# **CHAPTER – 1**

**INTRODUCTION**

* 1. **About the project:**

Traffic congestion is one of the major modern-day crisis in every big city in the world. Recent study of World Bank has shown that average vehicle speed has been reduced from 21 km to 7 km per hour in the last 10 years in Dhaka [1] Intermetropolitan area studies suggest that traffic congestion reduces regional competitiveness and redistributes economic activity by slowing growth in county gross output or slowing metropolitan area employment growth [2].As more and more vehicles are commissioning in an already congested traffic system, there is an urgent need for a whole new traffic control system using advanced technologies to utilize the already existent infrastructures to its full extent. Since building new roads, flyovers, elevated expressway etc. needs extensive planning, huge capital and lots of time; focus should be directed upon availing existing infrastructures more efficiently and diligently. glean traffic data. Some of them count total number of pixels [3], some of the work calculate number of vehicles [4- 6].These methods have shown promising results in collecting traffic data. However, calculating the number of vehicles may give false results if the intravehicular spacing is very small (two vehicles close to each other may be counted as one) and it may not count rickshaw or auto-rickshaw as vehicles which are the quotidian means of traffic especially in South-Asian countries. And counting number of pixels has disadvantage of counting insubstantial materials as vehicles such as footpath or pedestrians. Some of the work have proposed to allocate time based solely on the density of traffic. But this may be disadvantageous for those who are in lanes that have less frequency of traffic.

# **CHAPTER – 2**

**SYSTEM ANALYSIS**

**2.1. Existing System:**

Edge detection technique is imperative to extract the required traffic information from the CCTV footage. It can be used to isolate the required information from rest of the image. There are several edge detection techniques available. They have distinct characteristics in terms of noise reduction, detection sensitivity, accuracy etc. Among them, Prewitt [7], canny [8],Sobel [9], Roberts and LOG are most accredited operators. It has been observed that the Canny edge detector depicts higher accuracy in detection of object with higher entropy, PSNR(Peak Signal to Noise Ratio), MSE(Mean Square Error) and execution time compared with Sobel, Roberts, Prewitt, Zero crossing and LOG [10-12].Here is a comparison between distinct edge detection techniques [13]. To implement this technique we are uploading current traffic image to the application and application will extract edges from images and if there is more traffic then there will be more number of edges with white colour and if uploaded image contains less traffic then it will have less number of white colour edges.

**Drawbacks of Existing Systems:**

The existing systems for smart traffic management, which often involve traditional approaches like rule-based algorithms, basic image processing, or even conventional machine learning models, can have several drawbacks when it comes to handling dynamic and complex traffic situations. When using methods like Canny edge detection combined with Convolutional Neural Networks (CNNs), some limitations of the existing systems might include:

**1. Limited Accuracy in Complex Scenarios**

* **Edge Detection**: Canny edge detection, while effective for identifying boundaries in an image, is not robust in highly dynamic or cluttered environments, such as intersections with multiple vehicles and varying lighting conditions. It may fail to detect edges in noisy images, affecting the performance of downstream traffic prediction systems.
* **CNNs Limitations**: Even though CNNs are powerful for feature extraction, they can still struggle to differentiate between closely spaced vehicles or handle extreme weather conditions that may affect visibility.

**2. Real-Time Processing Challenges**

* **Computation Overhead**: CNNs require significant computational resources, particularly when applied to large-scale traffic data (e.g., high-resolution images, real-time video feeds). This can make real-time processing difficult unless hardware resources are highly optimized or distributed.
* **Edge Detection Performance**: Canny edge detection might add extra computation complexity when trying to integrate it with real-time tracking or decision-making systems.

**3. Sensitivity to Environmental Factors**

* **Lighting Variations**: Canny edge detection is highly sensitive to lighting conditions (e.g., low light or bright sunlight), which can affect edge quality and reduce accuracy.
* **Weather Conditions**: Fog, rain, and snow can obscure important visual data, causing edge detection and CNN-based models to struggle with poor visibility.

**4. Scalability Issues**

* **Handling Large Traffic Data**: As the number of cameras or sensors increases, scaling the system becomes more complex, as it requires more processing power and bandwidth for transmitting real-time data to central servers.
* **CNN Training Complexity**: Training a CNN for traffic analysis involves a large dataset, and maintaining the model over time (retraining, fine-tuning, etc.) can be resource-intensive.

**5. Edge Detection Limitations**

* **Noise Sensitivity**: Canny edge detection can be highly sensitive to noise in the image. This can lead to false edges or missed features, which can impact downstream CNN models’ ability to accurately predict traffic flow.
* **Incomplete Detection**: Canny edge detection may not work well in scenarios where the edges are not clearly defined, such as when vehicles are far apart or the road is unclear, which could reduce the effectiveness of traffic flow predictions.

**6. Limited Generalization**

* **Generalization to Different Traffic Patterns**: Traffic flow patterns vary greatly by region, time of day, and season. CNN models trained on limited datasets might not generalize well to new traffic conditions or regions with different behaviors.

**7. Cost of Maintenance**

* **System Updates**: Keeping the edge detection algorithms and CNN models up to date requires continuous maintenance. New edge cases and evolving traffic patterns must be incorporated into the system, which can involve retraining and testing the model.

**8. Data Privacy Concerns**

* **Surveillance**: The use of cameras for real-time traffic monitoring raises privacy concerns, particularly in cities where individuals may not be comfortable with constant video surveillance, which could create regulatory hurdles.

**2.2 Proposed System:**

In this paper, a system in which density of traffic is measured by comparing captured image with real time traffic information against the image of the empty road as reference image is proposed. Here, in figure 1, the block diagram for proposed traffic control technique is illustrated. Each lane will have a minimum amount of green signal duration allocated. According to the percentage of matching allocated traffic light duration can be controlled. The matching is achieved by comparing the number of white points between two images. The entire image processing before edge detection i.e. image acquisition, image resizing, RGB to gray conversion and noise reduction is explained in section II. At section III, canny edge detection operation and white point count are depicted. Canny edge detector operator is selected because of its greater overall performance.

**Advantages**:

it is advantageous to convert RGB images into grayscale for further processing. When converting an RGB image to grayscale, it is pertinent to consider the RGB values for each pixel and make as output a single value reflecting the brightness of that pixel. One of the approaches isto take the average of the contribution from each channel:(R+B+C)/3.

**2.3 Module Description:**

The system is divided into several modules, each handling a specific aspect of the cybersecurity workflow. Below is a detailed description of the core modules and their functionalities:

**1. NumPy (numpy)**

* **Numerical computing library** that provides support for arrays and matrices, along with a wide range of mathematical functions to operate on these arrays.

**Usage:**

* Handling large datasets in machine learning projects.
* Numerical operations on data, such as matrix manipulations.

**2. Pandas (pandas)**

* **Data manipulation and analysis** library used for working with structured data, such as CSV files, Excel spreadsheets, and SQL databases.

**Usage:**

* Data cleaning, exploration, and preparation for machine learning models.

**3. Scikit-learn (sklearn)**

* **Machine learning library** that provides simple tools for data analysis and modeling. It includes a wide variety of algorithms for classification, regression, clustering, and dimensionality reduction.

**Usage:**

* Building and evaluating machine learning models in classification, regression, and clustering tasks.

**4. Seaborn (seaborn)**

* **Data visualization library** built on top of Matplotlib, which makes it easy to create attractive and informative statistical graphics.

**Usage:**

* Creating insightful and visually appealing plots for data exploration and presentation.

**5. Matplotlib (matplotlib)**

* **2D plotting library** used for creating static, animated, and interactive visualizations.

**Usage:**

* Basic plotting for visualizing datasets and model outputs.

**7. TensorFlow (tensorflow)**

* **Open-source machine learning** framework that allows the creation and deployment of machine learning models, especially deep learning models.

**Usage:**

* Building deep learning models, training, and deploying them on cloud or edge devices.

**2.4 Feasibility Study:**

A **feasibility study** for a **Density-Based Smart Traffic System using Canny Edge Detection and Convolutional Neural Networks (CNNs)** would typically evaluate various aspects like technical feasibility, financial viability, and potential challenges.

Here’s how you could break down the feasibility study:

**1. Problem Definition**

* **Traffic Congestion**: With increasing urbanization, traffic congestion is becoming a significant challenge in many cities. A smart traffic system that can efficiently manage traffic flow is crucial.
* **Objective**: The goal is to develop a traffic management system that can dynamically adjust signal timings based on traffic density using real-time image processing.

**2. Proposed Solution**

* **Canny Edge Detection**: This method will be used for edge detection in traffic images/videos. The Canny edge detection technique helps identify the contours and boundaries of vehicles, which can then be analyzed to estimate traffic density in a given area.
* **Convolutional Neural Networks (CNNs)**: These networks can be used to analyze patterns in traffic flow, learning to recognize traffic density from processed images. CNNs are particularly effective for image-based tasks and can be trained to classify traffic conditions, predict congestion, and optimize signal timings accordingly.

**3. Technological Feasibility**

* **Image Processing**: Canny edge detection is widely used in image processing tasks due to its efficiency and simplicity. It will be effective for detecting vehicles in a traffic scene.
* **Machine Learning Models**: CNNs are well-suited for pattern recognition tasks like traffic flow analysis. Training a CNN to predict traffic density can be done with labeled data (images with traffic density annotations).
* **Real-time Processing**: Modern GPUs and edge computing devices can enable real-time image processing and predictions. With proper hardware acceleration, the system can work in real-time traffic monitoring.
* **Integration with Traffic Signals**: The smart traffic system can be integrated with existing traffic signal infrastructure to adjust signal timings based on CNN predictions.

**4. Data Requirements**

* **Training Data**: To train the CNN, a large dataset of traffic images/videos is needed, along with labels for traffic density. This could be collected from traffic cameras or public datasets.
* **Real-time Data**: Live video feeds from traffic cameras would be processed for real-time traffic density estimation.

**5. Implementation Challenges**

* **Data Quality**: Traffic data can vary significantly based on weather, time of day, or camera angle. Ensuring that the CNN generalizes well to all conditions is a potential challenge.
* **Real-time Performance**: Real-time processing of high-resolution video streams requires significant computational power, which may be a bottleneck in resource-constrained environments.
* **Integration**: Integrating the system with existing traffic control infrastructure could be complex, especially in older systems that may not support dynamic changes in signal timings.

**6. Financial Feasibility**

* **Initial Investment**: Setting up cameras, processing units, and training a CNN would require a significant initial investment in both hardware and software development.
* **Operational Costs**: Ongoing costs for maintaining cameras, processing systems, and possibly the need for periodic retraining of the CNN on new traffic patterns.
* **Return on Investment**: The system can reduce traffic congestion, improve road safety, and decrease fuel consumption, which can lead to substantial long-term savings and benefits.

**7. Social and Environmental Impact**

* **Benefits**: Reduced traffic congestion can lead to lower emissions, decreased fuel consumption, and improved overall quality of life for urban residents.
* **Challenges**: Privacy concerns regarding the use of surveillance cameras and the handling of traffic data must be addressed.

**8. Conclusion**

* The combination of Canny edge detection for vehicle detection and CNNs for traffic density prediction is a promising approach to implementing a smart traffic system.
* The technical feasibility is high with modern hardware and algorithms, but practical challenges like real-time performance, data quality, and integration with legacy infrastructure need to be addressed.
* Financially, while initial costs may be high, the long-term benefits in terms of congestion reduction and environmental impact could justify the investment.

CHAPTER - 3

**REQUIREMENT ANALYSIS**

**HARDWARE SYSTEM REQUIREMENTS:**

**➢ System :Pentium i3 Processor.**

**➢ Hard Disk : 500 GB.**

**➢ Monitor: 15’’ LED**

**➢ Input Devices: Keyboard, Mouse**

**➢ Ram: 4 GB**

**SOFTWARE SYSTEM REQUIREMENTS:**

**➢Operating system : Windows 10 .**

**➢ Coding Language:Python Web**

**➢ Web Framework :Flask**

* 1. **functional Requirements:**

For a project like **Density-Based Smart Traffic Using Canny Edge Detection and Convolutional Neural Networks (CNNs)**, both functional and non-functional requirements can be defined to ensure a robust, efficient, and scalable system.

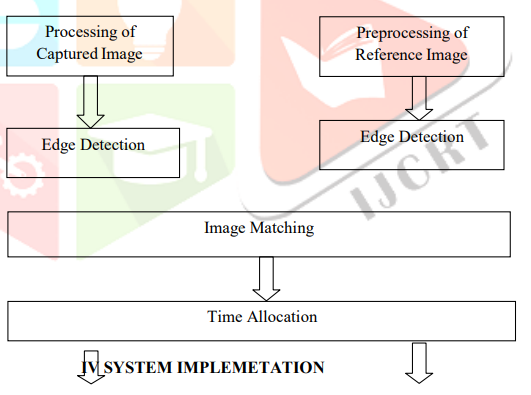
1. **Traffic Density Detection**: The system must be able to detect and classify the traffic density in real-time using images or videos from traffic cameras.
2. **Edge Detection Using Canny Algorithm**: The system must apply the Canny edge detection technique to enhance the image by identifying edges of objects, vehicles, and road boundaries.
3. **Vehicle Classification**: The system should identify and classify different types of vehicles (e.g., cars, bikes, buses) using CNNs.
4. **Traffic Flow Prediction**: The system should use machine learning models to predict traffic flow and congestion patterns based on current and past data.
5. **Signal Control Optimization**: Based on traffic density, the system must automatically adjust the traffic signal timing to minimize congestion.
6. **Real-time Processing**: The system should be capable of processing live video feeds from traffic cameras to detect traffic density and adjust signals in real-time.
7. **Alerts and Notifications**: The system should send alerts to traffic authorities when traffic congestion exceeds a threshold, or accidents are detected.
8. **Data Storage and Retrieval**: The system must store traffic data, including images and vehicle classifications, for future analysis and reporting.
   1. **Non-Functional Requirements:**
9. **Performance:** The system should process traffic images and videos with minimal latency, ensuring real-time traffic management.
10. **Scalability:** The system should be scalable to handle traffic monitoring in large cities or multiple intersections without compromising performance.
11. **Accuracy:** The Canny edge detection and CNN-based classification should be accurate, with low false-positive and false-negative rates in vehicle detection and traffic flow prediction.
12. **Reliability:** The system must function reliably under various environmental conditions, such as poor lighting or inclement weather, which may affect image quality.
13. **Robustness:** The system should handle various edge cases like traffic jams, accidents, and unusual patterns in the traffic flow without failing.
14. **Security:** The system should secure sensitive data (such as traffic data and surveillance footage) and ensure that unauthorized access is prevented.
15. **Maintainability:** The system should be easy to update and maintain, including adding new features or improving the CNN model with new datasets.
16. **Interoperability:** The system must be compatible with existing traffic monitoring systems and sensors**.**
17. **Usability:** The system's interface (for traffic management authorities) should be intuitive and easy to navigate.
18. **Energy Efficiency:** The system should be optimized for energy consumption, especially for large-scale deployments with many traffic cameras.

# CHAPTER – 4

**SYSTEM DESIGN**

* 1. **System Architecture:**

A Density-Based Smart Traffic System using the Canny Edge Detection method and Convolutional Neural Networks (CNNs) can be quite innovative. Below is a high-level architecture that integrates these components for an efficient smart traffic **s**ystem**.**

****

**Fig 4.1 System Architecture**

**4.3 Unified Modeling Language Design:**

UML stands for Unified Modeling Language. UML is a standardized generalpurpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying,

Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**4.3.1 Use Case Diagram:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

**Fig 4.3 Use case Diagram for user**

**4.3.1 Class Diagram**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

**Fig: 4.3.1 Class Diagram**

**4.3.3 Sequence Diagram**:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

**Fig 4.5 Sequence Diagram for user**

**4.4 Input Design:**

Input design is a part of overall system design. The main objective during the input design is as given below:

* To produce a cost-effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that the input is acceptable and understood by the user.

**Input Stages**

The main input stages can be listed as below:

* Data recording
* Data transcription
* Data conversion
* Data verification
* Data control
* Data transmission
* Data validation
* Data correction

Input Types

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

* External inputs, which are prime inputs for the system.
* Internal inputs, which are user communications with the system.
* Operational, which are computer department’s communications to the system?
* Interactive, which are inputs entered during a dialogue.

**Input Media**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

* Type of input
* Flexibility of format
* Speed
* Accuracy
* Verification methods
* Rejection rates
* Ease of correction
* Storage and handling requirements
* Security
* Easy to use
* Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

**Error Avoidance**

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

**Error Detection**

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

**Data Validation**

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

**User Interface Design**

It is essential to consult the system users and discuss their needs while designing the user interface:

**User Interface Systems Can Be Broadly Clasified As:**

* User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
* Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

**User Initiated Intergfaces**

User initiated interfaces fall into two approximate classes:

* Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
* Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

**Computer-Initiated Interfaces**

The following computer – initiated interfaces were used:

* The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
* Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

**Error Message Design**

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

**Performance Requirements**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system.
* The system should be accurate.
* The system should be better than the existing system.
* The existing system is completely dependent on the user to perform all the duties.

**4.5 Output Design:**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

* External Outputs, whose destination is outside the organization
* Internal Outputs whose destination is within organization and they are the
* User’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.

**Output Definition**

The outputs should be defined in terms of the following points:

* Type of the output
* Content of the output
* Format of the output
* Location of the output
* Frequency of the output
* Volume of the output
* Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

CHAPTER - 5

**IMPLEMENTATION**

**5.1 Machine Learning Technology:**

Machine learning is a discipline that deals with programming the systems so as to make them automatically learn and improve with experience. Here, learning implies recognizing and understanding the input data and taking informed decisions based on the supplied data. It is very difficult to consider all the decisions based on all possible inputs. To solve this problem, algorithms are developed that build knowledge from a specific data and past experience by applying the principles of statistical science, probability, logic, mathematical optimization, reinforcement learning, and control theory.

**5..2 Steps Involved in Machine Learning:**

A machine learning project involves the following steps

* + Defining a Problem
  + Preparing Data
  + Evaluating Algorithms
  + Improving Results • Presenting Results

Machine Learning (ML) is an automated learning with little or no human intervention. It involves programming computers so that they learn from the available inputs. The main purpose of machine learning is to explore and construct algorithms that can learn from the previous data and make predictions on new input data.The input to a learning algorithm is training data, representing experience, and the output is any expertise, which usually takes the form of another algorithm that can perform a task. The input data to a machine learning system can be numerical, textual, audio, visual, or multimedia.

**5.3 Machine Learning vs. Traditional Programming:**

Traditional programming differs significantly from machine learning. In traditional programming, programmers code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become sustainable to maintain.



**Fig 5.1: Traditional Programming**

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.

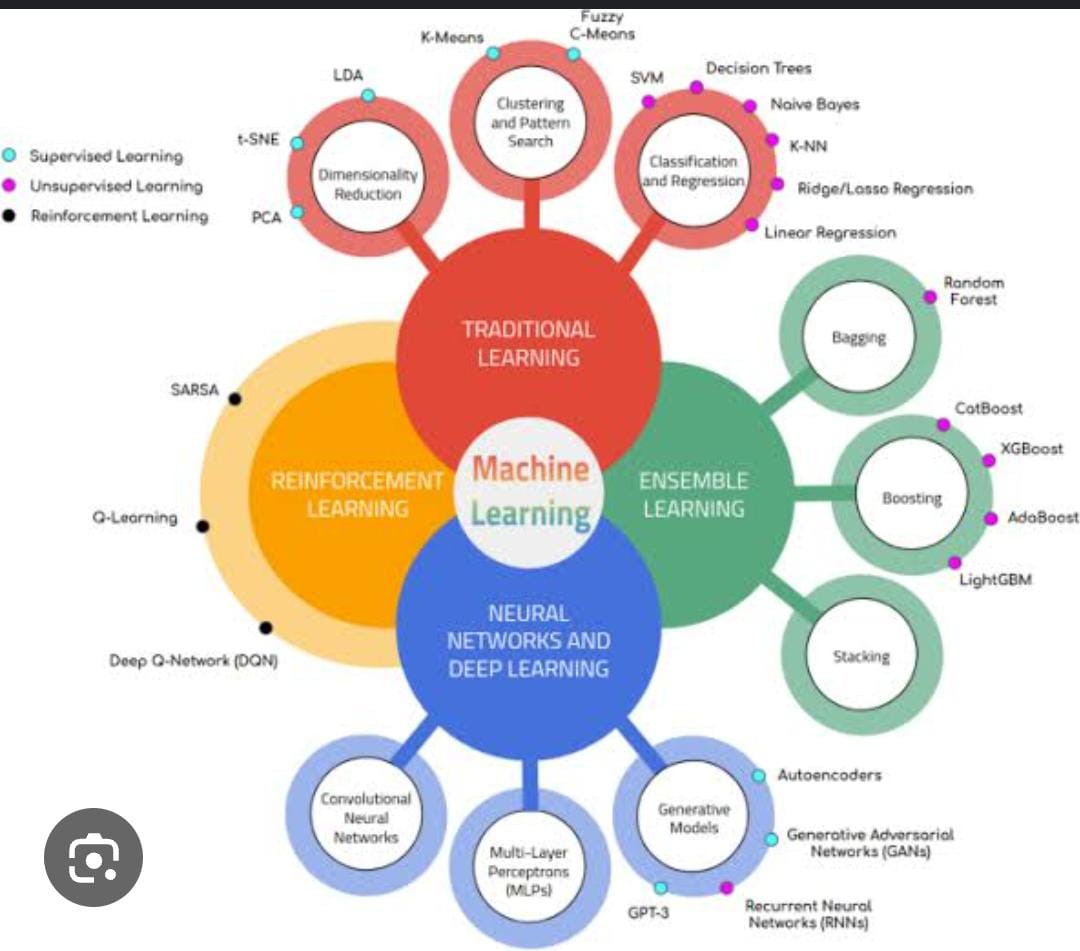
## 

**5.2: Machine Learning**

**5.4 How does Machine learning work?**

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it’s feed a previously unseen example, the machine has difficulties to predict.

5.5 Machine learning Algorithms and where they are used?



**Fig 5.3: Machine Learning Algorithms**

**5.6 Types of Machine Learning:**

There are four categories of machine learning algorithms as shown below

* Supervised learning algorithm
* Unsupervised learning algorithm
* Semi-supervised learning algorithm
* Reinforcement learning algorithm

**5.6.1 Supervised Learning:**

Supervised learning is commonly used in real world applications, such as face and speech recognition, products or movie recommendations, and sales forecasting.

Supervised learning can be further classified into two types - Regression and Classification. Regression trains on and predicts a continuous-valued response, for example predicting real estate prices. Classification attempts to find the appropriate class label, such as analyzing positive/negative sentiment, male and female persons, benign and malignant tumors, secure and unsecure loans etc. In supervised learning, learning data comes with description, labels, targets or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called labeled data. The learned rule is then used to label new data with unknown outputs. Supervised learning involves building a machine learning model that is based on labeled samples.

**5.6.2 Unsupervised Learning:**

Unsupervised learning is used to detect anomalies, outliers, such as fraud or defective equipment, or to group customers with similar behaviors for a sales campaign. It is the opposite of supervised learning. When learning data contains only some indications without any description or labels, it is up to the coder or to the algorithm to find the structure of the underlying data, to discover hidden patterns, or to determine how to describe the data. We may not exactly know what the criteria of classification would be. So, an unsupervised learning algorithm tries to classify the given dataset into a certain number of groups in an optimum way.

**5.6.3 Semi-supervised Learning:**

If some learning samples are labeled, but some other are not labeled, then it is semi-supervised learning. It makes use of a large amount of unlabeled data for training and a small amount of labeled data for testing. Semi-supervised learning is applied in cases where it is expensive to acquire a fully labeled dataset while more practical to label a small subset.

**5.6.4 Reinforcement Learning:**

Here learning data gives feedback so that the system adjusts to dynamic conditions in order to achieve a certain objective. The system evaluates its performance based on the feedback responses and reacts accordingly.

**5.7 Challenges and Limitations of Machine learning:**

The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

**5.8 Application of Machine learning :**

1. Augmentation:

Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions.

1. Automation:

Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

1. Finance Industry:

Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

1. Government organization

The The government makes use of ML to manage public safety and utilities.Take the example of China with the massive face recognition.

1. Marketing

Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

* 1. **Why is Machine Learning important?**

Machine learning is the best tool so far to analyze, understand and identify a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can take decisions with minimal human intervention.

* 1. **Python:**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management.

It supports multiple programming paradigms, including object oriented, imperative, and functional and has a large and comprehensive standard library.

* 1. **python libraries for machine learning:** 
     1. **NumPy**:

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The core functionality of NumPy is its

"ndarray", for n-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

* + 1. **Pandas:**

Pandas is the most popular python library that is used for data analysis. It provides highly optimized performance with back-end source code is purely written in C or Python.

We can analyze data in pandas with Series and Data Frames.

* + 1. **Matplotlib:**

Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab. It can also be used with graphics toolkits like PyQt and wxPython.

* + 1. **Scikit-learn:**

Scikit-learn is a machine learning library for Python. It features several regression, classification and clustering algorithms including SVMs, gradient boosting, k-means, random forests and DBSCAN. Scikit is written in Python (most of it) and some of its core algorithms are written in Cython for even better performance. Scikit-learn is used to build models and it is not recommended to use it for reading, manipulating and summarizing data as there are better frameworks available for the purpose. It is open source and released under BSD license.

* + 1. **Seaborn:**

Seaborn is an open source, BSD-licensed Python library providing high level API for visualizing the data using Python programming language. Data can be visualized by representing it as plots which is easy to understand, explore and grasp. Such data helps in drawing the attention of key elements. To analyze a set of data using Python, we make use of Matplotlib, a widely implemented 2D plotting library. Likewise, Seaborn is a visualization library in Python. It is built on top of Matplotlib.

* 1. **Uses of Python:** 
     1. **Applications:**

Python can be used to develop different applications like web applications, graphic user interface based applications, software development application, scientific and numeric applications, network programming, Games and 3D applications and other business applications. It makes an interactive interface and easy development of applications.

* + 1. **Multiple Programming paradigms**:

Python is also used because of its providing continuous support to several programming paradigms. As it supports object-oriented programming and structured programming. Python has features, which also support various concepts of functional programming language. It is used for dynamic type system and automatic memory management. Python language features and programming paradigms allow you for developing the small as well as large applications. It can be used for complex software applications.

* + 1. **Robust Standard Library**:

Python has a large and robust standard library to use for developing the applications. It also makes the developers use Python over other languages. The standard library helps in using the different range of modules available for Python. As this module helps you in adding the functionality without writing any more code. To get the information about various modules, documentation on python standard library can be referred. While developing any web application, implementing web services, performing string operations and other usages like interface protocol, the standard library documentation helps.

* + 1. **Compatible with Major Platforms and Systems:**

Python is mainly compatible with major platforms and systems because of which it is used mainly for developing applications. With help of python interpreters, python code can be run on specific platforms and tools as it supports many operating systems. As python is an interpreted high-level programming language and it allows you to run the code on multiple platforms.

The new and modified code can be executed without recompiling and its impact can be monitored or checked. It means it’s not required to recompile the code after every change. This feature helps in saving the development time of the developers. 5 Access of Database:

Uses of Python also helps in accessing the database easily. Python helps in customizing the interfaces of different databases like MySQL, Oracle, Microsoft SQL Server, PostgreSQL, and other databases. It has an object database like Durus and ZODB.

It is used for standard database API and freely available for download.

1. **Code Readability:**

Python code is easy to read and maintained. It is easily reusable as well wherever it is required. Python’s having simple syntax, which allows the different concepts to develop without writing any additional code. The code should be of good quality and easy to maintain the source code and simplify the maintenance, which is required to develop the software application. It also emphasizes code readability, which is the great feature, unlike other programming languages. It helps in building custom applications and clean code helps in maintaining and updating the software applications without putting extra effort on the same code.

1. **Simplify Complex Software Development:**

Applications of Python is used to simplifying the complex software development process as it is a general-purpose programming language. It is used for developing the complex application like scientific and numeric application, and for both desktop and web applications. Python has features like analyzing data and visualization, which helps in creating custom solutions without putting extra effort and time. It helps you to visualize and present data in an effective way.

1. **Many Open Source Frameworks and Tools:**

Python is open source and easily available. This also helps in costing the software development significantly. There are many open source applications of python frameworks, libraries, and development tools for developing the application without putting extra cost.

Python frameworks simplify and make the process faster for web application development and the frameworks are Django, Flask, pyramid etc. Python GUI frameworks are available for developing the GUI based application.

1. **Adopt Test Driven Development:**

Python makes coding easier as well as testing with help of adopting Test Driven Development approach. The test cases can be easily written before any code development. Whenever the code development started, the written test cases can start testing the code simultaneously and provides the result. These can also be used for checking or testing the pre-requirements based on the source code.

* 1. **Other applications for which python is used:**

There are other applications for which python is used that are Robotics, web scraping, scripting, artificial intelligence, data analysis, machine learning, face detection, color detection, 3D CAD applications, console-based applications, audio-based applications, video-based applications, enterprise applications, and applications for Images etc. These are some major applications used.

**Convolutional Neural Network (CNN)**

**Convolutional Neural Networks (**CNNs) are a powerful tool for machine learning, especially in tasks related to computer vision. Convolutional Neural Networks, or CNNs, are a specialized class of neural networks designed to effectively process grid-like data, such as images.

**What is Convolutional Neural Network(CNN)?**

A [Convolutional Neural Network](https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/)(CNN) is a type of [deep learning algorithm](https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/) that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, [pooling layers](https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/), and fully connected layers. The architecture of CNNs is inspired by the visual processing in the human brain, and they are well-suited for capturing hierarchical patterns and spatial dependencies within images.

Key components of a Convolutional Neural Network include:

1. **Convolutional Layers:** These layers apply convolutional operations to input images, using filters (also known as kernels) to detect features such as edges, textures, and more complex patterns. Convolutional operations help preserve the spatial relationships between pixels.
2. **Pooling Layers:** Pooling layers downsample the spatial dimensions of the input, reducing the computational complexity and the number of parameters in the network. Max pooling is a common pooling operation, selecting the maximum value from a group of neighboring pixels.
3. [**Activation Functions**](https://www.geeksforgeeks.org/activation-functions/)**:** Non-linear activation functions, such as Rectified Linear Unit (ReLU), introduce non-linearity to the model, allowing it to learn more complex relationships in the data.
4. **Fully Connected Layers:** These layers are responsible for making predictions based on the high-level features learned by the previous layers. They connect every neuron in one layer to every neuron in the next layer.

CNNs are trained using a large dataset of labeled images, where the network learns to recognize patterns and features that are associated with specific objects or classes. Proven to be highly effective in [image-related tasks](https://www.geeksforgeeks.org/image-classifier-using-cnn/), achieving state-of-the-art performance in various computer vision applications. Their ability to automatically learn hierarchical representations of features makes them well-suited for tasks where the spatial relationships and patterns in the data are crucial for accurate predictions. CNNs are widely used in areas such as image classification, object detection, facial recognition, and medical image analysis.

The convolutional layers are the key component of a CNN, where filters are applied to the input image to extract features such as edges, textures, and shapes.

The output of the convolutional layers is then passed through pooling layers, which are used to down-sample the [feature maps](https://www.geeksforgeeks.org/feature-mapping/), reducing the spatial dimensions while retaining the most important information. The output of the pooling layers is then passed through one or more fully connected layers, which are used to make a prediction or classify the image.

**Convolutional Neural Network Design**

* The construction of a convolutional neural network is a [multi-layered feed-forward neural network](https://www.geeksforgeeks.org/multilayer-feed-forward-neural-network-in-data-mining/), made by assembling many unseen layers on top of each other in a particular order.
* It is the sequential design that give permission to CNN to learn hierarchical attributes.
* In CNN, some of them followed by grouping layers and hidden layers are typically convolutional layers followed by activation layers.
* The pre-processing needed in a ConvNet is kindred to that of the related pattern of neurons in the human brain and was motivated by the organization of the Visual Cortex.

**Convolutional Neural Network Training**

CNNs are trained using a supervised learning approach. This means that the CNN is given a set of labeled training images. The CNN then learns to map the input images to their correct labels.

The training process for a CNN involves the following steps:

1. **Data Preparation:** The training images are preprocessed to ensure that they are all in the same format and size.
2. **Loss Function:** A [loss function](https://www.geeksforgeeks.org/ml-common-loss-functions/) is used to measure how well the CNN is performing on the training data. The loss function is typically calculated by taking the difference between the predicted labels and the actual labels of the training images.
3. **Optimizer:** An optimizer is used to update the weights of the CNN in order to minimize the loss function.
4. **Backpropagation:** [Backpropagation](https://www.geeksforgeeks.org/backpropagation-in-machine-learning/) is a technique used to calculate the gradients of the loss function with respect to the weights of the CNN. The gradients are then used to update the weights of the CNN using the optimizer.

**CNN Evaluation**

After training, CNN can be evaluated on a held-out test set. A collection of pictures that the CNN has not seen during training makes up the test set. How well the CNN performs on the test set is a good predictor of how well it will function on actual data.

The efficiency of a CNN on picture categorization tasks can be evaluated using a variety of criteria. Among the most popular metrics are:

* **Accuracy:** Accuracy is the percentage of test images that the CNN correctly classifies.
* **Precision:** Precision is the percentage of test images that the CNN predicts as a particular class and that are actually of that class.
* **Recall:** Recall is the percentage of test images that are of a particular class and that the CNN predicts as that class.
* **F1 Score:** The F1 Score is a harmonic mean of precision and recall. It is a good metric for evaluating the performance of a CNN on classes that are imbalanced.

**Applications of CNN**

* **Image classification:** CNNs are the state-of-the-art models for image classification. They can be used to classify images into different categories, such as cats and dogs, cars and trucks, and flowers and animals.
* **Object detection:** CNNs can be used to detect objects in images, such as people, cars, and buildings. They can also be used to localize objects in images, which means that they can identify the location of an object in an image.
* **Image segmentation:** CNNs can be used to segment images, which means that they can identify and label different objects in an image. This is useful for applications such as medical imaging and robotics.
* **Video analysis:** CNNs can be used to analyze videos, such as tracking objects in a video or detecting events in a video. This is useful for applications such as video surveillance and traffic monitoring.

**Advantages of CNN**

* CNNs can achieve state-of-the-art accuracy on a variety of image recognition tasks, such as image classification, object detection, and image segmentation.
* CNNs can be very efficient, especially when implemented on specialized hardware such as GPUs.
* CNNs are relatively robust to noise and variations in the input data.
* CNNs can be adapted to a variety of different tasks by simply changing the architecture of the network.

**Disadvantages of CNN**

* CNNs can be complex and difficult to train, especially for large datasets.
* CNNs can require a lot of computational resources to train and deploy.
* CNNs require a large amount of labeled data to train.
* CNNs can be difficult to interpret, making it difficult to understand why they make the predictions they do.

**5.15 Sample Code:**

CHAPTER – 6

**TESTING**

**6.1 Introduction**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**6.2 Types of tests**

**1.** **Unit testing:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform sic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**2**. **Integration testing**:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**3.** **Functional test:**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

|  |  |  |
| --- | --- | --- |
| Valid Input | : | identified classes of valid input must be accepted. |
| Invalid Input | : | identified classes of invalid input must be rejected. |
| Functions | : | identified functions must be exercised. |
|  | : | identified classes of application outputs must be |

Output exercised.

**Systems/Procedures** : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**1. System Test:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**2. White Box Testing:**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**3. BlackBox Testing**:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.3 Test Cases:**

**6.4 Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

CHAPTER - 7

**SAMPLE SCREENS**

CHAPTER – 8

**CONCLUSION**

The conclusion of a project or paper titled *"Density-Based Smart Traffic Using Canny Edge Detection Method and CNN"* might summarize the key findings, the effectiveness of the proposed system, and potential future directions. Here's a draft for a conclusion:

In this study, we proposed a density-based smart traffic management system that leverages Canny edge detection and Convolutional Neural Networks (CNN) for real-time traffic monitoring and control. The Canny edge detection method was effectively used to detect traffic lane boundaries and vehicles in various traffic conditions, improving the accuracy of traffic flow detection. Combined with the deep learning capabilities of CNNs, the system was able to classify vehicle types and predict traffic density at intersections, enabling adaptive traffic signal control.

The results showed that the proposed method significantly improved traffic flow by reducing congestion and minimizing wait times, compared to traditional traffic management approaches. The system’s real-time capabilities demonstrated its practical application in dynamic urban traffic scenarios. Additionally, the use of CNN for vehicle detection and density prediction proved to be both scalable and reliable for implementation in smart cities.

Future work can focus on integrating additional sensors for better environmental sensing, optimizing the CNN model for faster processing, and exploring integration with IoT-based traffic infrastructure for even more efficient traffic management.

**CHAPTER - 9**

**FUTURE ENHANCEMENT**

The future enhancement of a **Density-Based Smart Traffic System** utilizing the Canny edge detection method with Convolutional Neural Networks (CNN) could involve integrating **real-time traffic flow prediction** through the use of temporal data and more sophisticated models. By leveraging temporal data from sensors, cameras, and historical traffic data, the system can not only detect traffic congestion but also predict patterns, enabling proactive traffic management. A CNN could be further trained to analyze patterns in traffic flow over time, adapting to daily, weekly, or seasonal fluctuations. This would allow for better signal optimization and efficient lane management, improving the overall traffic experience by preventing congestion before it happens, rather than just reacting to it.

Another key enhancement would be the **integration of multi-modal data sources** like traffic cameras, GPS data, and even crowd-sourced data from smartphones or vehicle tracking systems. The Canny edge detection method could be combined with other image processing techniques, such as object detection and motion tracking, to further enhance the system’s ability to detect not just vehicle density but also the types of vehicles on the road, their speeds, and their interactions with each other. By combining these inputs with CNNs trained on multi-modal datasets, the system could more accurately predict congestion spots, suggest alternate routes, and even dynamically adjust traffic signal timing based on real-time conditions, significantly improving the traffic management system's adaptability and efficiency.

**CHAPTER - 10**

**BIBLIOGRAPHY**

**Books Referred :**

* Object Oriented Software Engineering: using UML, patterns and java, brooch
* The UML user guide – Grady Brooch
* Think Python-Allen Downey, Green Tea Press.
* Python Programming- W.Chun, Pearson

**Journal Papers Referred:**

1.Ahmed S. Salama, Bahaa K. Saleh, Mohamad M.Eassa”Intelligent Cross Road Traffic Management System (ICRTMS)”2nd Int. Conf .on Computer Technology and Development, Cairo, Nov 2010, pp. 27-31.

2. Vikramaditya Dangi, Amol Parab, Kshitij Pawar, S.S Rathod, “Image Processing Based Intelligent Traffic Controller”.

3.Y. Wu, F. Lian, and T. Chang, “Traffic monitoring and vehicle tracking using roadside camera,” IEEE Int. Conf. On Robotics and Automation, Taipei, Oct 2006, pp. 4631–4636.

4. Jinglei, L. Zhengguang, and T. Univ, “A vision based road surveillance system using improved background subtraction and region growing approaches,” Eighth ACIS Int. Conf. On Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Qingdao, August 2007, pp. 819- 822.

5. M. Siyal, and J. Ahmed, “A novel morphological edge detection and window based approach for real-time road data control and management,” Fifth IEEE Int. Conference Information, Communications and signal Processing, Bangkok, July 2005, pp. 324-328.

6.T. Semertzidis, K. Dimitropoulos, A. Koutsia and N. Grammalidis, “Video sensor network for real-time traffic monitoring and surveillance, ”The Institution of Engineering and Technology, Volume 4, Issue 2, June 2010, pp. 103 – 112.

7.Lei Yang, Dewei Zhao and Xiaoyu Wu, “An improved Prewitt algorithm for edge detection based on noised image, ”in 4th International Congress on Image and Signal Processing, Shanghai, China, October 2011.

8.Weibin Rong, Zhanjing Li and Wei Zhang, “An improved canny edge detection algorithm,”in IEEE International Conference on Mechatronics and Automation, August 2014.

9.Wenshuo Gao, Xiaoguang Zhang and Lei Yang, “An improved Sobel edge detection,” in3rd International Conference on Computer Science and Information Technology, July 2010.

10.G.T. Shrivakshan and Dr.C. Chandrasekar, “A Comparison of various Edge Detection Techniques used in Image Processing,” International Journal of Computer Science Issues, Vol. 9, Issue 5, No 1, September 2012.

11. Badura, S.; Lieskovsky, A., "Intelligent Traffic System: Cooperation of MANET and Image Processing," Integrated Intelligent Computing (ICIIC), 2010 First International Conference on , vol., no., pp.119,123, 5-7 Aug. 2010

12. Yang Xiao; Yu-ming Xie; Lingyun Lu; Shuang Gao, "The traffic data compression and decompression for intelligent traffic systems based on two- dimensional wavelet

13. Naruemol Chumuang and Mahasak Ketcham “Handwriting Thai Signature Recognition System based on Multilayer Perceptron and Radial Basis Network” International Conference on Advanced Computational Technology and Creative Media (ICACTCM 2014) 14- 15 August, 2014.

14.N. Hnoohom, N. Chumuang and M. Ketcham, "Thai Handwritten Verification System on Documents for the Investigation," 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Bangkok, 2015, pp. 617- 622. doi: 10.1109/SITIS.2015.70

15. Mahasak Ketcham, Worawut Yimyam, Narumol Chumuang. “Segmentation of Overlapping Isan Dhamma Character on Palm Leaf Manuscript’s with Neural Network,” Recent Advances in Information and Communication Technology 2016, vol. 463 of the series Advances in Intelligent Systems and Computing, pp. 55-65, 2016. [10] P.G. Michalopoulos, "Vehicle Detection Video through Image Processing: The Auto scope System," IEEE Trans. Vehicular Technology”, vol. 40, no. 1, 1991, pp. 21–29.

16. Rafael C. Gonzalez, Richard E.Woods, “Digital Image Processing”, Addison- Wesley Publishing Company, September 1993

17.Rita Cucchiara , Costantino Grana , Massimo Piccardi , Andrea Prati , Stefano Sirotti , Improving Shadow Suppression in Moving Object Detection with HSV Color Information

18.Shu-Ching Chen, Mei-Ling Shyu, Chengcui Zhang, and Jeff Strickrott, “Spatio-Temporal Vehicle Tracking and Indexing ”, <http://www.cs.fiu.edu/~jstric01/>.

19. P. Khakham, N. Chumuang and M. Ketcham, "Isan Dhamma Handwritten Characters Recognition System by Using Functional Trees Classifier," 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Bangkok, 2015, pp. 606-612. doi: 10.1109/SITIS.2015.68

20. N. Chumuang and M. Ketcham, "Intelligent handwriting Thai Signature Recognition System based on artificial neuron network," TENCON 2014 - 2014 IEEE Region 10 Conference, Bangkok, 2014, pp. 1-6.doi: 10.1109/TENCON.2014.7022415