IJCV* (2006), pp. 32–36.

[49] Glorot X., Bordes A., and Bengio Y. "Deep sparse rectifier networks". In: *14th International Conference on Artificial Intelligence and Statistics* 15 (2011), pp. 315–323.

[50] Le Cun Y. et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86 (1998), pp. 2278–2324.

[51] S. Yadav et al. "Crime pattern detection, analysis and prediction". In: *International conference of Electronics, Communication and Aerospace Technology (ICECA)* 1 (2017), pp. 225–230.