The Egyptian E-Learning University
Faculty of Computers and Information Technology

Final Project | Jun 2024

AUTO VISION: TRAFFIC SIGN DETECTION USING AI

Team

Aya zain Elabdeen Tawfeq	**2002052**
Ahmed Hamdy Gaber	**2000973**
Alhassan Mohamed Hassan	**1900859**
Haneen Hamdy Rabie	**2001821**
Hlaa Osama Mohamed	**2000160**
Yasmine Ehab Badr	**2000252**

Supervisor: Dr. Ahmed AbelReheem Advisor: Eng. Suhaila Moustafa

Table of Contents

Abstract

Automatic detection and recognition of traffic signs plays a crucial role in management of the traffic-sign inventory. It provides an accurate and timely way to manage traffic-sign inventory with a minimal human effort. In the computer vision community, the recognition and detection of traffic signs are a well-researched problem. A vast majority of existing approaches perform well on traffic signs needed for advanced driver-assistance and autonomous systems. However, this represents a relatively small number of all traffic signs (around 50 categories out of several hundred) and performance on the remaining set of traffic signs, which are required to eliminate the manual labor in traffic-sign inventory management, remains an open question. In this paper, we address the issue of detecting and recognizing a large number of traffic-sign categories suitable for automating traffic-sign inventory management. We adopt a Convolutional neural network (CNN) approach, the mask R-CNN, to address the full pipeline of detection and recognition with automatic end to-end learning. We propose several improvements that are evaluated on the detection of traffic signs and result in an improved overall performance. This approach is applied to detection of 200 traffic-sign categories represented in our novel data set. The results are reported on highly challenging traffic-sign categories that have not yet been considered in previous works. We provide comprehensive analysis of the deep learning method for the detection of traffic signs with a large intro-category appearance variation and show below 3% error rates with the proposed approach, which is sufficient for deployment in practical applications of the traffic-sign inventory management.

List of figures

Chapter 1
Introduction

Introduction

Proper management of traffic-sign inventory is an important task in ensuring safety and efficiency of the traffic flow. Most often this task is performed manually. Traffic signs are captured using a vehicle-mounted camera and manual localization and recognition is performed off-line by a human operator to check for consistency with the existing database. However, such manual work can be extremely time consuming when applied to thousands of kilometers of roads. Automating this task would significantly reduce the amount of manual work and improve safety through quicker detection of damaged or missing traffic signs. A crucial step towards the automation of this task is replacing manual localization and recognition of traffic signs with an automatic detection. In the computer-vision community the problem of traffic-sign recognition has already received a considerable attention, and excellent detection and recognition algorithms have already been proposed. But these solutions have been designed only for a small number of categories, mostly for traffic signs associated with advanced driver-assistance systems (ADAS) and autonomous vehicles. Detection and recognition of a large number of traffic-sign categories remains an open question. Various previous benchmarks have addressed the trafficsign recognition and detection task. However, several of them focused only on traffic-sign recognition (TSR) and ignored the much more complex problem of traffic-sign detection (TSD) where finding accurate location of traffic sign is needed. Other benchmarks that do address TSD mostly cover only a subset of traffic-sign categories, most often ones important for ADAS and autonomous vehicles applications. Most categories appearing in such benchmarks have a distinct appearance with low inter-category variance and can be detected using handcrafted detectors and classifiers. Such examples include round mandatory signs or triangular prohibitory signs.

Problem Statement

A problem statement for traffic sign detection could be:

"The goal is to develop an efficient computer vision system capable of accurately detecting and classifying traffic signs from images or video streams captured by on board cameras in vehicles. The system should be robust enough to recognize various types of traffic signs under different environmental conditions, including varying lighting, weather, and occlusion, to enhance road safety and facilitate autonomous driving."

the speed:

There is a direct relationship between the increase in average speed, the probability of accidents occurring,
and the severity of their consequences. As an example, every 1 km/h increase in average vehicle speed results in a 4%
increase in the risk of a fatal accident and a 3% increase in the risk of a serious accident.

Driving under the influence of alcohol and other psychoactive substances:

Driving under the influence of alcohol or any psychoactive substance or drug increases the risk of accidents that lead
to death or serious injury.

Not wearing motorcycle helmets, seat belts, or child restraints:
Wearing a helmet correctly can reduce the risk of a fatal accident by more than six times, and reduce the risk of a traumatic
brain injury by approximately 74%.
Driving in a distracted state of mind There are many types of distractions that can cause impaired driving:

Chapter 2
Background

2.1. Artificial intelligence:

Artificial Intelligence (AI) is a branch of computer science focused on creating systems capable of performing tasks that typically require human intelligence. These tasks include problem-solving, learning, understanding natural language, recognizing patterns, and making decisions. AI can be broadly categorized into narrow AI and general AI.

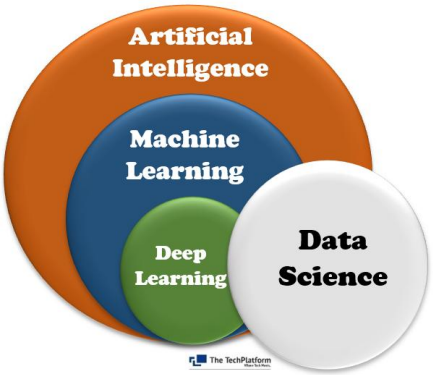

Fig. 2.1, Artificial intelligence [2]

2.1.1 Definition:

AI involves developing systems that exhibit intellectual processes similar to humans, such as reasoning, meaning discovery, generalization, and learning from experience. While computers can perform complex tasks (e.g., mathematical proofs, chess playing), they still lack the full flexibility and everyday knowledge of humans. to say that a program thinks like a human, we need to figure out how humans think. To comprehend how humans think, we must delve into the inner workings of the human mind.

2.1.2 Applications:

AI is used in diverse areas:

- Medical diagnosis: AI systems assist doctors in diagnosing diseases.
- Search engines: AI powers search algorithms to retrieve relevant information.
- Voice and handwriting recognition: AI enables accurate transcription.
- Chatbots: AI-driven conversational agents interact with users.

2.1.3 Components of AI:

- Learning: AI systems can learn from data and improve their performance over time.
- Reasoning: AI can make logical deductions and draw conclusions.
- Problem Solving: AI tackles complex problems by analyzing information.
- Perception: AI processes visual or sensory input (e.g., image recognition).

- Language: AI understands and generates human language

2.1.4 Personal use:

- **Virtual Assistants**:
  - **Examples**: Siri, Alexa, Google Assistant.
  - **Impact**: These AI-powered assistants help manage daily tasks such as setting reminders, playing music, controlling smart home devices, and providing weather updates through voice commands.
- **Smart Home Devices**:
  - **Examples**: Smart thermostats (like Nest), smart lights, robotic vacuum cleaners.
  - **Impact**: AI enhances home automation, optimizing energy use, improving security through smart cameras, and performing routine cleaning tasks autonomously.

2.1.5 The concept of Artificial intelligence:

Artificial intelligence (AI) is the theory and development of computer systems capable of performing tasks that historically required human intelligence, such as recognizing speech, making decisions, and identifying patterns. AI is an umbrella term that encompasses a wide variety of technologies, including machine learning, deep learning, and natural language processing (NLP).

Although the term is commonly used to describe a range of different technologies in use today, many disagree on whether these actually constitute artificial intelligence. Instead, some argue that much of the technology used in the real world today actually constitutes highly advanced machine learning that is simply a first step towards true artificial intelligence, or "general artificial intelligence" (GAI).

Yet, despite the many philosophical disagreements over whether "true" intelligent machines actually exist, when most people use the term AI today, they're referring

to a suite of machine learning-powered technologies, such as Chat GPT or computer vision, that enable machines to perform tasks that previously only humans can do like generating written content, steering a car, or analyzing data.

2.1.6 Example of Artificial intelligence:

of the most common examples of AI in use today include:

ChatGPT: Uses large language models (LLMs) to generate text in response to questions or comments posed to it.

Google Translate: Uses deep learning algorithms to translate text from one language to another.

Netflix: Uses machine learning algorithms to create personalized recommendation engines for users based on their previous viewing history.

Tesla: Uses computer vision to power self-driving features on their cars.

2.1.7 Levels of Artificial intelligence:

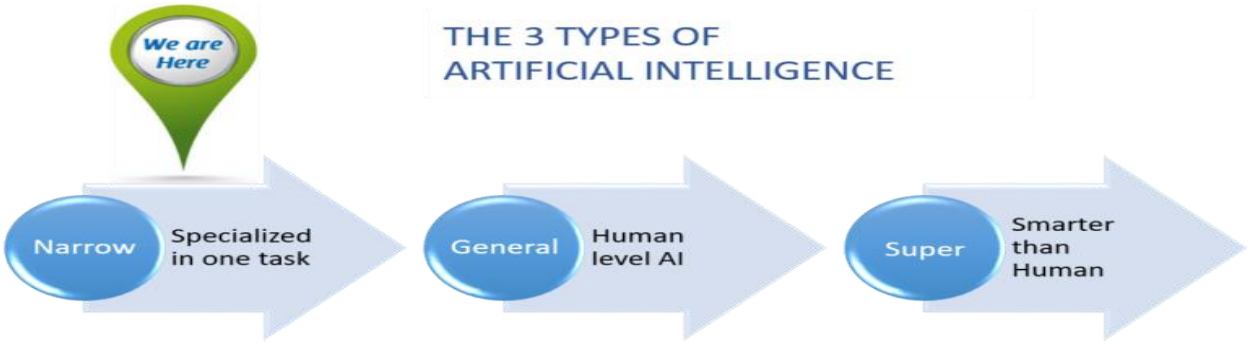

Fig. 2.2, Types of AI [3]

2.1.7.1 Narrow AI:

• **Definition:**

Narrow AI refers to AI systems that are designed to perform specific tasks or a set of closely related tasks. Unlike General AI, which aims to replicate human intelligence and perform any intellectual task that a human can, Narrow AI is **limited in scope**. It operates under a per-defined set of rules and cannot exhibit the same level of understanding or adaptability as a human.

• **Capabilities of Narrow AI**:

Within their domain, Narrow AI systems demonstrate impressive capabilities:

Data Analysis: They can analyze large datasets more quickly and accurately than humans.

Pattern Recognition: Narrow AI identifies patterns and trends.

Predictions and Decisions: It makes data-driven predictions or decisions.

Learning and Improvement: Techniques like machine learning, particularly deep learning, allow Narrow AI to adjust its algorithms based on processed data.

However, these capabilities are bounded by predefined functions. Narrow AI lacks the ability to think abstractly, understand context beyond its programmed domain, or exhibit emotional intelligence.

- **Applications of Narrow AI**:

Virtual Assistants: AI-powered virtual assistants like Siri, Alexa, and Google Assistant perform tasks such as setting reminders, playing music, or providing weather updates.

Recommendation Systems: Streaming services like Netflix and Spotify use Narrow AI to analyze user preferences and provide personalized content recommendations.

Autonomous Vehicles: Self-driving cars interpret sensory data using AI and make real-time decisions on the road.

Health care: AI algorithms assist in diagnosing diseases, analyzing medical images, and predicting patient outcomes.

Finance: AI is used for algorithmic trading, fraud detection, and customer service automation in the banking sector.

Manufacturing: AI optimizes production processes, improves supply chain management, and enhances quality control.

Implications for the Future:

While Narrow AI excels in specific tasks, achieving General AI remains a distant goal. Ethical considerations, data quality, and explainability are crucial as AI continues to evolve.

In summary, Narrow AI is the most common form of AI encountered today, proficiently performing singular tasks while lacking consciousness and the ability to generalize beyond its programming.

2.1.7.2 General AI:

Definition:

AGI is a theoretical form of AI that aims to emulate human intelligence across a wide range of cognitive tasks.

Key Distinctions:

Scope of Tasks: General AI: Can perform any intellectual task that a human can.

Learning and Adaptation: General AI: Learns from experience and applies knowledge to unfamiliar scenarios.

Flexibility: General AI: Exhibits adaptability across diverse domains, akin to human intelligence.

2.1.7.3 Super AI:

Super AI, also known as Artificial Super intelligence (ASI)**,** represents the pinnacle of artificial intelligence. It's a theoretical form of AI that surpasses human intelligence in every aspect. While it doesn't yet exist, many researchers believe it has the potential to be smarter than the human brain itself . Imagine machines with self-awareness, abstract thought, and interpretive abilities beyond human comprehension. These super intelligent entities would redefine what we think of as intelligence.

Characteristics of Super AI:

Cognitive Abilities: Super AI possesses cognitive abilities similar to human intelligence. It can understand, learn, and apply knowledge across various domains. In contrast, narrow AI is designed for specific tasks and lacks the broader cognitive capabilities of super AI .

Self-Awareness: Unlike current AI systems, super AI would be aware of its own existence and purpose.

Abstract Thought: It can engage in complex reasoning, abstraction, and creative problem-solving.

Beyond Human Comprehension: Super AI's intellectual scope would extend far beyond what humans can grasp.

Potential Applications and Challenges:

Financial Services**:** Super AI could enhance transaction data, improve KYC standards, automate interbank settlements, and more—all without requiring humans to code [2].

Insurance: Streamlining claims processing, automating damage assessment, and improving customer service quality checks are just a few areas where super AI could revolutionize the insurance industry.

Healthcare, Research, and More: From drug discovery to scientific breakthroughs, super AI could accelerate progress across various fields.

Ethical Considerations and Future Prospects**:** As we explore the possibilities of super AI, we must address ethical concerns, safety precautions, and the impact on society. The journey toward super AI is both exciting and challenging. It remains a topic of intense research and speculation.

In summary, super AI represents a future where machines transcend human intelligence, opening limitless possibilities and raising profound questions about our place in the universe.

2.2. Machine learning:

Machine learning (ML) is a fascinating field within artificial intelligence (AI). It focuses on developing and studying statistical algorithms that can learn from data and generalize to unseen situations, enabling them to perform tasks without explicit instructions. Here are some key points about machine learning:

1.Learning from Data: ML algorithms learn patterns and relationships from training data**.** They adapt and improve their performance as they encounter more examples. Unlike traditional rule-based systems, ML models don't rely on explicit programming; they learn directly from data.

2.Generalization: ML models aim to generalize their knowledge to new, unseen data. This means they can make predictions or classifications beyond the training examples. Generalization helps ML systems handle real-world variability and noise.

3.Applications:

Natural Language Processing (NLP): ML powers chatbots, language translation, sentiment analysis, and more.

Computer Vision: ML algorithms recognize objects, faces, and scenes in images and videos.

Speech Recognition: ML enables voice assistants and transcription services.

Predictive Analytic s: In business, ML is used for forecasting, recommendation systems, and fraud detection.

4. Mathematical Foundations:

Optimization: ML models are optimized using mathematical techniques to minimize errors or maximize performance.

Statistical Methods: ML draws heavily from computational statistics.

2.2.1. Types of machine learning:

Supervised Learning, Unsupervised Learning, Reinforcement Learning.

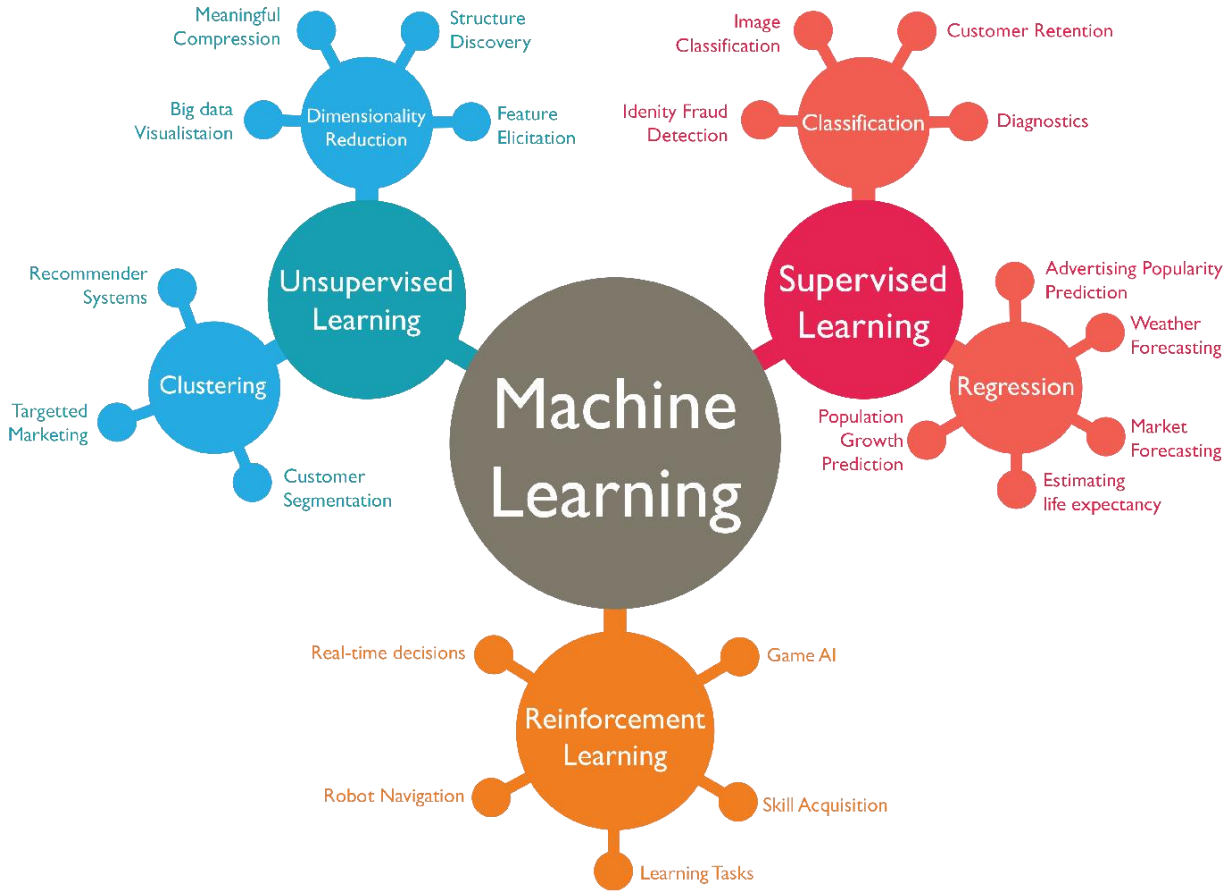

Fig. 2.3, Machine learning types [4]

2.2.1.1. Supervised Learning:

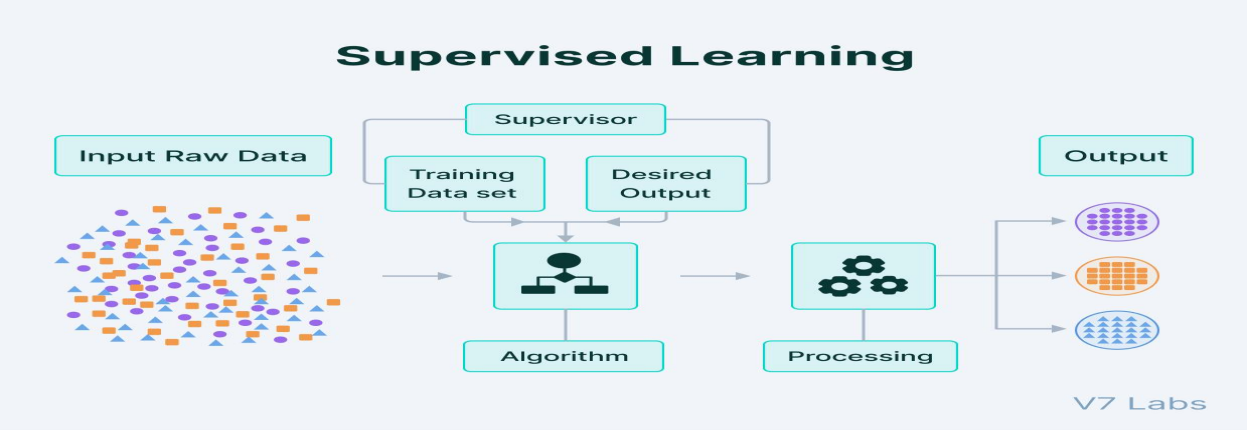

Fig. 2.4, Supervised learning [5]

Supervised learning is a fundamental technique in machine learning where labeled data sets are used to train algorithms. it involves using input data (such as a vector of predictor variables) along with corresponding desired output values (also known as human-labeled supervisory signals) to train a model. The training data is processed, allowing the model to learn a function that maps new data to expected output values. In supervised learning, the training data includes both inputs and outputs, and the goal is to learn a function that can map inputs to outputs. For example, a supervised learning algorithm can be trained to recognize handwritten digits by using a data set of images of digits and their corresponding labels. The algorithm learns to map each image to the correct digit label, based on the patterns in the training data. Once the algorithm has learned this mapping, it can be used to predict the label of new images that were not included in the training data.

Components of Supervised learning:

- **Classification**
- **regression**

2.2.1.2. Unsupervised Learning:

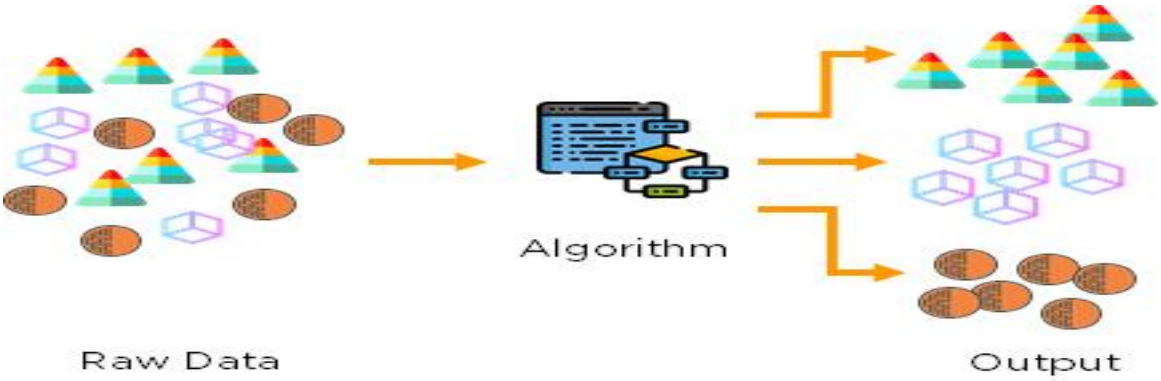

Fig. 2.5, Unsupervised learning [6]

Unsupervised learning is a fascinating branch of machine learning where algorithms explore and analyze unlabeled data sets without any human-provided guidance. the focus is on discovering patterns, structures, and relationships within data. It's a type of learning where the training data consists only of inputs, and the goal is to discover patterns or structures in the data. In unsupervised learning, the algorithm does not receive any feedback or labels about the correctness of its output. Instead, it tries to find meaningful patterns in the data on its own. For example, an unsupervised learning algorithm can be used to cluster customers based on their purchasing behavior. The algorithm would group similar customers together, without any prior knowledge of what the groups should be.

Components of Unsupervised Learning:

- **clustering**
- **Dimensionality Reduction**

2.2.1.3. Reinforcement Learning:

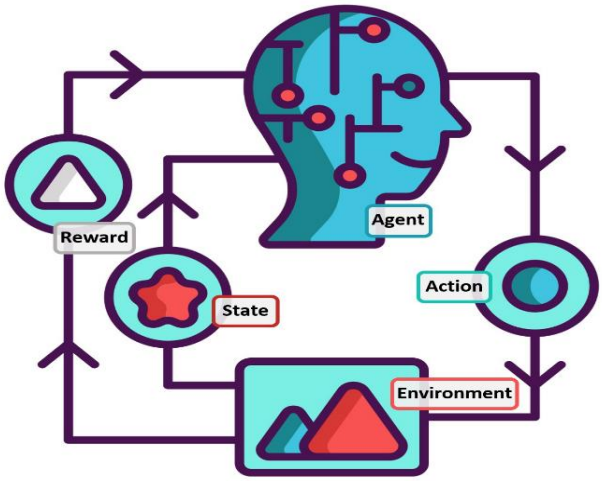

Fig. 2.6, Reinforcement learning [7]

Reinforcement learning (RL) is a fascinating area of machine learning that focuses on decision-making by autonomous agents. RL addresses how an intelligent agent should take actions in a dynamic environment to maximize cumulative rewards.

• **Components of RL**:

- **Agent**: The learner or decision-maker interacting with the environment.
- **Environment**: The external system with which the agent interacts.
- **State**: A representation of the environment at a given time.
- **Action**: Choices made by the agent to influence the environment.
- **Reward**: Feedback received after taking an action.
- **Policy**: The strategy or rule guiding the agent's actions.
- **Value Function**: Estimates the expected cumulative reward from a given state or action.
- **Model**: A representation of the environment's dynamics (optional)[1].

2.2.2. Data Preparation:

Data preparation is a crucial step in machine learning that involves cleaning, transforming, and organizing raw data to create a suitable dataset for training models.

2.2.2.1. Steps of Data Preparation:

a. Data Collection:

Gather raw data from various sources (databases, APIs, files, etc.). Ensure data quality by addressing missing values, outliers, and inconsistencies.

b. Data Transformation:

Numerical and Categorical Data: Convert categorical variables (like gender or city) into numerical representations (one-hot encoding, label encoding). Normalize numerical features (scaling to a common range).

Feature Engineering: Create new features from existing ones (e.g., extracting day of the week from a timestamp). Select relevant features based on domain knowledge.

Data Splitting: Divide the dataset into training, validation, and test sets.

c. Data Cleaning:

Address missing values (impute or remove them).

Detect and handle outliers.

Remove duplicate records.

d. **Data Integration**:

Combine data from multiple sources (if needed).

Ensure consistency in formats and units.

e. Data Reduction:

Reduce dimensionality (using techniques like PCA) to improve efficiency.

Select relevant features to avoid noise.

f. Data Formatting:

Ensure consistent data formats (dates, currencies, etc.).

2.2.2.2. Examples of Data Preparation:

- **Google Translate**:

Google Translate improved translation quality by selecting the best subset of training data rather than tuning models. Quality data played a crucial role in their success.

- **Google Brain's Diabetic Retinopathy Project**:

Instead of tweaking models, they created a dataset of 120,000 labeled examples. The quality of the data led to successful disease detection.

2.3. Deep Learning:

Deep learning is a subset of machine learning that harnesses the power of multi-layered neural networks, known as deep neural networks (DNNs). These networks simulate the intricate decision-making capabilities of the human brain.Its market size is expected to reach $29.8 billion by 2025, showing its growing importance in AI. Deep learning has a rich history, with significant advancements in the 2000s. It has driven breakthroughs in computer vision, natural language processing, and speech recognition. Models like CNNs excel in image tasks, while RNNs shine in language and speech tasks. Deep learning has the potential to transform technology and solve complex problems as data continues to grow.[16]

2.3.1. Deep Neural Network:

- A DNN consists of three or more layers (though most practical DNNs have many more).
- These networks are trained on large datasets to:
  - Identify and classify phenomena.
  - Recognize patterns and relationships.
  - Evaluate possibilities.
  - Make predictions and decisions.
- While a single-layer neural network can provide approximate predictions, the additional layers in a DNN refine and optimize outcomes for greater accuracy.

2.3.2. Applications and Impact:

- Deep learning powers numerous applications and services, including:

  Digital Assistants: Think Siri, Alexa, or Google Assistant.

  Voice-Enabled TV Remotes: Changing channels with voice commands.

  Credit Card Fraud Detection: Identifying suspicious transactions.

  Self-Driving Cars: Navigating complex environments.

Generative AI: Creating art, music, and text.

- It's also behind emerging technologies like self-driving cars and advanced robotics.

2.3.3. Deep Learning vs. Machine Learning:

- **Data Type**:
  - Machine learning relies on structured, labeled data for predictions.
  - Deep learning handles unstructured data (like text and images) without extensive pre-processing.
- **Feature Extraction**:
  - In machine learning, human experts manually define feature hierarchies.
  - Deep learning automates feature extraction, reducing reliance on experts.
- **Learning Types**:
  - Both machine learning and deep learning support supervised, unsupervised, and reinforcement learning.

In summary, deep learning empowers AI systems to learn complex patterns directly from raw data, making it a driving force behind modern AI applications[123].

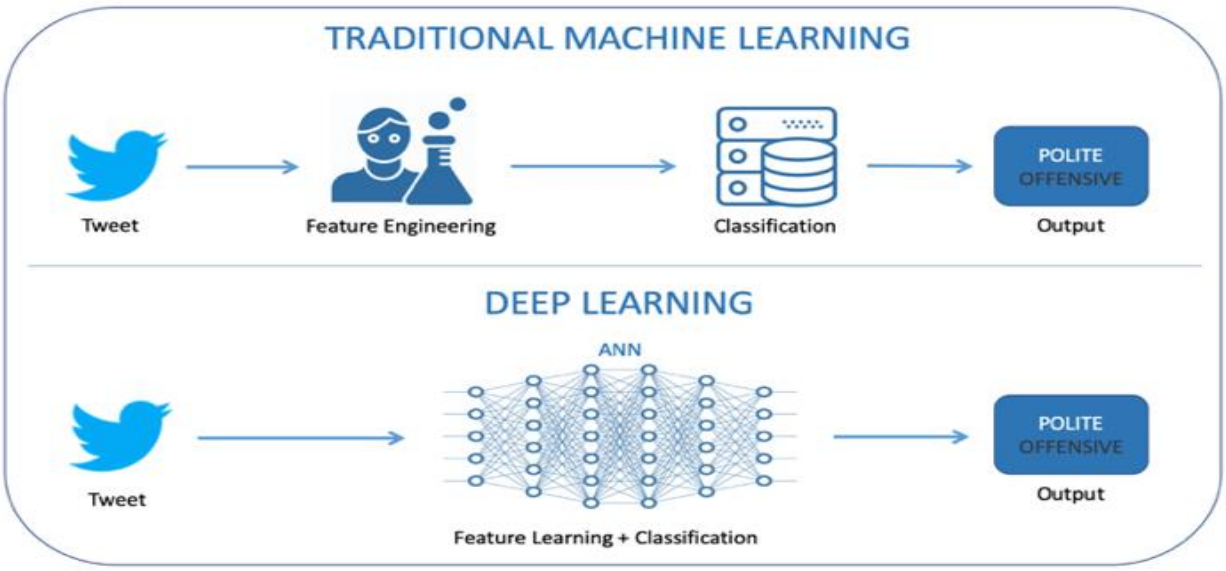

Fig. 2.7, ML vs DL [8]

2.3.4. Deep Learning Algorithms:

Deep learning algorithms are at the forefront of artificial intelligence and have revolutionized various domains. These algorithms have revolutionized many fields, including computer vision, natural language processing, and speech recognition.

Let's explore some essential deep learning algorithms:

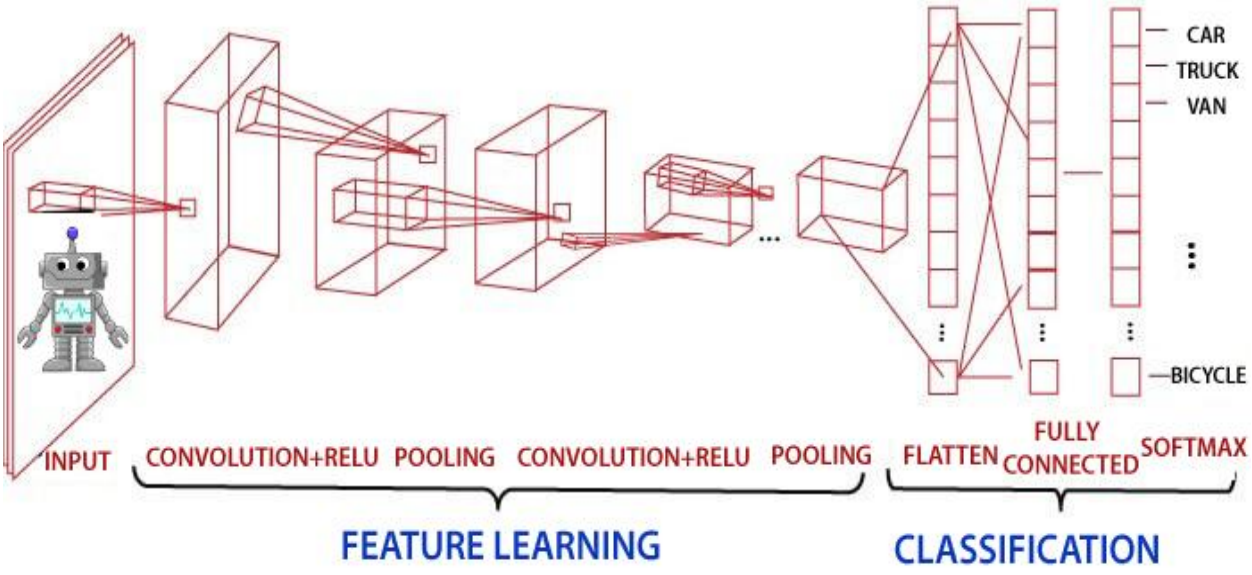

Fig. 2.8, Convolutional Neural Network (CNN) [9]

2.4. Convolutional Neural Network (CNN):

a **convolutional neural network** (**CNN/ConvNet**) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network, we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other. CNN is designed to learn spatial hierarchies of features automatically and adaptively through backpropagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers. This review article offers a perspective on the basic concepts of CNN and its application to various radiological tasks and discusses its challenges and future directions in the field of radiology. Two challenges in applying CNN to radiological tasks, small dataset, and overfitting, will also be covered in this article, as well as techniques to minimize them. Being familiar with the concepts and advantages, as well as limitations, of CNN is essential to leverage its potential in diagnostic radiology, with the goal of augmenting the performance of radiologists and improving patient care.

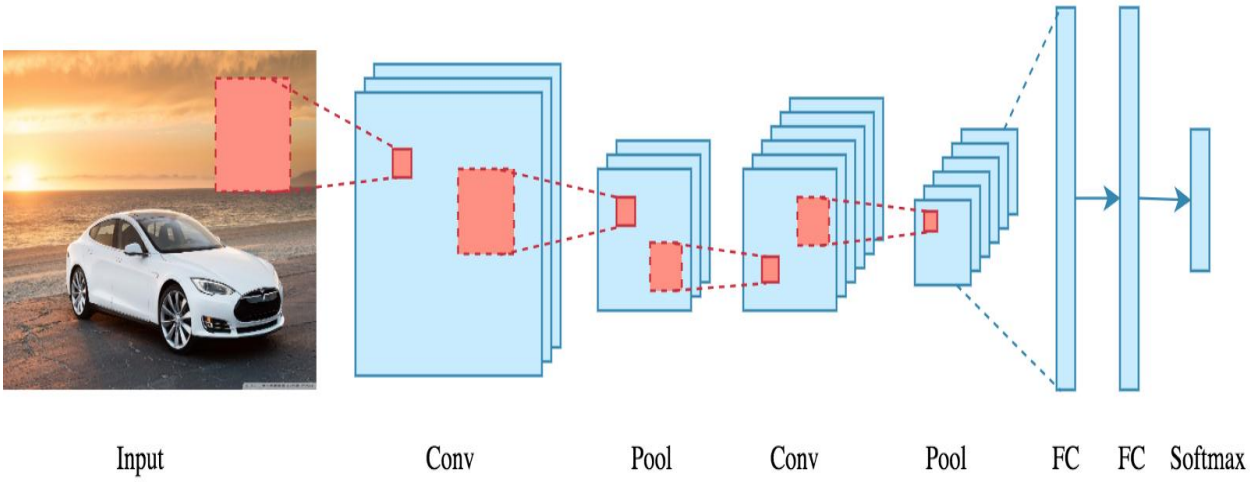

Fig. 2.9, Convolutional Neural Network (CNN) [10]

2.4.1. Types of CNN:

There are two types of CNN the first is:

2.4.1.1. CNN One Dimensional:

1-D CNN can perform activity recognition task from accelerometer data, such as if the person is standing, walking, jumping etc. This data has 2 dimensions. The first dimension is time-steps and other is the values of the acceleration in 3 axes. Following plot illustrate how the kernel will move on accelerometer data.

Human activity recognition is the problem of classifying sequences of accelerometer data recorded by specialized harnesses or smart phones into known well-defined movements. Classical approaches to the problem involve hand crafting features from the time series data based on fixed-sized windows and training machine learning models, such as ensembles of decision trees. The difficulty is that this feature engineering requires deep expertise in the field.

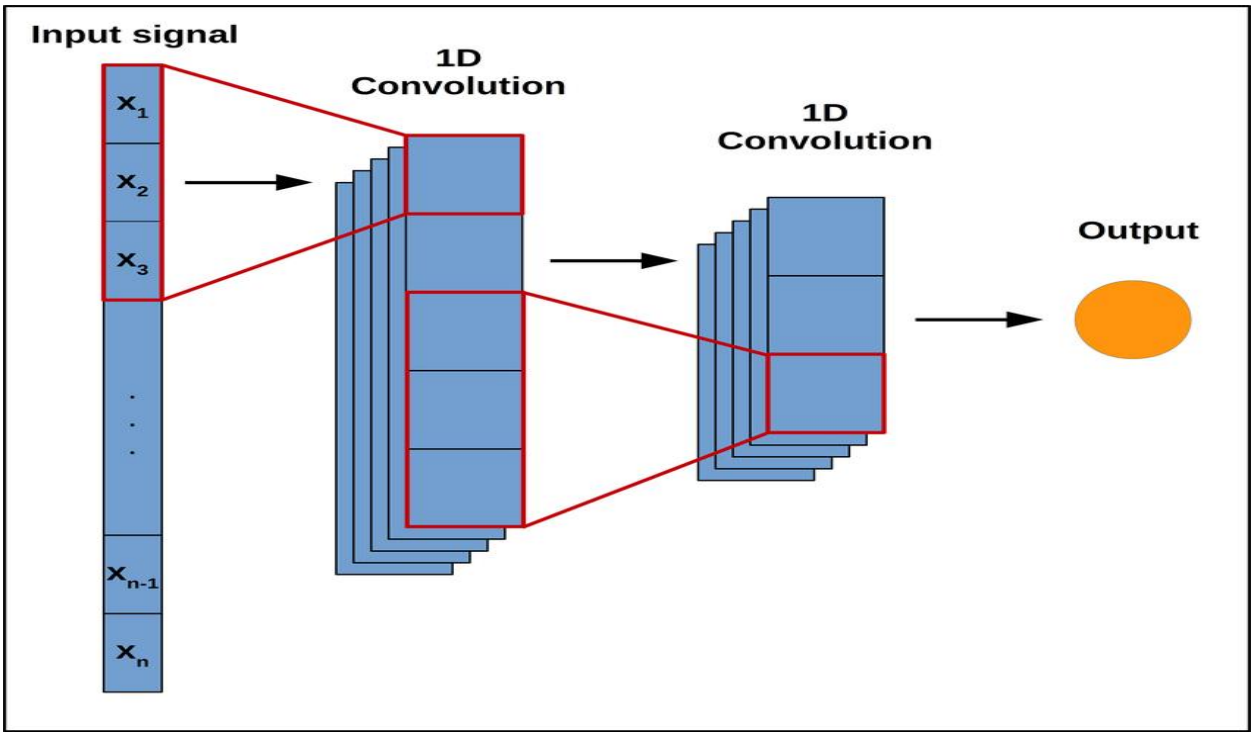

Fig. 2.10, 1D-CNN architecture [11]

2.4.1.2 CNN Two Dimensional:

It is called 2 dimensional CNN because the kernel slides along 2 dimensions on the data as shown in the following image. The whole advantage of using CNN is that it can extract the spatial features from the data using its kernel, which other networks are unable to do. We perform several stages, the first of which is to identify the inputs, and the second is to divide the images and perform operations on them to fully understand them and divide them into the simplest images for them. Then we purify them and produce the best output for training the data.

2.4.2. Layers of CNN:

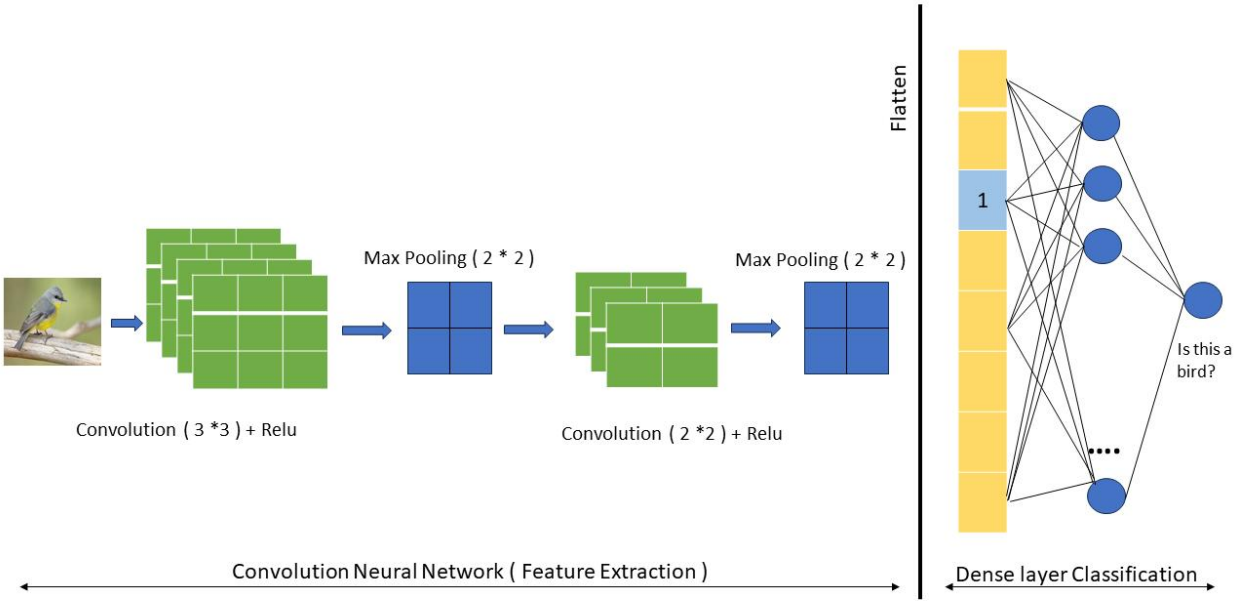

Fig. 2.11, CNN Architecture [12]

1. Convolution Layer
2. Pooling Layer
3. Fully Connected Layer
4. Dropout
5. Activation Function

2.4.2.1 Convolution Layer:

Convolutional layers are the backbone of CNNs, enabling automatic feature extraction from images. They revolutionized computer vision and continue to drive breakthroughs in various fields. play a pivotal role in convolutional neural networks (CNNs), especially when dealing with two-dimensional image data. These layers are responsible for extracting meaningful features from input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM). The convolution layer in CNN passes the result to the next layer once applying the convolution operation in the input. Convolutional layers in CNN benefit a lot as they ensure the spatial relationship between the pixels is intact.

Fig. 2.12, convolution layer [13]

2.4.2.2 pooling Layer:

Max pooling is a crucial operation commonly used in CNNs to reduce the spatial dimensions of an input volume. Max pooling is a non-linear down sampling technique. It makes the representation smaller, more manageable, and reduces the number of parameters and computation in the network. The goa l is to retain the most important features while discarding less relevant information. This is

performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations. It basically summarizes the features generated by a convolution layer. In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling.

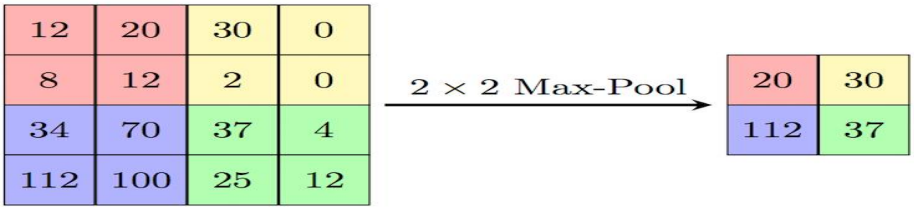

Fig. 2.13, max pooling layer [14]

2.4.2.3 Fully Connected Layer:

Fully connected layers, also known as **dense layers**, are fundamental components.in neural networks. In other words, all possible connections between input and output nodes exist. These layers play a crucial role in capturing complex relationships and making predictions. The flattened vector then undergoes few more FC layers where the mathematical functions operations usually take place. In this stage, the classification process begins to take place. The reason two layers are connected is that two fully connected layers will perform better than a single connected layer. These layers in CNN reduce the human supervision.

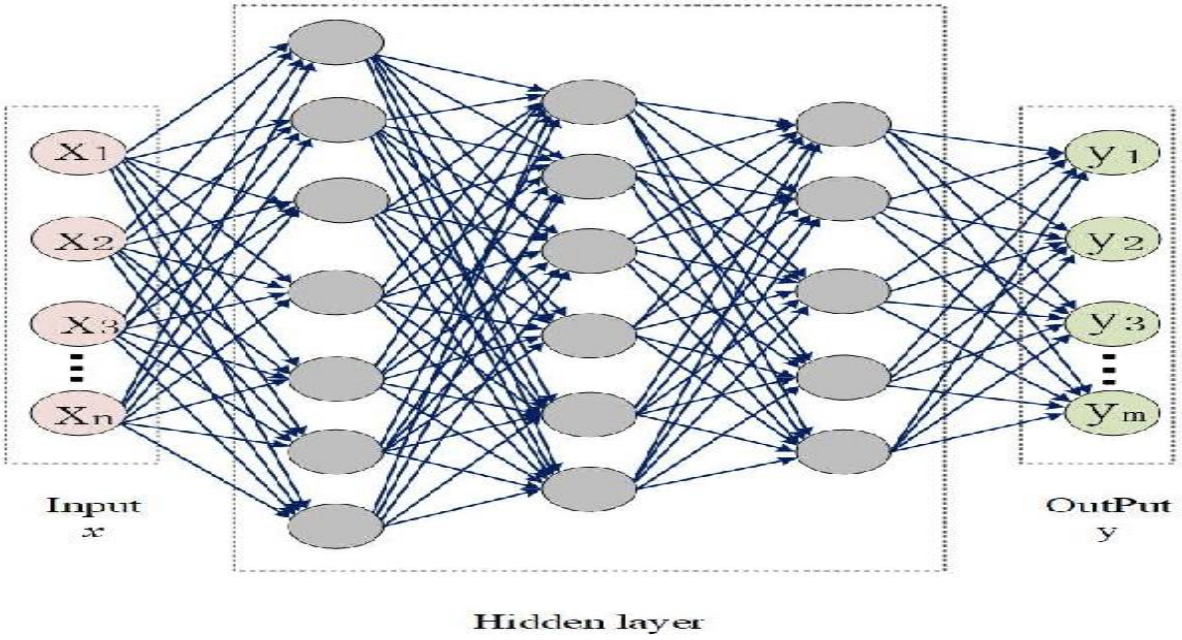

Fig. 2.14, fully connected layer [15]

2.4.2.4 Dropout:

when all the features are connected to the FC layer, it can cause overfitting in the training dataset. Overfitting occurs when a particular model works so well on the training data causing a negative impact in the model's performance when used on a new data. To overcome this problem, a dropout layer is utilized wherein a few neurons are dropped from the neural network during training process resulting in reduced size of the model. On passing a dropout of 0.3, 30% of the nodes are dropped out randomly from the neural network Dropout results in improving the performance of a machine learning model as it prevents overfitting by making the network simpler. It drops neurons from the neural networks during training.

2.4.2.5 Activation Functions:

one of the most important parameters of the CNN model is the activation function. They are used to learn and approximate any kind of continuous and complex relationship between variables of the network. In simple words, it decides which

information of the model should fire in the forward direction and which ones should not at the end of the network. It adds non-linearity to the network. There are several commonly used activation functions such as the ReLU, softmax, tanH and the Sigmoid functions. Each of these functions have a specific usage. For a binary classification CNN model, sigmoid and softmax functions are preferred for a multi-class classification, generally softmax us used. In simple terms, activation functions in a CNN model determine whether a neuron should be activated or not. It decides whether the input to the work is important or not to predict using mathematical operations.

2.4.3 Sigmoid or Logistic Activation Function:

The Sigmoid Function curve looks like a S-shape.

The sigmoid activation function is often used in binary classification problems. It maps the input to a range between 0 and 1, which can be interpreted as a probability. The sigmoid function is defined as follows:

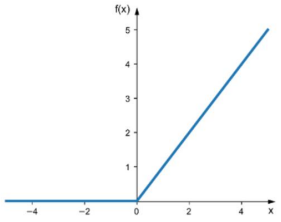

Fig. 2.15, RelU diagram [16]

$$f(x) = \frac{1}{(1 + e^{-x})}$$

2.4.4 Tanh or hyperbolic tangent Activation Function:

tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped). The hyperbolic tangent activation function is commonly used in neural networks. It maps the input to a range between -1 and 1.

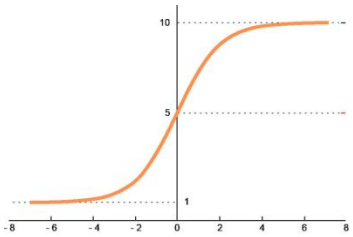

Fig. 2.16, SoftMax diagram [17]

The tanh function is defined as follows:

$$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$$

2.4.5 SoftMax:

The SoftMax activation function is typically used in multi-class classification problems. It normalizes the output of a network into a probability distribution over multiple class. The SoftMax function is defined as follows: Given an input vector x = [x_1, x_2, ..., xn], the softmax function outputs a vector y = [y_1, y_2, ..., yn] such that: (for all i)

$$y_i = \frac{e^{x_i}}{(\sum_j e^{x_j})}$$

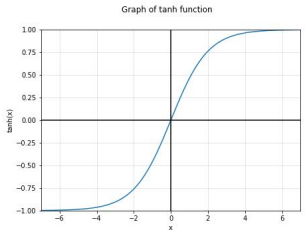

Fig. 2.17, TanH diagram [18]

2.4.6 ReLU (Rectified Linear Unit):

The ReLU activation function is defined as follows:

$$f(x) = \max(0, x)$$

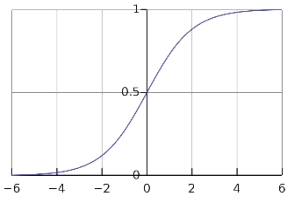

Fig. 2.18, Sigmoid diagram [19]

2.5. Visual Geometry Group 16 network (VGG16):

VGG-16, short for Visual Geometry Group 16**,** is a convolutional neural network (CNN) architecture that gained prominence in the field of computer vision. Developed by the Visual Geometry Group at the University of Oxford, VGG-16 was introduced in 2014.It has become one of the most popular deep learning models for image recognition.

2.5.1 Architecture:

- VGG-16 is characterized by its depth and simplicity.
- It consists of **16 layers**, including:
  - **13 convolutional layers**: These layers perform feature extraction.
  - **3 fully connected layers**: These layers make predictions based on the extracted features.

2.5.2 Effectiveness:

- Despite its straightforward design, VGG-16 achieves strong performance on various computer vision tasks.
- It excels in tasks such as image classification and object recognition.

2.5.3 Hierarchical Feature Extraction:

- VGG-16's architecture includes a stack of convolutional layers followed by max-pooling layers.
- This design enables the model to learn intricate hierarchical representations of visual features.
- The depth allows it to capture both low-level and high-level features

In summary, VGG-16's simplicity, effectiveness, and ability to learn intricate features make it a versatile choice for various deep learning applications.

2.6. Visual Geometry Group 19 network (VGG19):

VGG-19, short for Visual Geometry Group 19, is a convolutional neural network (CNN) model developed by the Visual Geometry Group at the University of Oxford**.** It is an extension of the earlier VGG16 model and is known for its simplicity and effectiveness in image classification tasks.

2.6.1 Architecture:

- VGG-19 is an extension of the earlier VGG-16 model.
- It consists of 19 layers, including 16 convolutional layers and 3 fully connected layers.
- The architecture is characterized by its **depth** and simplicity.

2.6.2 Hierarchical Feature Extraction:

- Similar to VGG-16, VGG-19 stacks convolutional layers followed by max-pooling layers.
- This design enables the model to learn intricate hierarchical representations of visual features.
- The depth allows it to capture both **low-level** and **high-level** features.

2.6.3 Image Recognition and Classification:

- VGG-19 excels in tasks such as image classification and object recognition.
- Given an input image (usually 224x224 pixels with RGB channels), VGG-19 outputs a vector of 1000 values representing classification probabilities for corresponding classes.

- It was a top performer in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC).

In summary, VGG-19's depth, feature extraction capabilities, and transfer learning potential make it a valuable tool in the world of deep learning.

2.7. Empowering Cross-Platform Development:

In today's rapidly evolving digital landscape, the demand for mobile applications has skyrocketed. To meet this demand efficiently, developers are turning to cross-platform development, a revolutionary approach that enables the creation of applications compatible with multiple operating systems. Among the tools driving this transformation is the Dart programming language and the Flutter framework. This essay explores the benefits of cross-platform development and discusses how Dart and Flutter empower developers to build robust, visually appealing, and highly performant mobile applications.

2.7.1 Understanding Cross-Platform Development:

Cross-platform development involves building applications that can run seamlessly on multiple platforms, such as iOS and Android. Traditionally, developers had to build separate native applications for each platform, resulting in increased development time, effort, and maintenance costs. However, with cross-platform development, developers can leverage a single codebase to create applications that function across various platforms, saving time and resources.

2.7.2 The Dart Programming Language:

Dart is an open-source general-purpose programming language developed by Google. It is designed to be approachable, productive, and versatile for building high-quality applications across various platforms. Its modern features, such as asynchronous programming and garbage collection, contribute to enhanced performance and smooth execution.

2.7.3 The Flutter Framework:

Flutter is an **open-source framework** developed by Google for building beautiful, natively compiled, multi-platform applications from a single codebase.Flutter

allows developers to create visually stunning and highly responsive applications with a native-like user experience. Its rich set of pre-built widgets simplifies the UI development process, ensuring consistency across platforms. Flutter's "hot reload" feature enables developers to instantly view code changes, making the development and debugging process more efficient.[30]

Chapter 3

Literature Review

Abstract

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system. The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process.

A. T. Guofeng, "Traffic Sign Recognition Based on SVM and Convolutional Neural Network". [1]

The technique in this research was broken down into three stages: a) HSV colour space conversion b) Localizing traffic signs using HOG and SVM; c) classifying using enhanced CNN. Conversion to HSV colour space is the first step since this colour model performs better under varying lighting conditions and is more in line with how humans understand data [2]. By adjusting the threshold values of the H and S components, the red, yellow, and blue are transformed into binary pictures. This makes it easier to obtain ROIs for the image by eliminating extraneous noise. The HOG feature vector of the ROI is extracted in the following phase and sent to the SVM classifier, which will determine if it is a traffic sign. In this case, they employed a histogram of nine-bin-oriented gradients, which implies that the gradient direction of thecell, which is 360o, is divided into nine orientations. HOG of the cell is obtained by utilising weights of gradient direction in the histogram. Later, each cell unit is joined in series to form a large, continuous block to get a 9-dimensional feature vector of thecorresponding cell. By doing this, the HOG feature vector of ROI is derived, which further categorises it as a probable traffic sign.In the last phase, the author suggests classifying data using an upgraded CNN model and provides four layers: Input, Convolution,Pooling, and Classifier. The CNN receives an RGB image matrix as its input rather than a grayscale value of the pixel since doing otherwise would waste data. GTSDB and GTSRB are the datasets utilised in this study for the training and testing of 43 types of traffic signs. The experimental is suggested for indication signs, prohibition signs, and warning

signs. The results obtained were 94.33%, 97.44%, and 96.47%, respectively. The proposed algorithm's overall accuracy rate is 99.21%, with an average picture processing time of fewer than 10 milliseconds.

Based on the description provided, it sounds like an interesting and detailed approach to traffic sign detection using a combination of computer vision techniques. Here are a few thoughts on the outlined methodology:

HSV Color Space Conversion: Using HSV color space for image processing, especially for tasks like traffic sign detection, can indeed be advantageous due to its robustness under varying lighting conditions. The approach to adjust threshold values for specific color components (Hue and Saturation) to isolate red, yellow, and blue colors is a common preprocessing step in image analysis.

HOG (Histogram of Oriented Gradients) and SVM (Support Vector Machines): The use of HOG as a feature descriptor followed by SVM for classification is a classic method in object detection. HOG effectively captures the shape and gradient information of objects in localized regions (ROIs), and SVM serves as a robust classifier for these feature vectors.

CNN (Convolutional Neural Network) for Classification: Transitioning to a CNN for the final classification step is logical, given CNN's success in image classification tasks. The architecture described (Input, Convolution, Pooling, and Classifier) is standard for CNNs and allows the model to automatically learn hierarchical features from the data.

Integration of Different Techniques: This paper seems to integrate multiple computer vision techniques (color space conversion, HOG feature extraction, SVM classification, and CNN classification) effectively to achieve the task of traffic sign detection. Each stage contributes to refining the features and improving the accuracy of the final classification.

B. A. Vennelakanti, "Traffic sign detection and recognition using a CNN Ensemble". [2]

In this study, the author suggested two-stage systems—detection and recognition—which were analysed and tested using publicly accessible Belgian Dataset and German dataset. Because Hue Saturation Value (HSV) processes images similar to how humans do and because it has a wide spectrum of colours, it is used in the detection phase. The following phase is locating the prospective traffic sign based on colour and determining its shape. Red signs are carefully reviewed in colour-based identification, and if they meet a particular threshold, that area is checked to see if they are traffic signs. The author concentrated on two shapes, namely the circle and triangle, for shape-based detection. It makes use of the Douglas-Peucker algorithms. This algorithm counts the edges and locates contour regions; if they meet the minimal requirement, bounding boxes are generated to demarcate the intended region. By using image thresholding and inversion filter, this sign is verified and moved on to the next phase. In the recognition phase, the author used the CNN feed-forward neural network which consists of six convolutional layers and has dropout layers in between the fully connected hidden layers to prevent overfitting. They employed a trio of CNNs in an ensemble to increase accuracy. Each of these CNNs is trained using weights that are initialised at random. The outputs of each of these three CNNs are eventually averaged to create a single ensemble model. To train this model effectively and save training time, it would be beneficial to train all these models simultaneously. Overall train sets for triangles and circles were 10582 and 16106 images. Similarly, the test sets were 3456 and 5277 images respectively When put to the test, they scored 98.11% accuracy for triangles and 99.18% accuracy for circles.

Based on the description provided, this paper seems to present a comprehensive approach to traffic sign detection and recognition using a combination of traditional image processing techniques and deep learning methods. Here are some thoughts on the methodology described:

Two-Stage System: The division of the system into detection and recognition stages is a common and effective strategy in object recognition tasks. Detection focuses on locating potential objects (traffic signs) in the image based on color and shape, while recognition involves classifying these detected objects into specific categories (e.g., triangle or circle).

Color-Based Detection: Leveraging HSV color space for traffic sign detection is a reasonable choice, as it can handle a wide range of colors and is more robust to

varying lighting conditions. The use of specific thresholds for red signs and subsequent shape-based filtering (circle and triangle) helps narrow down potential candidates efficiently.

Shape-Based Detection with Douglas-Peucker Algorithm: The adoption of the Douglas-Peucker algorithm for shape simplification and contour detection is a standard practice in computer vision. This technique aids in reducing noise and extracting essential features for bounding box generation.

CNN for Recognition: Employing a CNN-based approach for traffic sign recognition is a state-of-the-art method. The use of dropout layers in between fully connected layers is effective for preventing overfitting, and using an ensemble of CNNs (with random weight initialization) can enhance overall accuracy by leveraging different learned representations.

Training and Testing: The scale of the dataset (over 10,000 images for training) is substantial and necessary for training deep learning models effectively. The reported accuracy scores (98.11% for triangles and 99.18% for circles) are impressive and suggest the effectiveness of the proposed method.

C. Sudha S. K, "A Real-Time System for Detection and Recognition of Traffic Signs". [3]

The author presented the technique in this research in two stages, Detection and Recognition. It contains layers for image capture, pre-processing, detection, and classification, the author further clarifies. In the detection phase, 720p images are recorded from the moving car's roof at a predetermined frames-per-second rate. The visibility of the traffic sign is impacted by variations in lighting and weather. To reduce noise, the Weiner filter and Gaussian noise are utilised. Then, RGB is converted to YCbCr, where Y stands for chrominance information, the difference between the blue component and a reference value is represented by Cb, the difference between the red component and a reference value is represented by Cr. Red and blue colour objects are retrieved using the threshold approach, and all stray pixels are separated using an adjusted morphological process. In this case, the author concentrated mostly on four shapes: triangle, rectangle, circle, and octagon. The candidate object is chosen using a shape-based analysis. If there is a discrepancy in similarity between the colour-segmented sign and the template database greater than the threshold value, the sign is recognised as a traffic sign A pictogram threshold is used in the sign recognition stage. All pixels are divided into white and black pixels, depending on whether they are above or below the

threshold value. Traffic signs are therefore retrieved from the pictograms inside and used for classification using neural networks. For classification, the model employs a multilayer feed-forward CNN, and the network applies a backpropagation learning technique to train multiple layers. The primary function of the classification module is to assign the appropriate category of traffic signs to the extracted regions of interest that are supplied as input.

This paper outlines a systematic approach to real-time traffic sign detection and recognition, integrating various image processing techniques and neural network-based classification. Here are some thoughts on the described methodology:

Image Capture and Pre-processing: Recording 720p images from a moving car roof poses challenges due to variations in lighting and weather conditions. The use of a Weiner filter and Gaussian noise reduction is sensible for improving image quality and reducing noise, enhancing the visibility of traffic signs.

Color Space Conversion and Thresholding: Converting RGB images to YCbCr is a common practice in image processing. Extracting red and blue color objects using thresholding methods helps isolate potential traffic signs from the background. Morphological operations are then applied to clean up the segmented regions and remove stray pixels.

Shape-Based Analysis: Focusing on specific shapes (triangle, rectangle, circle, octagon) for candidate object selection is a practical approach for narrowing down potential traffic signs. Shape-based analysis can efficiently filter out irrelevant objects from the scene.

Sign Recognition and Classification: Using a template matching approach combined with pictogram thresholding for sign recognition is interesting. Thresholding based on pixel intensity (black and white) helps extract pictograms from the segmented regions. Employing a multilayer feed-forward CNN for classification is a robust choice, allowing the model to learn complex patterns and features from the extracted regions of interest. Backpropagation is used for training the neural network, enabling the model to optimize its parameters based on training data.

Overall Integration and Evaluation: The described system appears to be well-structured, with clear stages for detection and recognition. The combination of image processing techniques (color space conversion, thresholding, shape analysis) with deep learning (CNN-based classification) is effective for traffic sign detection in real-time scenarios. It would be beneficial for the paper to provide more details

on the dataset used for training and testing, as well as the performance metrics (e.g., accuracy, precision, recall) to evaluate the effectiveness of the proposed system.

D. Hasan, "Traffic Sign Recognition System (TSRS): SVM and Convolutional neural Network". [4]

The author of this study explains in detail how to localise traffic signs using SVM and classify them using CNN. The real-time dataset was initially built by cropping the region around the traffic signs in the video frames. To propose CNN and SVM models, they split the entire dataset into training and validation, each with 1200 images. They have considered 12 different traffic signs and each class has 100 images. Grayscale images are created by first applying a colour transformation to RGB images. The Region of Interest (ROI) is more competently refined in the second step based on the management of shape. The next stage is feature extraction using the HOG descriptor to train the SVM classifier. Here they rescaled the image to 32x32 pixels and employed a 9-bin histogram for the HOG feature vector. The SVM model will validate that it is a traffic sign. If a traffic sign is present, it will be processed further by the CNN model, which will classify it and label the sign using a bounding box. Convolutional, pooling, and fully connected layers are listed by the author as the CNN model's layers. The Rectified linear activation function was employed in this study to train the CNN model since it outperforms the competition in a variety of scenarios. They achieved 96.40% validation accuracy in the CNN model and 98.33% accuracy for SVM classification during the training procedure with 80:20 data split ratio. They created a system with real-time video and tested it to assess the outcomes.

This paper presents a systematic approach to localizing and classifying traffic signs using a combination of support vector machine (SVM) and convolutional neural network (CNN) models. Here are some insights and thoughts on the methodology described:

Dataset Preparation and Training Split: Building a real-time dataset by cropping regions around traffic signs in video frames is a practical approach for collecting relevant training data. The dataset split into training and validation sets with 1200 images each (12 classes with 100 images per class) provides a balanced training setup.

Image Pre-processing: Converting RGB images to grayscale is a common pre-processing step, particularly for scenarios where color information might not be essential for the task at hand (e.g., traffic sign detection). Refining the Region of

Interest (ROI) based on shape management helps isolate potential traffic signs, streamlining subsequent feature extraction and classification.

Feature Extraction using HOG Descriptor and SVM Classification: Utilizing Histogram of Oriented Gradients (HOG) for feature extraction and training an SVM classifier is a traditional but effective approach in object detection tasks. Rescaling images to 32x32 pixels and using a 9-bin histogram for the HOG feature vector simplifies the input representation for the SVM model.

CNN Model for Traffic Sign Classification: Integrating a CNN model for further classification after SVM validation is a logical choice, as CNNs excel at learning hierarchical representations from image data. The described CNN architecture with convolutional, pooling, and fully connected layers, along with Rectified Linear Unit (ReLU) activation, is standard and proven effective for image classification tasks.

Model Performance: Achieving a 96.40% validation accuracy with the CNN model and 98.33% accuracy for SVM classification during training is impressive. These metrics indicate that the proposed approach is effective in accurately detecting and classifying traffic signs within the given dataset.

Real-Time Testing and Evaluation: Implementing the system with real-time video processing and testing its performance is crucial for assessing its practical usability. Real-world testing helps validate the effectiveness and robustness of the developed model under varying environmental conditions.

E. M. Swathi, "Automatic Traffic sign Detection and recognition in Video Sequences". [5]

The detection and recognition stages of the system are divided into two fundamental categories by the author of this work. The detection step is separated into three sections: colour reformation from RGB to HSV, colour segmentation, and shape recognition. To separate the chromatic and achromatic components and make them resistant to changes in illumination, the image frame is first retrieved from the video sequence using ROI and transformed to HSV colour space. By removing the distracting backdrops, the search area in the image sequences is reduced and the traffic sign is segregated. The following phase was using the Douglass Peucker method algorithm, a contour approximation technique frequently used for Shape identification. The author primarily focused on two shapes, triangles and circles. If the total number of lines obtained is 3 and 8 in the final curve, then it is categorized as triangle and circles respectively. Because it is based on lines and curves, it functions well for traffic signals with certain geometric aberrations. With the HOG feature extracted, a multi-layer feed-forward neural network is used to classify traffic signs during the recognition phase. The recognized traffic sign is rescaled to 24x20 pixels and by gradient computation HOG feature vector of length 1080 is obtained. The input, hidden, and output layers of a multi-layer perceptron design each include a set of nodes that are coupled to the preceding node with certain weights. The activation function in this design is nonlinear, except for the input layer. The sigmoid function described in (4) is the activation function utilised in this design.

$$f(x) = \beta * \frac{1 - e^{-\alpha x}}{1 + e^{-\alpha x}}$$

By comparing the output error with the predicted outcomes using the backpropagation technique, the weights are modified each time the data is analysed. Seven circular and thirteen triangular classes out of a total of 21 classes were trained, with 70 photos for each class considered. The detection rate of the experiment's real-time data, which was recorded at three separate times during the day (morning, afternoon, and evening), was 82.94%, 90%, and 89.41%. They attained 97.14% and 95.71% accuracy for circular and triangular, respectively, at an average frame rate of 15 to 26 frames per second.

The paper you've described outlines an interesting approach to automatic traffic sign detection and recognition in video sequences, focusing on using techniques like color transformation, shape recognition, and neural network classification. Here are some thoughts on the paper based on the details you provided:

Detection Methodology: The paper employs a sequence of steps for traffic sign detection. By transforming images into the HSV color space, the system aims to make detection more robust against changes in illumination. This approach helps isolate the traffic signs by removing distracting backgrounds.

Shape Recognition: Using the Douglas-Peucker algorithm for contour approximation is a well-established method for shape identification. By focusing on specific shapes like triangles and circles, the system can effectively classify basic traffic sign shapes, despite potential geometric variations due to perspective or image noise.

Neural Network for Recognition: The use of a multi-layer feed-forward neural network, coupled with HOG (Histogram of Oriented Gradients) features, is a common strategy for object recognition in images. The paper's approach of rescaling detected signs and using gradient computation to extract features for classification indicates a thoughtful methodology.

Training and Results: Training the system with 70 photos per class and achieving high accuracy rates, particularly around 90% and above for circular and triangular signs, showcases the effectiveness of the proposed method. Additionally, the experiment's real-time performance metrics across different times of day demonstrate promising results in terms of detection rates and accuracy.

Limitations and Future Directions: While the described method appears effective, it's important to consider potential limitations such as performance under extreme weather conditions, complex urban environments, or uncommon sign variations. Future work could explore how to handle these challenges to enhance the system's robustness and applicability.

Notes of Literature Review:

Paper	Author and Year	Title	Reflections
[1]	Tong Guofeng, et al. (IEEE Access, 2017).	"Traffic Sign Recognition Based on SVM and Convolutional Neural Network"	In challenging scenarios, the suggested method has a high detection rate. This may be used in actual-world situations in several nations.
[2]	Aashrith Vennelakanti, et al. (IEEE Access, 2019).	"Traffic sign detection and recognition using a CNN Ensemble"	With the improved CNN model, it has a high recognition accuracy. Due to its great degree of shape reliability, any shape deviations result in categorization errors.
[3]	Sudha S.K, et al. (IJERT, 2016).	"A Real-Time System for Detection and Recognition of Traffic Signs"	This algorithm is mostly influenced by colour, and variations in lighting have an impact on the model. Since template matching is used, it will be difficult to see traffic signs from varied angles.
[4]	Nazmul Hassan, et al. (ICICCT, 2020).	"Traffic Sign Recognition System (TSRS): SVM and Convolutional neural Network"	Only 12 classes were considered for the suggested approach, which has good accuracy. Instead of using a built-in system for testing, a few more classes might be added and evaluated in complicated real-time settings.
[5]	Swathi M, et al. (IEEE Access, 2017).	"Automatic Traffic sign Detection and recognition in Video Sequences"	The geometric distortions of traffic signs and variations in lighting are well detected by this model. As a result, it may be improved and tested at night for the detection of traffic signs.

Fig. 3.1, Notes of Literature Review

Chapter 4
Proposed System

4.1. Methodology:

4.1.1 Dataset Collection:

A benchmark image dataset is required for successful research. This study is based on Bangladeshi traffic signs; hence it must use a data set that includes Bangladeshi traffic signs. However, there is no standard dataset available for Bangladeshi traffic signs. As a result, a new dataset was required to be created. We roamed around the street of Dhaka and took picture of various traffic signs and we gathered More than 50.000 photos of 43 traffic signs. As a result, there are 43 different classes for detecting and recognizing traffic signs.

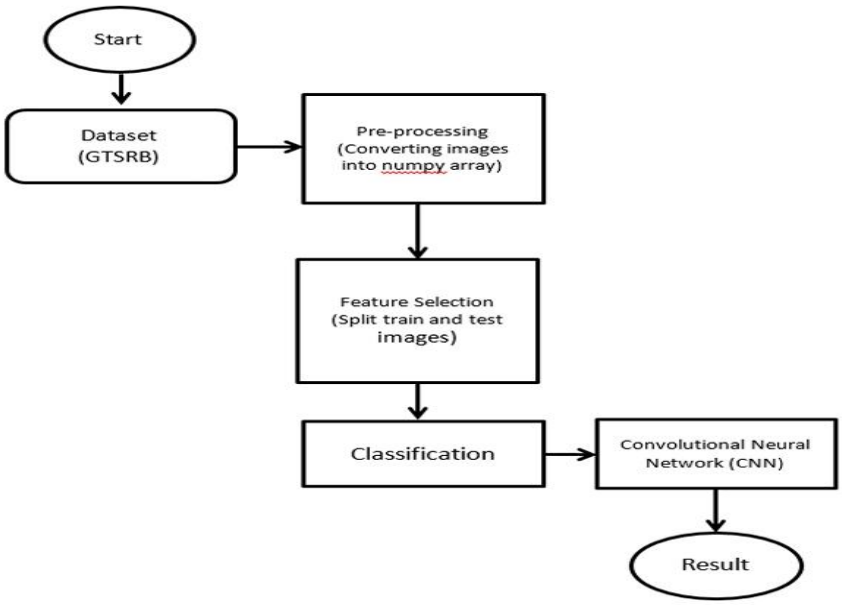

Fig. 4.1, Methodology

Fig. 4.2, Dataset Images

4.1.2 Preprocessing:

Preprocessing The Keras image data generator class generates tensor image data in batches with real time data augmentation. At first, all photos are resized to 128 × 128 pixels, which is the standard input for CNN models. Zoom range, shear range, rotation range, as well as additional factors, are applied.

4.1.3 Detection Method:

Detection Method Object detection is a computer technique for locating objects in photos and videos. Computer vision, image processing, and deep learning are all related to it. In this work, Haar cascades [11] are employed to detect signs.

Detection with OpenCV. OpenCV is an essential open-source library utilized in mod ern systems to support real-time activities. It can be used to detect objects, faces, and even human handwriting in photos and videos. Detecting objects with Haar Cascade classifiers is a useful technique. Haar Cascade is a machine learning-based method that requires training the classifier using a large number of positive and negative images.

Positive Images: These photos contain the images that our classifier should recognize.

Negative Images: Images of everything else that is not the object.

4.1.3 Detection Method:

We employed three approaches for classification:

CNN. Convolutional Neural Network (CNN): A CNN model [12] for classification is constructed with ten convolution layers, five max pooling layers, six dropout layers, one flatten layer and two dense layers.

Res Net : achieved remarkable performance in the Image Net Large Scale Visual Recognition Challenge (ILSVRC) in 2015.

It surpassed previous models by a significant margin, demonstrating the effectiveness of its architecture . Res Net serves as a robust backbone model in various computer vision tasks, including image classification, object detection, and segmentation.

Researchers and practitioners often use pre-trained ResNet models as feature extractors or fine-tune them for specific tasks.

Remember, ResNet's skip connections revolutionized deep learning by enabling the training of extremely deep networks.

VGG16 : was proposed by the Visual Geometry Group (VGG) at the University of Oxford. It's characterized by its depth, consisting of 16 layers in total.

VGG16 achieved 92.7% top-5 test accuracy on the ImageNet dataset.ImageNet contains over 14 million labeled images across 1000 classes.

to train the model. There are nearly 14 million images in the ImageNet dataset, divided into a thousand classes.

4.1.4 Experimental Results:

After preparing the model, it performed 175 steps per epoch and 35 epochs for each model. For the CNN model, validation loss is greater than training loss. To visualize the training and valid losses, see Fig. 4.3. The training accuracy data has remained stable. However, the validation accuracy results displayed some degree of variation. Despite this, the accuracy values obtained through training and validation are identical. The difference between the training loss and the validation loss is relatively small, which suggests that the CNN model did an excellent job on the dataset.

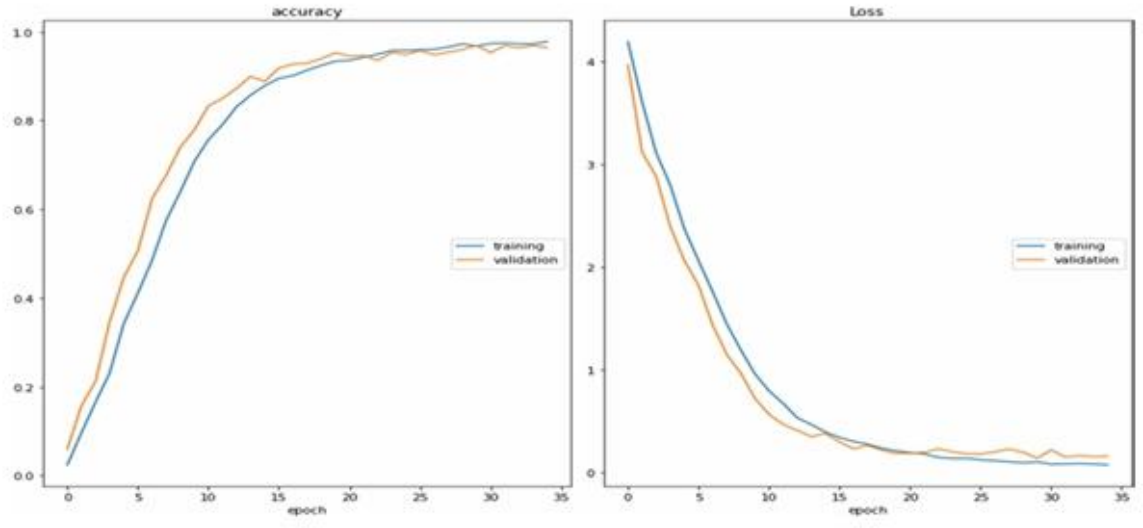

Fig. 4.3, Training and validation curve of CNN.

The number of epochs and the number of steps in each epoch remain the same across all models. In addition, this model consists of 35 epochs, each of which consists of 175 steps. For the Res Net model, the accuracy and loss curves are shown in Fig. 4.4 This model generates the most accurate results when applied to our dataset. When compared to other models, the data utilized for training and validation show no noticeable difference in accuracy. As seen in Fig. 4.4, the Res Net model achieves very good results when applied to the dataset. Most of the time, there is not much of a gap between the validation loss data and the training loss data. If the amount of training loss and validation loss data is essentially equal to that of other models, we can anticipate a better outcome. As a consequence of this, the Res Net model carried out an outstanding performance on the dataset.

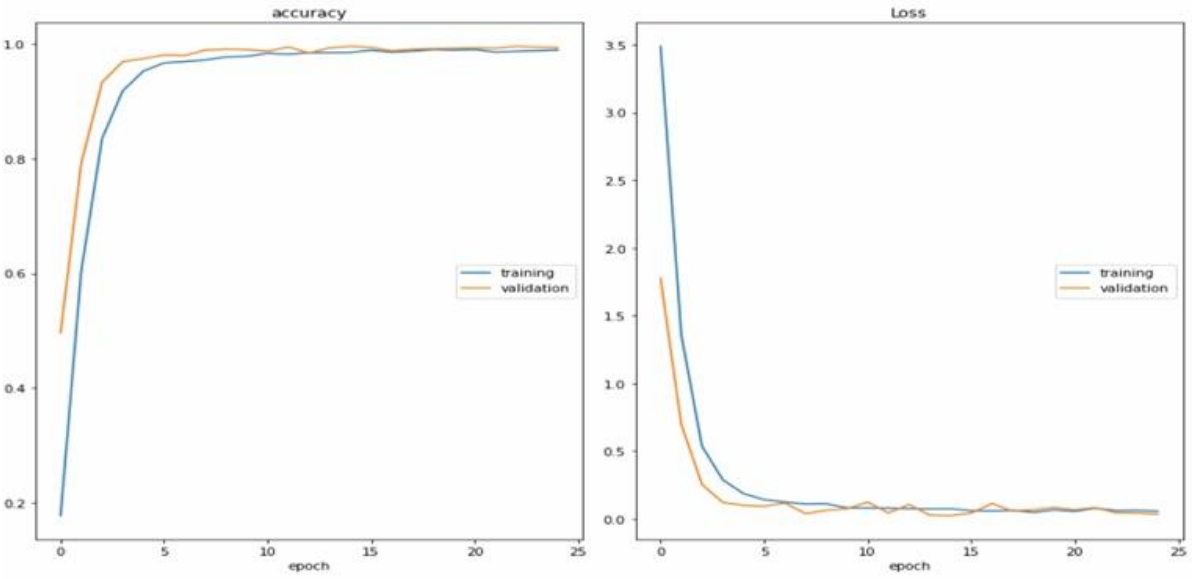

Fig. 4.4, Training and validation curve of ResNet.

this model utilizes 35 epochs, with each epoch consisting of 175 steps. In spite of the fact that the model got off to a good start at the end, the validation curve in Fig. 4.5 exhibited some instability. The amount of data lost due to training has been gradually going down. On the other hand, the data on the validation losses has a significant value that is marginally increased.

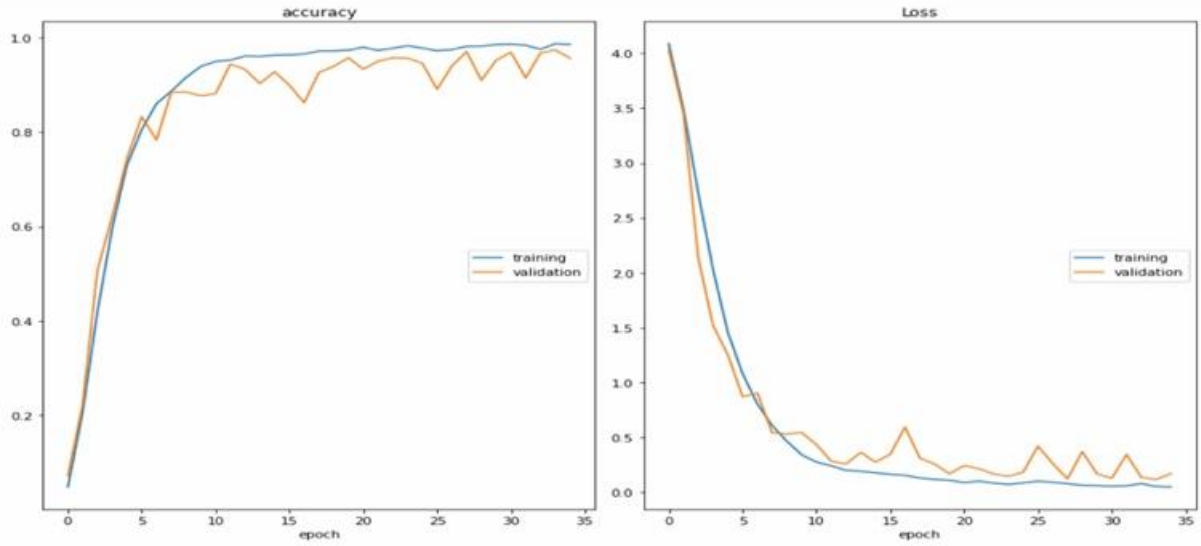

Fig. 4.5, Training and validation curve of Vgg16.

4.1.5 Model Evaluation and Analysis:

Classification Report of the Models CNN Model:

Figure 4.6 displays the precision, recall, and f1-score of the CNN model. We achieved a 97% accuracy rate.

Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score
0	1	1	1	24	0.91	1	0.95	48	1	0.9	0.95
1	1	1	1	25	1	1	1	49	1	1	1
2	1	1	1	26	0.91	1	0.95	50	1	1	1
3	1	1	1	27	1	1	1	51	0.71	1	0.83
4	0.91	1	0.95	28	0.91	1	0.95	52	1	0.7	0.82
5	1	1	1	29	0.95	1	0.98	53	1	0.75	0.86
6	1	1	1	30	0.95	1	0.98	54	1	1	1
7	1	1	1	31	1	1	1	55	1	0.9	0.95
8	1	1	1	32	0.94	0.85	0.89	56	0.83	1	0.91
9	1	1	1	33	1	1	1	57	1	1	1
10	1	0.95	0.97	34	1	0.8	0.89	58	1	1	1
11	1	1	1	35	0.87	1	0.93	59	1	0.95	0.97
12	1	0.8	0.89	36	1	0.9	0.95	60	1	1	1
13	0.91	1	0.95	37	1	1	1	61	1	1	1
14	1	1	1	38	1	0.9	0.95	62	1	1	1
15	1	1	1	39	1	1	1	63	0.87	1	0.93
16	1	1	1	40	0.91	1	0.95	64	0.95	0.95	0.95
17	1	0.95	0.97	41	1	1	1	65	1	1	1
18	0.95	0.9	0.92	42	1	0.95	0.97	66	1	0.9	0.95
19	1	1	1	43	1	0.95	0.97	67	0.87	1	0.93
20	0.95	1	0.98	44	1	0.95	0.97	68	1	1	1
21	1	1	1	45	0.87	1	0.93	69	1	0.95	0.97
22	1	1	1	46	0.95	1	0.98	**Accuracy 0.97**			
23	1	1	1	47	1	0.95	0.97				

Fig. 4.6, Classification Report of CNN.

Res Net Model: The classification report in Figure 4.7 shows the precision, recall, andf1-score for resnet model. We attained a 99% accuracy rate.

Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score
0	1	1	1	24	1	1	1	48	1	1	1
1	1	1	1	25	1	1	1	49	1	1	1
2	1	1	1	26	1	1	1	50	1	1	1
3	1	1	1	27	1	1	1	51	1	1	1
4	0.95	1	0.98	28	1	1	1	52	1	1	1
5	1	0.95	0.97	29	1	1	1	53	1	1	1
6	1	1	1	30	1	1	1	54	1	1	1

Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score
7	1	1	1	31	1	1	1	55	1	1	1
8	1	1	1	32	1	1	1	56	1	1	1
9	1	1	1	33	1	0.95	0.97	57	1	1	1
10	1	1	1	34	1	0.95	0.97	58	1	1	1
11	1	1	1	35	0.91	1	0.95	59	0.95	1	0.98
12	1	1	1	36	1	1	1	60	1	1	1
13	1	1	1	37	1	1	1	61	1	1	1
14	1	1	1	38	1	1	1	62	1	1	1
15	1	1	1	39	1	1	1	63	1	1	1
16	1	1	1	40	1	1	1	64	1	1	1
17	1	0.95	0.97	41	1	0.95	0.97	65	1	1	1
18	1	1	1	42	1	1	1	66	1	1	1
19	1	1	1	43	1	1	1	67	1	1	1
20	1	1	1	44	0.95	1	0.98	68	1	1	1
21	1	1	1	45	1	1	1	69	1	1	1
22	1	1	1	46	1	1	1	**Accuracy 0.99**			
23	1	1	1	47	1	1	1				

Fig. 4.7, Classification Report of Res Net.

Vgg16 Model: The precision, recall ,andf1-score for the vgg16 model can be found in figure 4.8 and the accuracy rate is 97%

Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score
0	1	1	1	24	0.95	1	0.98	48	1	1	1
1	1	1	1	25	1	1	1	49	0.95	1	0.98
2	1	1	1	26	1	1	1	50	1	0.95	0.97
3	1	1	1	27	1	1	1	51	1	1	1
4	0.91	1	0.95	28	0.86	0.95	0.9	52	1	1	1
5	0.95	1	0.98	29	0.87	1	0.93	53	1	0.75	0.86
6	1	1	1	30	1	1	1	54	1	1	1
7	1	1	1	31	0.95	1	0.98	55	0.91	1	0.95
8	1	1	1	32	1	1	1	56	1	1	1
9	1	1	1	33	1	0.95	0.97	57	1	1	1
10	1	1	1	34	1	0.8	0.89	58	0.91	1	0.95
11	1	1	1	35	0.91	1	0.95	59	1	0.8	0.89

Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score	Class	Precision	Recall	F1-score
12	0.83	0.95	0.88	36	1	0.95	0.97	60	1	1	1
13	0.95	1	0.98	37	1	1	1	61	1	1	1
14	1	1	1	38	1	0.9	0.95	62	1	0.95	0.97
15	1	1	1	39	1	1	1	63	1	1	1
16	1	0.9	0.95	40	0.95	1	0.98	64	0.91	1	0.95
17	0.83	1	0.91	41	0.91	1	0.95	65	1	1	1
18	0.89	0.8	0.94	42	1	0.7	0.82	66	1	0.95	0.97
19	1	1	1	43	1	1	1	67	1	1	1
20	1	1	1	44	0.95	1	1	68	1	1	1
21	1	1	1	45	1	1	1	69	1	0.95	0.97
22	1	0.95	0.97	46	1	1	1	**Accuracy 0.97**			
23	1	1	1	47	0.91	1	0.95				

Fig. 4.8, Classification Report of Res Net.

4.1.6 Prediction Results:

After the model training is completed, the prediction result can be considered an output, shown in Fig.4.9. We used the 20 images to examine our model's prediction out comes ,which are good because our models properly predict all of the given signs ,shown in Fig.4.9. As for test data we have used 1400 images.

Fig. 4.9, Prediction result of CNN model.

Fig. 4.10, Prediction result of Res Net model.

Fig. 4.11, Prediction result of Vgg16 model.

4.1.7 Comparative Analysis:

Figure4.6. indicates that the CNN model has a 97% accuracy, the Resnet model has a 99% accuracy, the Vgg16 model has a 97% accuracy

Fig. 4.12, Accuracy comparison.

4.2. Prototype:

The purpose of this study is to delve into the latest developments in traffic sign recognition using deep learning techniques. As the demand for autonomous vehicles con-tinues to grow, the reliability of traffic sign recognition algorithms is becoming increasingly important for ensuring the safety of all road users.

Traffic sign recognition technology enables vehicles to read and understand important road signs, such as speed limit signs, danger signs, and turn ahead signs. This technology not only improves the safety of drivers, but also contributes to a safer road environment for all users by providing essential information and reminders of important regulations. In order to fully comprehend the state-of-the-artwork in this field, this study conducts a comprehensive review of existing work on traffic sign recognition. These studies are divided into two categories: conventional machine learning and deep learning approaches. Additionally, this study explores the commonly used traffic sign recognition datasets and their associated challenges as well as the limitations of current studies and future research prospects. The findings of this study provide valuable insights for researchers and practitioners working in the field of autonomous vehicle technology.

4.2.1 Application:

1. Sign Up and Login:

These pages serve as entry points for users to access the application and interact with its features. Here's a brief overview of sign up and login pages.

- Sign Up page:
  - Purpose: The registration page allows new users to create an account or profile within the application.
  - User Input: It typically includes form fields to collect user information, such as email address, and password as required.
  - Validation: The registration page should perform validation checks on user inputs to ensure data integrity and accuracy.
  - Account Creation: the user's information is stored in a firebase or user management system to create a new account.
  - Feedback: The registration page should provide clear feedback to the user, indicating whether the registration.
- Login page:
  - Purpose: The login page allows registered users to authenticate themselves and gain access to their accounts.
  - User Input: It typically includes form fields to collect user information, such as email address, and password as required.
  - Authentication: the login page verifies the provided credentials against the stored user data in the firebase or user management system.
  - Access Control: If the provided credentials are valid,

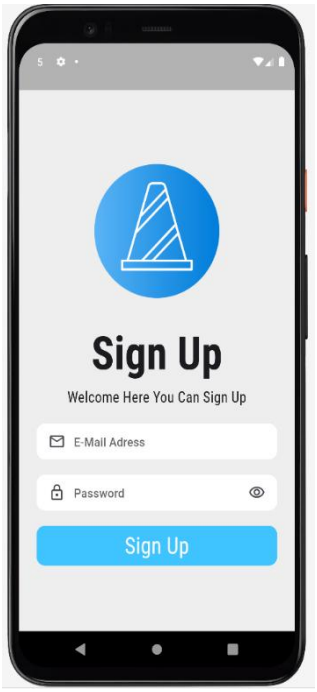

Fig. 4.13, Sign Up screen.

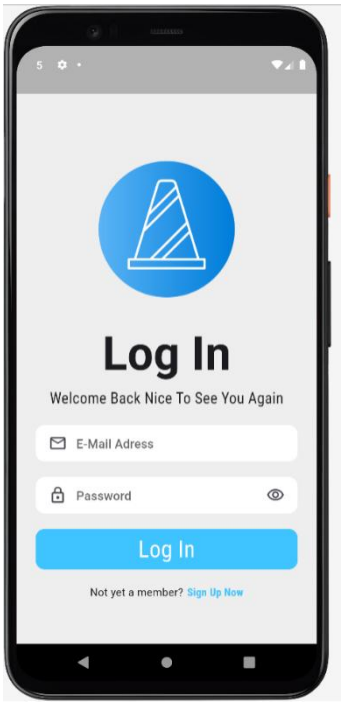

Fig. 4.14, Login screen.

the user is granted access to their account of main application.

2. AI Model:
   - Online AI Model:
   - Purpose: Online AI Model help the user to recognition the traffic sign and after the recognition the model will take the result and confirm the action.
   - User Input: Users are typically required to upload the image from the gallery or from camera to predict from the Ai model.
   - Validation: The system will implement the prediction and recognition of the image
   - Result: When the system implements the prediction, the result is the sign traffic

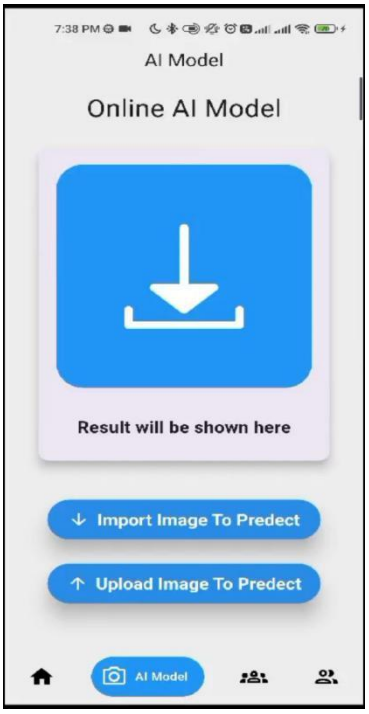

Fig. 4.15, Online AI Model screen.

- Offline AI Model:
- Purpose: Offline AI Model help the user to recognition the traffic sign and after the recognition the model will take the result and confirm the action all this option occurs without internet.
- User Input: Users are typically required to import the image from gallery to predict from the Ai model.
- Validation: The system will implement the prediction and recognition of the image.
- Result: When the system implements the prediction, the result is the sign traffic.

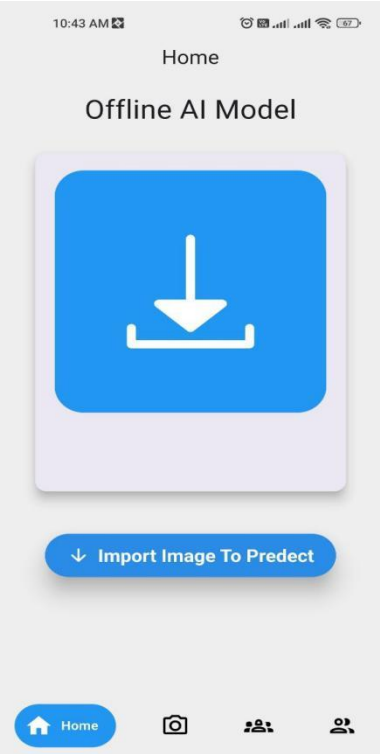

Fig. 4.16, Offline AI Model screen.

3. Result Page:
   The result page in your application provides users with information about what is this sign traffic. Based on the result, the user can take different actions. Here's a brief overview of the result page workflow:

Fig. 4.17, Online AI Model screen.

- Online AI Model: After we upload the image, it will be predicted by ai model.
  And after that the car will take the action.
  - Sum of examples for that:

1. In this example the image was uploaded, and the result is 28: children crossing we will brief that the number 28 is the number of the class that have all image like children crossing.

2. In this example the image was uploaded, and the result is 31: wild animals crossing we will brief that the number 31 is the number of the class that have all image like wild animals crossing.

Fig. 4.18, Online AI Model screen.

- Offline AI Model: After we import the image, it will be predicted by ai model this option occurs without internet.
  And after that the car will take the action.
  - Sum of examples for that:

1. In this example the image was imported, and the result is 23: Slippery Road we will brief that the number 23 is the number of the class that have all image like Slippery road.

Fig. 4.19, Offline AI Model screen.

2. In this example the image was uploaded, and the result is 8: speed limit(120km/h) we will brief that the number 8 is the number of the class that have all image like speed limit(120km/h).

Fig. 4.20, Offline AI Model screen.

4. Logout Page:

- Purpose: The logout page allow to user to leave the application after using it .

In this page all accounts are recorded in it

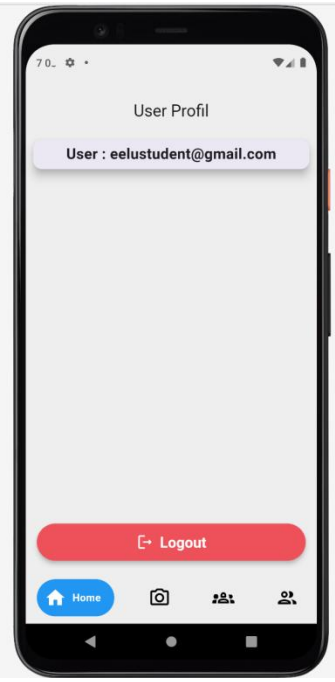

Fig. 4.21, Logout screen.

5. About Us Page:
   - Purpose: in this page contain all the team and the department of all member in the team.

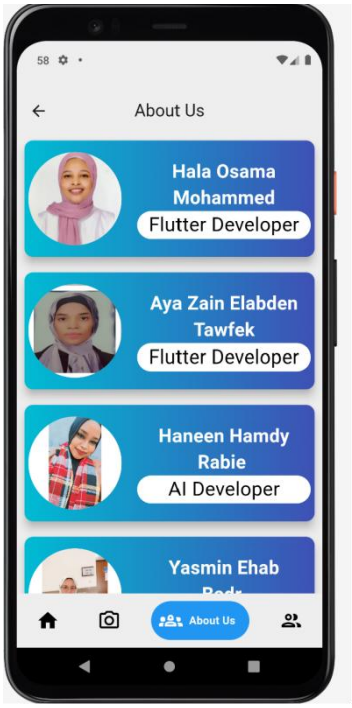

Fig. 4.22, About Us screen.

Fig. 4.23, About Us screen.

4.2.2 User Journey:

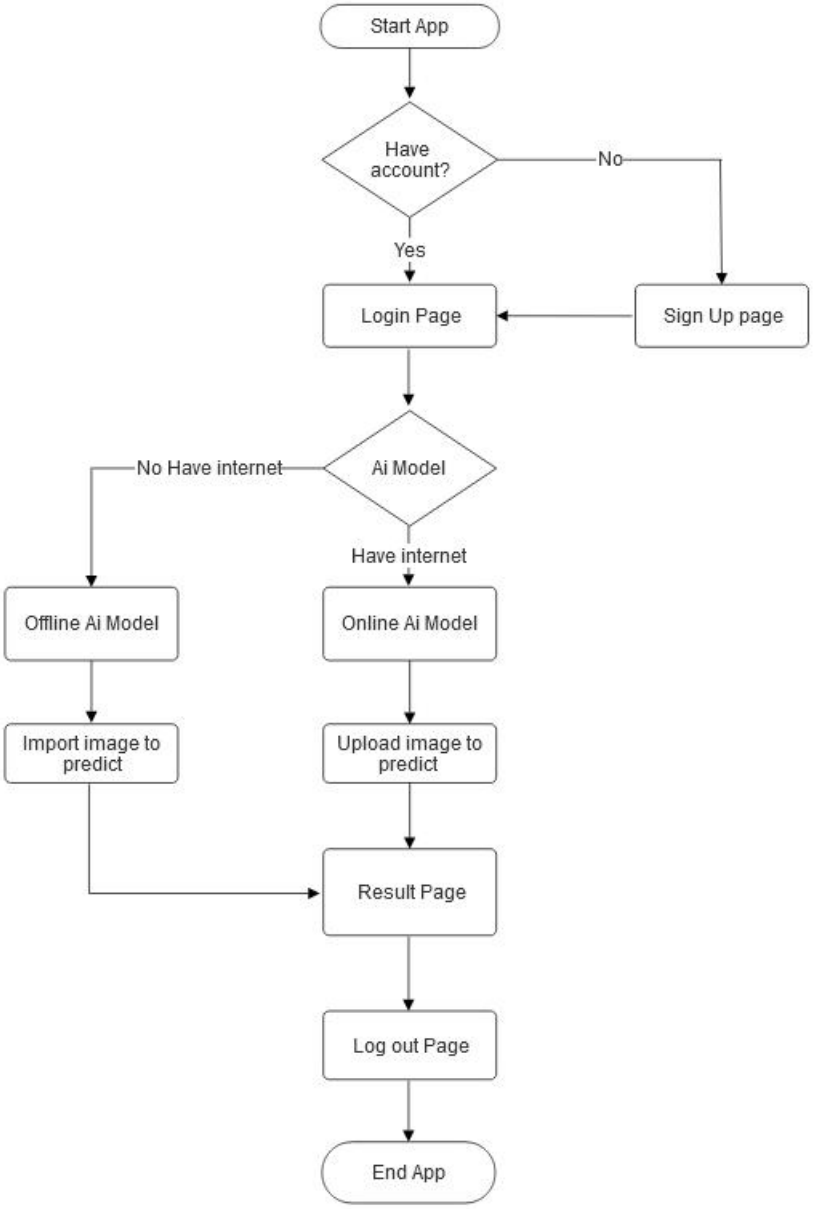

Fig. 4.24, user journey screen.

4.3. Technology:

4.3.1 Introduction:

In this project we've utilized a combination of programming languages and development tools to achieve its goals. Python, Google Colab, Visual Studio, Dart Flutter, and Android Studio played critical roles in data analysis, backend development, mobile application development, and more. These tools provided the necessary capabilities, libraries, and environments required to effectively implement various aspects of the project.

4.3.2 python:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. Python can be used to create web applications on the server side. Python runs on an interpreter system, allowing code execution as soon as it's written. This makes prototyping quick and efficient, Python was primarily used for data analysis, machine learning, and backend development. Various libraries, such as NumPy, Pandas, Matplotlib, and TensorFlow, were leveraged to handle data manipulation, visualization, and building machine learning models.

Fig.4.25, python

4.3.3 Google Colab:

Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education. Google Colab is a free cloud service that allows you to write and execute Python code directly in your browser. It provides a Jupyter Notebook environment where you can create and run code cells, combine executable code with rich text, and include images, HTML, LaTeX, and more.

Fig.4.26, Google Colab

4.3.4 Visual Studio:

Visual Studio Code, also commonly referred to as VS Code,[9] is a source-code editor developed by Microsoft for Windows, Linux, macOS and web browsers.[10][11] Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded version control with Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add functionality. Visual Studio Code was first announced on April 29, 2015 by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.[13]

Fig4.27 : Visual Studio

4.3.5 Dart:

Dart is an open-source, general-purpose programming language developed by Google. It's primarily used for building web, mobile, and desktop applications. Dart is known for its fast performance, productivity features like strong typing and asynchronous programming support, and its ability to compile both to native code and JavaScript. It's often associated with the Flutter framework, which is used for building cross-platform mobile applications. Dart's syntax is similar to other C-style languages, making it relatively easy to learn for developers familiar with those languages.

Fig.4.28 : Dart

4.3.6 Flutter

Flutter is an open-source UI software development kit created by Google. It can be used to develop cross platform applications from a single codebase for the web, Fuchsia, Android, iOS, Linux, macOS, and Windows. Flutter consists of both a UI language and a rendering engine. When a Flutter application is compiled, it ships with both the UI code and the rendering engine, which is about 4 MB compressed. This is in contrast to many other UI frameworks that rely on a separate rendering engine and only ship the UI code, such as native Android apps which rely on the device-level Android SDK or HTML/JavaScript Web apps that rely on the user's HTML engine and JavaScript engine.Framework Architecture Dart language.

Fig. 4.29: Flutter

4.3.7 Android Studio:

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from IntelliJ IDEA , Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices

- Live Edit to update composables in emulators and physical devices in real time
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

Fig.4.30 : Android Studio

Chapter 5
Conclusion & Future work

5.1. Conclusion:

The main objective of this study is to categorize the prime approaches into sections to make the concept easily understandable and to give a comprehensive review of different methods available for traffic sign detection and recognition; including along with database and research trends associated with TSDR systems [6]. In this paper, the detection of traffic signs using colour and shape based on different techniques was explored. This study suggests that support vector machine and convolutional neural network based methods were found most efficient and used because they have a high rate of detection, are very flexible, and are simple to adopt. [2] With the aid of new datasets from other nations, the total performance might be enhanced and tailored. After we used CNN model achieving an accuracy score of 97%.

5.2. Future work:

We will make a hardware system:

Esp32 Camera

Esp32 CAM.USB

Arduino

Breadboard

Jumper Wires

We will train the model for this hardware system:

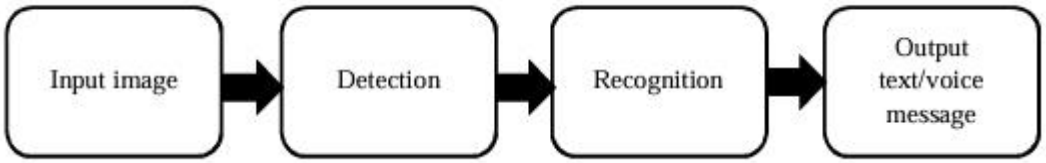

Fig.5.1 : Hardware system screen

References

[1] T. Guofeng, C. Huairong, L. Yong and Z. Kai, "Traffic sign recognition based on SVM and convolutional neural network," 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2017, pp. 2066-2071.

[2] A. Vennelakanti, S. Shreya, R. Rajendran, D. Sarkar, D. Muddegowda and P. Hanagal, "Traffic Sign Detection and Recognition using a CNN Ensemble," 2019 IEEE International Conference on Consumer Electronics (ICCE), 2019, pp. 1-4.

[3] Sudha S. K., Mahathi L, 2016, A Real-Time System for Detection and Recognition of Traffic Signs, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NCETET – 2016 (Volume 4 – Issue 17)

[4] Hasan, N.; Anzum, T.; Jahan, N. Traffic Sign Recognition System (TSRS): SVM and Convolutional Neural Network. In Inventive Communication and Computational Technologies (ICICCT); Springer: Singapore, 2021; pp. 69–79

[5] M. Swathi and K. V. Suresh, "Automatic traffic sign detection and recognition in video sequences," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 476-480, doir: 10.1109/RTEICT.2017.8256642.

[6] Wali, Safat B., Majid A. Abdullah, Mahammad A. Hannan, Aini Hussain, Salina A. Samad, Pin J. Ker, and Muhamad Bin Mansor. 2019. "Vision-Based Traffic Sign Detection and Recognition Systems: Current Trends and Challenges" Sensors 19, no. 9: 2093.

[7] Glory Reuben Maxwell and Dr.Dinesh D.Patil, “A review on traffic sign detection and recognition system”, International Research Journal of Engineering and Technology (IRJET) – 2020 (Volume 7 – Issue 05)

[8] https://morth.nic.in/sites/default/files/Road_Accidednts.pdf

[9] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno and F. Lopez-Ferreras, "Road-Sign Detection and Recognition Based on Support Vector Machines," in IEEE Transactions on Intelligent Transportation Systems, vol. 8, no. 2, pp. 264-278, June 2007.

[10] Y. Sun, P. Ge and D. Liu, "Traffic Sign Detection and Recognition Based on Convolutional Neural Network," 2019 Chinese Automation Congress (CAC), 2019, pp. 2851-2854.

[11] Viola, P., Jones, M.: Rapid Object Detection Using a Boosted Cascade of Simple Features (2001).

[12] Shin, H.-C., Roth, H.R., Gao, M., et al.: Deep convolutional neural networks for computer aided detection: CNN architectures, dataset characteristics and transfer learning CNN model analysisandvaluableinsightscanbeextendedtothedesignofhighperformanceCADsyst ems for other medical imaging tasks. IEEE Trans. Med. Imaging 35, 1285 (2016). https://doi.org/ 10.1109/TMI.2016.2528162 .

[13] Lin,C.,Li,L., Luo,W.,etal.: Transfer learning based traffic sign recognition using inception v3 model. Period. Polytech. Transp. Eng. 47, 242–250 (2019). https://doi.org/10.3311/PPTR. 11480 .

[14] Deng, J., Dong, W., Socher, R., et al.: ImageNet: a large-scale hierarchical image database, pp. 248–255 (2010). https://doi.org/10.1109/CVPR.2009.5206848 .

[15] Alom, M.Z., Taha, T.M., Yakopcic, C., et al.: The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches (2018).