Project ID: 24/CSE/3/GID-10

Fire and Smoke Detection Using Machine Learning

A PROJECT REPORT
Submitted By

Shivam Pandey
(2000270100148)
Vipin Kumar Maurya
(2000270100187)
Suryakant Patel
(2000270100162)
Saral Mittal
(2000270100135)

Under the Guidance of
Dr. Shashank Sahu

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Technology in Computer Science and Engineering

to

Department of Computer Science & Engineering
AJAY KUMAR GARG ENGINEERING COLLEGE, GHAZIABAD
DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW

May 25, 2024

Declaration

We hereby declare that the work presented in this report entitled "**Fire And Smoke Detection Using Machine Learning**", was carried out by us. We have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute. I have given due credit to the original authors / sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors / sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name	: Shivam Pandey
Roll No.	: 2000270100148
Name	: Vipin Kumar Maurya
Roll No.	: 2000270100187
Name	: Suryakant Patel
Roll No.	: 2000270100162
Name	: Saral Mittal
Roll No.	: 2000270100135

Certificate

This is to certify that the report entitled **Fire and Smoke Detection using Machine Learning** submitted by **Shivam Pandey(2000270100148), Vipin Kumar Maurya(2000270100187), Suryakant Patel (2000270100162)** and **Saral Mittal (2000270100135)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering is a bonafide record of the project work carried out by the students himself under my guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Dr. Shashank Sahu
Professor In-charge
Department of Computer Science and Engineering
Ajay Kumar Garg Engineering College

Place: Ghaziabad
Date: 25-05-24

Acknowledgements

First and foremost, we express our sincere appreciation to **Ajay Kumar Garg Engineering College** for their support and encouragement throughout this project. Their vision and commitment to innovation have been instrumental in bringing this system to fruition.

We are deeply thankful to our guide, Professor **Dr. Shashank Sahu**, whose expert guidance and mentorship have been pivotal in steering this project to success. Additionally, we extend our gratitude to the Head of Department, Professor **Dr. Anu Chaudhary**, for their unwavering support and belief in the potential of our work.

We would also like to thank the individuals and organizations who provided valuable feedback and input during the testing phase. Your insights have helped us refine the system and improve its functionality.

Last but not least, we acknowledge the trust and confidence placed in us by the users of the Fire and Smoke Detection System. Your participation and engagement are essential in shaping the future of safety technologies. We are also grateful to our parents for their unwavering support and encouragement throughout this journey.

Thank you to everyone involved for your unwavering support and commitment to advancing technology for the betterment of society.

Shivam Pandey
Vipin Kumar Maurya
Suryakant Patel
Saral Mittal

Abstract

The fire and Smoke Detection System presented in this report is an innovative solution designed so accurately detect fires and smoke and minimize false alarms and detect fires in their early stages. The increasing threat of fire hazards necessitates the development of efficient and accurate fire and smoke detection systems. Traditional methods, which rely on sensors and manual monitoring, often result in delayed detection and high false alarm rates. This project explores the implementation of a machine learning approach using Convolutional Neural Networks (CNNs) to enhance the reliability and speed of fire and smoke detection.

Leveraging a comprehensive dataset of images and videos depicting various fire and smoke scenarios, the CNN model is trained to recognize and differentiate between fire, smoke, and non-hazardous conditions. The model's architecture includes several convolutional and pooling layers, followed by fully connected layers for accurate classification.

Extensive experimentation with hyperparameters and optimization techniques ensures the model achieves high precision and recall rates, minimizing false positives and false negatives. The deployment of the trained model on real-time video feeds demonstrates its capability to provide timely alerts, thereby significantly enhancing safety measures.

This project highlights the potential of machine learning in critical safety applications and underscores the importance of continuous model improvement and dataset expansion to adapt to evolving fire detection needs. The results indicate a promising advancement in automated fire and smoke detection, with implications for various industries, including residential safety, industrial monitoring, and environmental protection.

Contents

List of Figures

Chapter 1

Introduction

Machine learning (ML) is a dynamic and rapidly evolving field within artificial intelligence (AI) and computer science. It focuses on developing algorithms and statistical models that allow computer systems to progressively improve their performance on specific tasks through learning from data, rather than through explicit programming.

At its core, machine learning enables computers to identify patterns and make decisions based on data, with the goal of improving outcomes over time. This process is akin to how humans learn, as machines are trained on large datasets to recognize patterns and make predictions or decisions based on that data.

ML algorithms can be broadly categorized into supervised, unsupervised, semi-supervised, and reinforcement learning, each serving different purposes. Supervised learning involves training a model on labeled data, where the algorithm learns the relationship between inputs and corresponding outputs. Unsupervised learning, on the other hand, deals with unlabeled data and aims to uncover hidden patterns or structures. Semi-supervised learning combines elements of both supervised and unsupervised learning, while reinforcement learning teaches an agent to make decisions in an environment to maximize cumulative reward.

Advances in machine learning have led to the development of sophisticated models such as neural networks, decision trees, support vector machines, and ensemble methods. These models have been successfully applied to a wide range of tasks, including image and speech recognition, natural language processing, recommendation systems, autonomous vehicles, and healthcare diagnostics.

Moreover, machine learning techniques are essential for handling and making sense of large-scale datasets, often referred to as big data. ML algorithms are capable of processing and analyzing vast amounts of data far more quickly and accurately than humans can, enabling businesses and organizations to extract valuable insights and make data-driven decisions.

As the field continues to grow, machine learning is increasingly being integrated into various industries and domains, revolutionizing fields as diverse as finance, marketing, manufacturing, and scientific research. The ongoing research and development in ML are aimed at enhancing model accuracy, interpretability, and scalability, ensuring its continued applicability and impact in the future.

1.1 Types of Machine Learning

1.1.1 Supervised Learning

In supervised learning, the model is trained on a labeled dataset, which means that each training example is paired with an output label. Examples: Classification (e.g., spam detection, image recognition) and regression (e.g., predicting house prices).

1.1.2 Unsupervised Learning

Description: The model learns from unlabeled data and tries to find hidden patterns or intrinsic structures in the input data. Examples: Clustering (e.g., customer segmentation), association (e.g., market basket analysis).

1.1.3 Semi-supervised Learning

Description: This approach uses a small amount of labeled data and a large amount of unlabeled data. It sits between supervised and unsupervised learning. Examples: Improving accuracy in image classification when labels are scarce.

1.1.4 Reinforcement Learning

Description: The model learns by interacting with an environment, receiving rewards or penalties based on its actions, and aims to maximize cumulative rewards. Examples: Game playing (e.gAlphaGo), robotic control.

1.2 Key Concepts in Machine Learning

1.2.1 Algorithms

Neural Networks:

Inspired by the human brain, these are networks of nodes (neurons) that can capture complex patterns in data.

Support Vector Machines (SVMs):

Used for classification and regression tasks, focusing on finding a hyperplane that best separates different classes.

k-Nearest Neighbors (k-NN):

A simple, instance-based learning algorithm that classifies data based on the closest training examples in the feature space.

1.2.2 Model Training

Training Set:

The portion of the dataset used to train the model

Validation Set:

Used to tune model parameters and prevent overfitting.

Test Set:

Used to evaluate the final model's performance.

1.2.3 Overfitting and Underfitting

Overfitting:

When a model learns the training data too well, including noise and outliers, leading to poor performance on new, unseen data.

Underfitting:

When a model is too simple to capture the underlying pattern in the data, resulting in poor performance on both training and new data.

1.2.4 Feature Engineering

The process of selecting, modifying, or creating new features (variables) to improve model performance.

1.2.5 Evaluation Metrics

Accuracy:

The ratio of correctly predicted instances to the total instances.

Precision and Recall:

Metrics used for evaluating classification models, especially when dealing with imbalanced datasets.

F1 Score:

The harmonic mean of precision and recall, providing a balance between the two.

Mean Squared Error (MSE):

A common metric for regression tasks, measuring the average squared difference between predicted and actual values.

1.3 Advantages of Machine Learning

There are several advantages of using machine learning, including:

1.3.1 Improved accuracy

Machine learning algorithms can analyze large amounts of data and identify patterns that may not be apparent to humans. This can lead to more accurate predictions and decisions.

1.3.2 Ability to learn from experience

Machine learning models can improve over time as they are exposed to more data, which enables them to learn from their mistakes and improve their performance.

1.3.3 Better predictions

Machine learning models can make predictions with greater accuracy than traditional statistical models.

1.4 Disadvantages of Machine Learning

While there are many advantages to using machine learning, there are also some potential disadvantages to consider, including:

1.4.1 Complexity

Machine learning algorithms can be complex and difficult to understand, which can make it difficult for non-experts to use or interpret the results.

1.4.2 Data requirements

Machine learning algorithms require large amounts of data to train and be accurate, which can be difficult to collect and preprocess.

1.4.3 Biased data

Machine learning models are only as good as the data they are trained on, and if the data is biased, the model will also be biased.

1.5 Applications of Machine Learning

1.5.1 Natural Language Processing (NLP)

Machine learning techniques applied to understand and generate human language. Applications include language translation, sentiment analysis, and chatbots.

1.5.2 Computer Vision

Enabling computers to interpret and make decisions based on visual data. Applications include facial recognition, object detection, and autonomous driving.

1.5.3 Healthcare

Predictive analytics for disease outbreaks, personalized treatment plans, and medical imaging diagnostics.

1.5.4 Finance

Fraud detection, algorithmic trading, and credit scoring.

1.5.5 Retail

Customer segmentation, recommendation systems, and inventory management.

1.5.6 Manufacturin

Predictive maintenance, quality control, and supply chain optimization.

1.5.7 Transportation

Autonomous driving, route optimization, and demand forecasting.

1.6 Challenges in Machine Learning

1.6.1 Data Quality and Quantity

High-quality and large amounts of data are crucial for training effective models.

1.6.2 Overfitting and Underfitting

Overfitting occurs when a model learns the training data too well, including noise, and performs poorly on new data. Underfitting occurs when a model is too simple to capture the underlying pattern of the data.

1.6.3 Bias and Fairness

Ensuring that machine learning models do not perpetuate or amplify biases present in the training data.

1.6.4 Interpretability

Understanding how and why a model makes certain predictions, especially in critical applications like healthcare and finance.

Machine learning continues to evolve rapidly, driven by advancements in computational power, availability of large datasets, and innovations in algorithms and model architectures. This technology is becoming increasingly integral to various industries, providing insights and automations that were previously unattainable .

Chapter 2

Basics

A machine learning project for fire and smoke detection using Convolutional Neural Networks (CNNs) typically involves several key sections. Below is a structured outline of these sections:

2.1 Problem Definition and Objectives

- Problem Statement: Clearly define the goal of the project, e.g., "Develop a CNN-based model to detect fire and smoke in images and videos."
- Objectives: List the specific aims, such as achieving a high detection accuracy, minimizing false positives, and ensuring real-time detection capabilities.

Data Collection

- Data Sources: Identify and gather data from various sources, such as:
  - Public datasets (e.g., CIFAR-10 for preliminary testing, specialized fire datasets)
  - Custom datasets from video surveillance footage or drone imagery
  - Synthetic data if real-world data is scarce
- Data Annotation: Label the collected data for training, validation, and testing. Labels typically include "fire," "smoke," and "no fire/smoke."

Data Preprocessing

- Data Cleaning: Remove any corrupted or irrelevant images.
- Data Augmentation: Apply transformations like rotation, scaling, and flipping to increase the diversity of the training data.
- Normalization: Scale pixel values to a standard range (e.g., 0-1 or -1 to 1).
- Splitting: Divide the data into training, validation, and test sets (e.g., 70

2.2 Model Design and Architecture

- CNN Architecture: Design a CNN suitable for fire and smoke detection. Typical components include:
- Convolutional Layers: For feature extraction.
- Pooling Layers: For downsampling and reducing dimensionality.
- Fully Connected Layers: For classification based on extracted features.
- Activation Functions: Commonly ReLU for non-linearity.
- Output Layer: Using softmax or sigmoid activation for multi-class or binary classification.
- Transfer Learning: Consider using pre-trained models like VGG16, ResNet, or MobileNet for faster and more accurate training.

Model Training

- Hyperparameter Tuning: Experiment with different learning rates, batch sizes, and optimizer types (e.g., Adam, SGD).

- Loss Function: Choose an appropriate loss function, such as categorical cross-entropy for multi-class classification.
- Training Process: Train the model on the training data, monitoring performance on the validation set to avoid overfitting.

Model Evaluation

- Metrics: Evaluate the model using metrics such as accuracy, precision, recall, F1-score, and confusion matrix.
- Validation: Ensure the model performs well on unseen validation data.
- Testing: Finally, test the model on the test set to assess its generalization capability.

Model Optimization

- Fine-Tuning: Adjust hyperparameters and retrain the model for better performance.
- Regularization: Apply techniques like dropout and L2 regularization to prevent overfitting.
- Model Compression: Optimize the model for deployment, possibly using techniques like pruning and quantization.

2.3 Deployment

- Integration: Embed the model into an application or system (e.g., CCTV surveillance, drone monitoring systems).
- Real-Time Processing: Implement real-time processing capabilities if necessary, using frameworks like TensorFlow Lite or OpenCV.
- Monitoring: Set up monitoring to track the model's performance post-deployment and update it as needed.

Documentation and Reporting

- Documentation: Provide comprehensive documentation of the project, including data sources, preprocessing steps, model architecture, and training process.

- Reporting: Prepare a detailed report summarizing the findings, performance metrics, challenges encountered, and future work.

Future Work and Improvements

- Continuous Learning: Implement mechanisms for the model to learn from new data over time.

- Enhanced Features: Explore additional features like temporal analysis in video frames or multi-sensor data integration.

- User Feedback: Incorporate feedback from users to further refine and improve the model.

Chapter 3

Feature Submodels

In a fire and smoke detection project using Convolutional Neural Networks (CNNs), the logic revolves around leveraging image data to automatically identify the presence of fire or smoke. Here's a concise explanation:

3.1 Feature Selection and Engineering

- Feature selection involves choosing the most relevant attributes or characteristics from the data that are informative for distinguishing between fire, smoke, and non-fire/smoke instances.
- The logical rationale behind feature selection lies in identifying attributes that exhibit distinct patterns or behaviors associated with fire and smoke, such as changes in temperature, presence of particulate matter, or spectral characteristics in images.
- Feature engineering involves transforming and creating new features from raw data to better represent the underlying patterns. For example, you might compute temporal derivatives of sensor readings to capture rate-of-change information.
- Discuss the logical considerations behind each feature choice, including domain knowledge, experimentation, and validation techniques used to ensure the selected features are relevant and

Algorithm Selection:

- Different machine learning algorithms have varying strengths and weaknesses, making algorithm selection a critical decision.
- For fire and smoke detection, convolutional neural networks (CNNs) are often chosen for image-based detection tasks due to their ability to capture spatial patterns effectively.
- Decision trees or ensemble methods like random forests may be suitable for combining multiple sensor inputs or features from different modalities.
- Justify your algorithm choice based on factors such as the complexity of the data, scalability requirements, interpretability needs, and the trade-off between accuracy and computational efficiency.

Decision Thresholds:

- Decision thresholds determine the sensitivity of the model to detecting fire and smoke.
- A low threshold increases sensitivity but may result in more false positives (incorrectly detecting fire/smoke when there isn't any), while a high threshold reduces false positives but may lead to more false negatives (missing actual fire/smoke instances).
- The logical rationale for setting decision thresholds involves balancing the consequences of false alarms (e.g., wasted resources or panic) against the risks of missed detections (e.g., delayed response to a fire incident).

3.2 Model Interpretability

- Model interpretability refers to the ability to understand and explain the decisions made by the machine learning model.

- In fire and smoke detection systems, interpretability is crucial for building trust and facilitating human understanding of the system's behavior.
- Techniques such as feature importance analysis, SHAP (SHapley Additive exPlanations), LIME (Local Interpretable Model-agnostic Explanations), or model visualization can enhance interpretability by highlighting the most influential features or providing insights into the model's decision-making process.

Uncertainty and Confidence Estimation:

- Uncertainty estimation quantifies the model's confidence in its predictions and helps assess the reliability of those predictions.
- Fire and smoke detection systems can benefit from uncertainty estimation to handle ambiguous or noisy inputs more effectively.
- Techniques such as Bayesian neural networks, dropout uncertainty, or Monte Carlo sampling can be employed to estimate uncertainty levels and provide confidence intervals around predictions.

3.3 Feedback Mechanisms

- Feedback mechanisms enable the model to learn and adapt based on new information or user feedback.
- Logical feedback loops involve collecting data from deployed systems, evaluating model performance, and incorporating feedback into model updates.
- Techniques such as online learning, active learning, or human-in-the-loop approaches can facilitate continuous improvement and adaptation of fire and smoke detection models.

Chapter 4

Search Algorithm

When discussing search algorithms for fire and smoke detection, we typically refer to algorithms that systematically explore and analyze data (such as images or video frames) to identify the presence of fire or smoke. Here's an in-depth look at search algorithms used in the context of fire and smoke detection:

4.1 Heuristic Search Algorithms

(A-star) Algorithm*:

- **Usage:** Primarily used for pathfinding and graph traversal, but can be adapted for searching through pixel data in images.
- **Mechanism:** Uses heuristics to prioritize nodes, combining the cost to reach a node and the estimated cost to the goal.
- **Adaptation for Fire/Smoke Detection:** Can be used to detect areas of interest by searching through image pixels and prioritizing areas based on color intensity, motion vectors, or other features indicative of fire or smoke.

Best-First Search:

- **Usage:** Searches the most promising nodes first based on a specified rule or heuristic.

- **Mechanism:** Similar to A*, but without considering the cost to reach the node.
- **Adaptation for Fire/Smoke Detection:** Can be used to identify regions of interest by searching pixels that exhibit characteristics of fire or smoke first, such as high intensity or rapid changes in pixel values.

4.2 Image Processing Techniques Edge Detection Algorithms

Canny Edge Detector:

- **Usage:** Detects edges in an image, which can highlight the contours of flames or smoke.
- **Mechanism:** Uses gradients to find edges and suppresses non-maximum edges to thin out the result.

Sobel Operator:

- **Usage:** Detects edges based on gradients.
- **Mechanism:** Applies convolution with Sobel kernels to find gradient magnitudes and directions.

Color-Based Search Algorithms:

Color Thresholding:

- **Usage:** Identifies pixels within a certain color range.
- **Mechanism:** Searches for pixels with RGB values indicative of fire (e.g., shades of orange, yellow) or smoke (gray, white).

Color Space Transformations:

- **Usage:** Converts image from one color space to another (e.g., RGB to HSV) and searches for color-specific patterns.
- **Mechanism:** Transforms colors to a space where fire/smoke colors are more easily separable, then applies thresholding.

4.3 Machine Learning-Based Search Algorithms

Convolutional Neural Networks (CNNs):

- **Usage:** Automatically learn features indicative of fire or smoke from training data.
- **Mechanism:** Consists of layers that learn hierarchical feature representations, which can detect complex patterns in images.
- **Adaptation for Fire/Smoke Detection:**
  - **Training:** Train on labeled datasets containing images of fire, smoke, and non-fire/smoke scenes.
  - **Prediction:** Sliding window or region proposal techniques to search and classify regions within an image or frame.

  Object Detection Algorithms:

  YOLO (You Only Look Once):

  - **Usage:** Real-time object detection.
  - **Mechanism:** Divides the image into a grid and predicts bounding boxes and probabilities for each grid cell.
  - **Adaptation for Fire/Smoke Detection:** Trained to detect and localize fire and smoke within images.

  R-CNN (Region-based CNN):

  - **Usage:** Object detection with regions of interest.
  - **Mechanism:** Proposes regions of interest, extracts features, and classifies them.
  - **Adaptation for Fire/Smoke Detection:** Trained to identify regions of fire or smoke.

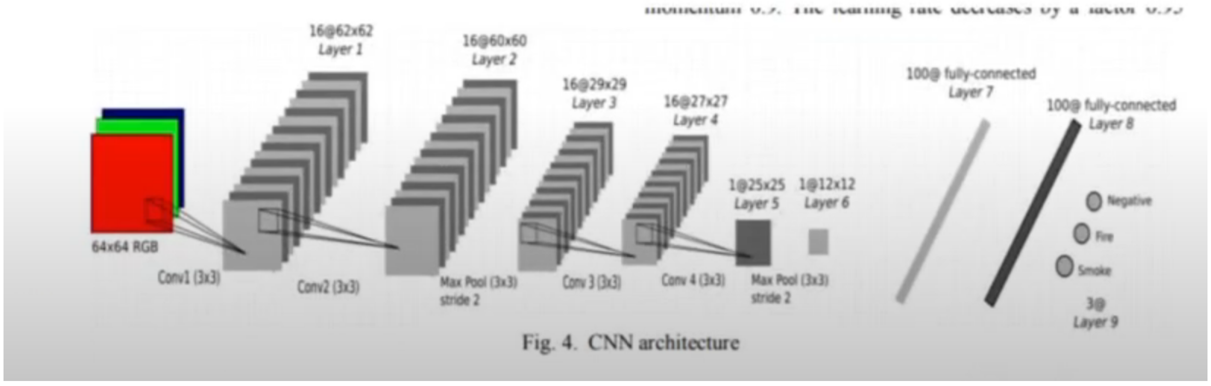


Figure 4.1: Enter Caption

4.4 Search Algorithm for Fire and Smoke Detection

Data Preparation: The initial step is to gather a diverse dataset consisting of images and videos of fire and smoke. This dataset should be balanced and include various scenarios to enhance the model's robustness. Data augmentation techniques like rotation, zoom, and shifts are used to increase the dataset's variability, which helps in better generalization of the model (Machine Learning Projects).

Model Architecture: A typical CNN model for fire and smoke detection may start with several convolutional layers for feature extraction, followed by pooling layers to reduce dimensionality. Batch normalization and activation functions like ReLU are used to improve training efficiency and model performance. The final layers are usually fully connected, leading to a softmax classifier for the detection output (Machine Learning Projects).

Training Process: The training process involves splitting the dataset into training and testing sets, using techniques like one-hot encoding for labels. Class weights can be adjusted to handle any imbalance in the dataset. The model is trained using an optimizer such as Stochastic Gradient Descent (SGD) with a specified learning rate and decay. Loss functions like binary cross-entropy are commonly used for training binary classification models (Machine Learning Projects).

Advanced Techniques: To enhance detection accuracy, advanced techniques like multi-scale feature extraction, channel attention mechanisms, and deep supervision can be employed. These techniques help in capturing more nuanced features and improving the model's sensitivity to fire and smoke (SpringerOpen).

Evaluation and Optimization: After training, the model is evaluated using metrics like accuracy, precision, recall, and F1-score. Visualization tools are used to plot training and validation losses and accuracies to monitor the model's performance over epochs. Techniques such as cross-validation can further ensure that the model generalizes well to unseen data (Machine Learning Projects).

Real-time Implementation: For real-time detection, the trained model can be integrated into a system that processes live video feeds. This involves resizing and normalizing the video frames before passing them to the model for prediction. The system can then trigger alerts when fire or smoke is detected, aiding in early intervention (MDPI).
Addressing Challenges: One of the major challenges in fire and smoke detection using CNNs is the high rate of false positives caused by environmental factors like fog or varying lighting conditions. Techniques like thermal imaging and the use of additional sensors can help mitigate these issues, providing more reliable detection in diverse environments (MDPI).

By combining these techniques and continually refining the model with new data and advanced algorithms, a robust and effective fire and smoke detection system using deep learning can be developed, enhancing public safety and response times.

Chapter 5

Machine Learning and Pattern Recognition

Fire and smoke detection using deep learning and pattern recognition involves integrating advanced machine learning techniques with traditional image processing methods. Here's an overview of the approach:

5.1 Data Collection and Preprocessing

5.1.1 Data Collection

The first step in developing a robust fire and smoke detection system using Convolutional Neural Networks (CNNs) is to collect a comprehensive dataset. This dataset should include a wide range of images and videos depicting fire and smoke in various conditions to ensure the model can generalize well to different real-world scenarios.

– **Diverse Conditions:** The dataset should capture fire and smoke under different environmental conditions such as daylight, night, fog, and different weather conditions. This diversity is crucial for training a model that can perform reliably in varied real-world situations.

– **Different Scenarios:** It should include images from different settings such as forests, urban areas, industrial sites, and residential buildings. Each of these environments has unique characteristics that the model needs to learn.

– **Data Sources:** Sources for this data can include public datasets like CIFAR-10 for preliminary testing, specialized fire datasets such as FireNet and FiSmo, custom datasets from video surveillance footage or drone imagery, and synthetic data generated using techniques like Generative Adversarial Networks (GANs) if real-world data is scarce.

5.1.2 Preprocessing

Once the data is collected, it needs to be preprocessed to ensure it is suitable for training the CNN model. Preprocessing steps include:

Resizing and Normalization

– **Resizing:** All images should be resized to a uniform size, such as 224x224 pixels. This ensures consistency in the input data, allowing the CNN to process the images efficiently.

– **Normalization:** Pixel values should be scaled to a standard range (e.g., 0-1 or -1 to 1). Normalization helps in speeding up the convergence of the training process and ensures that the model does not become biased towards any particular range of pixel values.

Data Augmentation

– **Rotation:** Randomly rotating images can help the model learn to recognize fire and smoke from different angles.

– **Zoom:** Applying random zoom can help in making the model invariant to the scale of the objects within the images.

- **Flipping:** Horizontal and vertical flipping of images can increase the variability of the dataset, helping the model to generalize better.
- **Other Transformations:** Additional augmentations like brightness adjustment, contrast adjustment, and cropping can further increase the diversity of the training data, enabling the model to handle various visual appearances of fire and smoke.

Labeling

- **Annotation:** Each image in the dataset should be annotated with appropriate labels indicating the presence of fire, smoke, or neither. This is essential for supervised learning, where the model learns to associate specific patterns in the images with the corresponding labels.
- **Quality Control:** Ensuring the accuracy of these labels is crucial. Mislabeling can lead to incorrect learning and poor model performance. It might be beneficial to use a combination of manual and automated labeling techniques to improve label accuracy.

5.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) represent a pivotal advancement in deep learning, specifically tailored for processing structured grid data like images. This architectural innovation has had a profound impact on computer vision, significantly enhancing the accuracy and efficiency of tasks such as image classification, object detection, and segmentation.

At the core of CNNs are several specialized layers designed to extract and learn hierarchical representations from image data. The convolutional layers are the workhorses of CNNs, where filters are applied across the input image to capture local patterns and features. Each filter learns to detect different aspects of the image, such as edges, textures, or shapes, enabling the network to understand the spatial structure of the data.

Following convolutional layers, pooling layers reduce the spatial dimensions of the feature maps while retaining the most important information. Common pooling methods include max pooling, which selects the maximum value within a window, and average pooling, which computes the average value. These operations help in controlling overfitting and reducing the computational complexity of the model.

Moreover, CNNs utilize activation functions like ReLU (Rectified Linear Unit) to introduce non-linearity, enabling the network to learn complex relationships within the data. ReLU activation functions transform negative pixel values to zero, ensuring that the network can model more intricate patterns in the data.

A crucial aspect of CNNs is their ability to handle high-dimensional data efficiently. By leveraging fully connected layers towards the end of the network, CNNs integrate the features extracted by previous layers for final classification. These layers aggregate and process the learned representations to produce predictions for specific tasks.

Overall, CNNs have demonstrated exceptional performance across various computer vision tasks, making them indispensable in fields such as medical imaging, autonomous vehicles, and video surveillance. Their capability to automatically learn features from raw pixel data has transformed the landscape of artificial intelligence, paving the way for more sophisticated and accurate systems in the future.

5.2.1 Convolutional Layers

– **Function:** The primary function of convolutional layers is to extract local features from the input images. This is achieved by applying a set of learnable filters (also known as kernels) across the input image.

- **Operation:** Each filter convolves across the image, performing element-wise multiplication and summing the results to produce a feature map. This operation highlights specific patterns such as edges, textures, and colors in different parts of the image.
- **Multiple Filters:** Typically, multiple filters are used in each convolutional layer, allowing the network to learn a variety of features at different levels of abstraction. Early layers might detect simple features like edges, while deeper layers can capture more complex structures such as shapes and objects.
- **Activation Functions:** After convolution, an activation function, usually ReLU (Rectified Linear Unit), is applied to introduce non-linearity into the model, allowing it to learn more complex patterns.

5.2.2 Pooling Layers

- **Function:** Pooling layers are used to reduce the spatial dimensions (width and height) of the feature maps while retaining the most important features. This helps in reducing the computational complexity and preventing overfitting.
- **Types of Pooling:** The most common type of pooling is max pooling, which selects the maximum value from each patch of the feature map. Another type is average pooling, which takes the average of the values in each patch.
- **Operation:** For example, in a max pooling operation with a 2x2 filter and a stride of 2, the feature map is divided into non-overlapping 2x2 regions, and the maximum value from each region is taken to form a downsampled feature map.

- **Impact:** Pooling layers help in making the detection of features invariant to small translations, rotations, and distortions in the input image. This enhances the robustness of the CNN.

5.2.3 Fully Connected Layers

- **Function:** Fully connected layers, also known as dense layers, are used to integrate the features extracted by the convolutional and pooling layers for the purpose of classification. They connect every neuron in one layer to every neuron in the next layer.
- **Operation:** The high-level feature maps produced by the convolutional and pooling layers are flattened into a one-dimensional vector. This vector is then fed into one or more fully connected layers.
- **Learning Complex Relationships:** These layers enable the network to learn complex relationships and interactions between the features. They apply a series of weights and biases to the input vector, transforming it into an output vector.
- **Output Layer:** The final fully connected layer typically outputs a vector whose length is equal to the number of classes in the classification task. For multi-class classification, a softmax activation function is applied to convert the output into probabilities, whereas for binary classification, a sigmoid activation function is used.

5.2.4 Integration and Training

- **End-to-End Learning:** CNNs are trained end-to-end using backpropagation. During training, the network adjusts the weights of the filters and fully connected layers to minimize the loss function, which measures the difference between the predicted and actual labels.

- **Gradient Descent:** Optimization algorithms such as stochastic gradient descent (SGD) or Adam are used to update the weights iteratively based on the gradients of the loss function with respect to the weights.
- **Feature Hierarchies:** By stacking multiple convolutional and pooling layers, CNNs build a hierarchy of features, from low-level edges and textures in the initial layers to high-level object parts and semantics in the deeper layers. This hierarchical feature extraction is what makes CNNs highly effective for image-related tasks.

5.3 Traditional Pattern Recognition Methods

Traditional pattern recognition methods have been extensively used in computer vision tasks and are often combined with deep learning techniques to achieve better performance. These methods include:

5.3.1 Feature Extraction

- **Edge Detection:** Techniques like Sobel and Canny edge detectors are used to identify sharp changes in pixel intensity, which often correspond to object boundaries or edges. These methods are crucial for pre-processing images and extracting important structural information.

5.3.2 Statistical Methods

- **Principal Component Analysis (PCA):** PCA is a statistical method used for dimensionality reduction. It identifies the most important features in a dataset, allowing for the reduction of the number of variables under consideration while preserving the most relevant information. In image processing, PCA can be applied to reduce the dimensionality of feature vectors extracted from images.

5.3.3 Machine Learning Models

- **Support Vector Machines (SVM):** SVMs are supervised learning models that can be used for classification or regression tasks. They work by finding a hyperplane that best separates classes in a high-dimensional feature space. In traditional pattern recognition, SVMs are often used for initial classification tasks before refining the results using more complex models such as deep neural networks.

5.3.4 Role and Integration with Deep Learning

Traditional pattern recognition methods are typically used as preprocessing steps or as initial classifiers in a pipeline that includes deep learning models:

- **Preprocessing:** Techniques like edge detection and color analysis are used to extract meaningful features from raw images before feeding them into a deep learning model. These methods can help simplify the learning task by providing more relevant and less redundant information.

Integrated Approach Combining CNNs with pattern recognition techniques:

- **Hybrid models:** Using CNNs for feature extraction and traditional classifiers for final classification.

- **Multi-scale analysis:** Applying pattern recognition at various scales to detect fire and smoke in different sizes and locations within the image.

5.4 Integrated Approach: Combining CNNs with Pattern Recognition Techniques

Combining CNNs with traditional pattern recognition techniques involves leveraging the strengths of both approaches to achieve better results in fire and smoke detection:

5.4.1 Hybrid Models

- **Feature Extraction with CNNs:** CNNs are well-suited for learning hierarchical features directly from raw image data. They excel at capturing spatial dependencies and patterns across different scales.
- **Final Classification with Traditional Classifiers:** After extracting features using CNNs, traditional classifiers such as Support Vector Machines (SVM) or k-Nearest Neighbors (k-NN) can be employed for final classification. These classifiers can make use of the high-level features extracted by CNNs to make decisions based on learned patterns.
- **Benefits:** This approach combines the ability of CNNs to learn complex features with the interpretability and effectiveness of traditional classifiers, potentially improving overall accuracy and reducing false positives.

5.4.2 Multi-scale Analysis

- **Application of Pattern Recognition at Various Scales:** Fire and smoke can appear in images at different scales, from large fires in landscapes to small smoke plumes in confined spaces. Applying pattern recognition techniques at multiple scales allows the system to detect fire and smoke in different sizes and locations within the image.

– **Techniques:** This can involve using different window sizes for feature extraction or applying multi-resolution analysis techniques to cover a wide range of scales.

– **Benefits:** Ensures comprehensive detection capabilities across various contexts and environmental conditions.

5.4.3 Training and Optimization Data Splitting

– **Training and Validation Sets:** The dataset is typically divided into training, validation, and testing sets. The training set is used to train the model, the validation set is used to tune hyperparameters and monitor the model's performance, and the test set is used to evaluate the final model's performance.

Loss Functions

– **Binary Cross-Entropy:** This loss function is commonly used for binary classification tasks, such as detecting the presence or absence of fire and smoke in images.

Optimization Algorithms

– Stochastic Gradient Descent (SGD) and Adam:These are popular optimization algorithms used to adjust the weights of the neural network during training. Adam, in particular, is known for its adaptive learning rate and efficient handling of sparse gradients.

5.4.4 Evaluation Metrics

– **Accuracy, Precision, Recall, F1-Score:** These metrics are used to evaluate the performance of the fire and smoke detection model. Accuracy measures the overall correctness of the model, precision measures the accuracy of positive predictions, recall measures the ability of the model to find all positive instances, and F1-score is the harmonic mean of precision and recall, providing a balanced measure between the two.

5.4.5 Real-Time Implementation:

Deploying the trained model for real-time detection involves:

- **Live video feed processing:** Capturing frames from video streams, resizing, and normalizing them.
- **Prediction:** Passing frames through the model to detect fire or smoke.
- **Alert system:** Triggering alarms when fire or smoke is detected.
- Challenges and Solutions
- **False positives:** Environmental factors like fog can cause false alarms. Combining thermal imaging with visual data can reduce this.
- **Computational efficiency:** Real-time detection requires optimization to ensure quick processing. Techniques like model pruning and quantization can help.

Case Studies and Applications Several studies have demonstrated the effectiveness of these methods:

- **Forest fire detection**: Using CNNs to detect smoke in satellite images and integrating with environmental data for robust detection.
- **Urban fire detection:** Real-time systems deployed in surveillance cameras for early detection in urban areas (MDPI) (Machine Learning Projects).

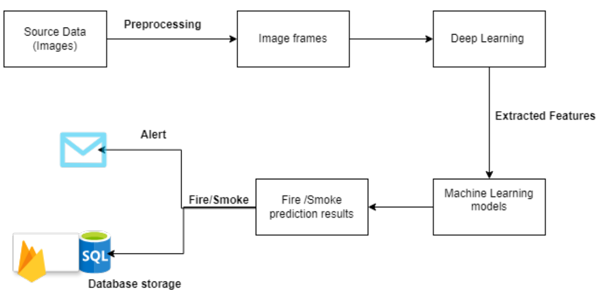


Figure 5.1: System Overview

By leveraging the strengths of both deep learning and traditional pattern recognition, fire and smoke detection systems can achieve high accuracy and reliability, providing timely alerts and enhancing safety measures. The system overview depicted in Figure 5.1 provides a comprehensive view of the workflow of our machine learning model designed for fire and smoke detection. It begins with several passive components, including data preprocessing, feature engineering, and model selection scripts. These components play a crucial role in preparing the input data for effective analysis.

The input data, typically consisting of images or videos, undergoes initial preprocessing steps to standardize and enhance its quality. For images, preprocessing often involves resizing them to a uniform size and normalizing pixel values to a standard range (e.g., [0, 1]). This normalization step ensures that the model can learn effectively without being affected by differences in pixel intensity. Videos are processed by splitting them into individual frames, a crucial step that allows the model to analyze each frame independently for the presence of fire or smoke.

After preprocessing, the next step involves feature engineering. This process extracts meaningful features from the preprocessed data that are relevant to the task of fire and smoke detection. Traditional computer vision techniques such as edge detection, color analysis (e.g., using HSV or YUV color spaces), and texture analysis may be employed to extract features that can help distinguish between normal scenes and those containing fire or smoke.

Once the data is preprocessed and feature-engineered, it is fed into our custom deep learning model. This model is specifically designed to extract high-level features from the input frames using convolutional neural networks (CNNs). CNNs are particularly well-suited for this task because they can automatically learn hierarchical representations of the input data, capturing both spatial and temporal dependencies in images and videos.

The deep learning model outputs a feature vector known as bottleneck features. These features serve as a compact representation of the essential characteristics of the input frames. By compressing the information into bottleneck features, the model reduces the dimensionality of the data while retaining the most critical information needed for subsequent classification.

The bottleneck features are then input into a classification model. This model has been trained on labeled datasets to classify each frame as containing fire, smoke, or neither. The training process involves feeding the labeled data into the model and adjusting its parameters until it achieves a high level of accuracy in predicting the correct class label for each frame.

The classification model's performance is evaluated using various metrics such as accuracy, precision, recall, and F1-score. These metrics measure how well the model performs in detecting fire and smoke, and they help in fine-tuning the model to reduce false positives and negatives.

Upon detection of fire or smoke in a frame, the system triggers an automatic alert mechanism. This involves sending an email notification to designated stakeholders, accompanied by the corresponding image frame and a timestamp for context. The email address for notifications can be customized by the user, ensuring that the relevant parties are promptly informed in case of an emergency.

Simultaneously, the system logs an entry in a cloud-based database. This database serves multiple purposes, including incident analysis, system performance monitoring, and the storage of historical data for future reference. By maintaining a record of all detected incidents, the system enables stakeholders to review past events and identify patterns or trends over time.

Operating in real-time or near-real-time, the system continuously monitors the data stream. It leverages the cloud database to analyze incident patterns over time, aiming to improve detection accuracy and reduce false positives. The scalable and robust nature of the system enables it to handle diverse environmental conditions and scenarios, ensuring effective fire and smoke detection and response.

In conclusion, the system described in Figure 5.1 represents a comprehensive approach to fire and smoke detection using machine learning. By integrating advanced data preprocessing, custom deep learning-based feature extraction, and classification models, along with real-time alert mechanisms and cloud-based logging, the system provides a reliable solution for early detection and response to potential fire hazards.

With its ability to learn and adapt from data without being explicitly programmed, machine learning plays a pivotal role in enhancing the accuracy and efficiency of fire and smoke detection systems. As technology continues to evolve, these systems will become increasingly sophisticated, further improving their ability to protect lives and property.

Chapter 6

Usecases

A use case diagram serves as a visual representation of the interactions between actors and the functionalities of a system. Actors, depicted as stick figures or blocks outside the system boundary, represent users or external systems interacting with the system being modeled. These actors have specific roles and responsibilities within the system and may initiate or participate in various use cases. Use cases, represented as ovals within the system boundary, describe specific interactions or scenarios between actors and the system to achieve particular goals or tasks.

Each use case is named in a verb-noun format, indicating the action performed by the actor and the outcome of the interaction. Relationships between actors and use cases, depicted as lines, illustrate how actors interact with the system to perform tasks. Associations indicate communication links between actors and use cases, while generalizations represent inheritance relationships between use cases.

The system boundary defines the scope of the system being modeled, enclosing all relevant actors and use cases. Additionally, included and extended use cases represent additional behaviors or functionalities invoked or modified by primary use cases, serving to provide more comprehensive system functionality. Overall, a use case diagram provides stakeholders with a high-level overview of system interactions, aiding in requirements analysis, system design, and communication among stakeholders. A use case diagram formally depicts system interactions with users (actors) to achieve specific goals. It visually documents system requirements and facilitates communication among stakeholders.

1. Image Classification CNNs can classify images into predefined categories.

Example: Classifying images of animals into categories like cats, dogs, and birds.

2. Object Detection Identifying and locating objects within an image by drawing bounding boxes around them.

Example: Detecting pedestrians, cars, and traffic signs in self-driving car systems.

3. Image Segmentation Segmenting an image into regions by classifying each pixel into different classes.

Example: Medical image analysis where different tissues, organs, or abnormalities are segmented in MRI scans.

4. Facial Recognition Recognizing and verifying identities by comparing facial features.

Example: Security systems for unlocking devices or surveillance.

5. Activity Recognition Analyzing video frames to recognize activities or actions.

Example: Detecting actions such as running, walking, or jumping in sports analytics.

6. Image Generation Generating new images based on learned patterns from a dataset.

Example: Creating realistic images using Generative Adversarial Networks (GANs).

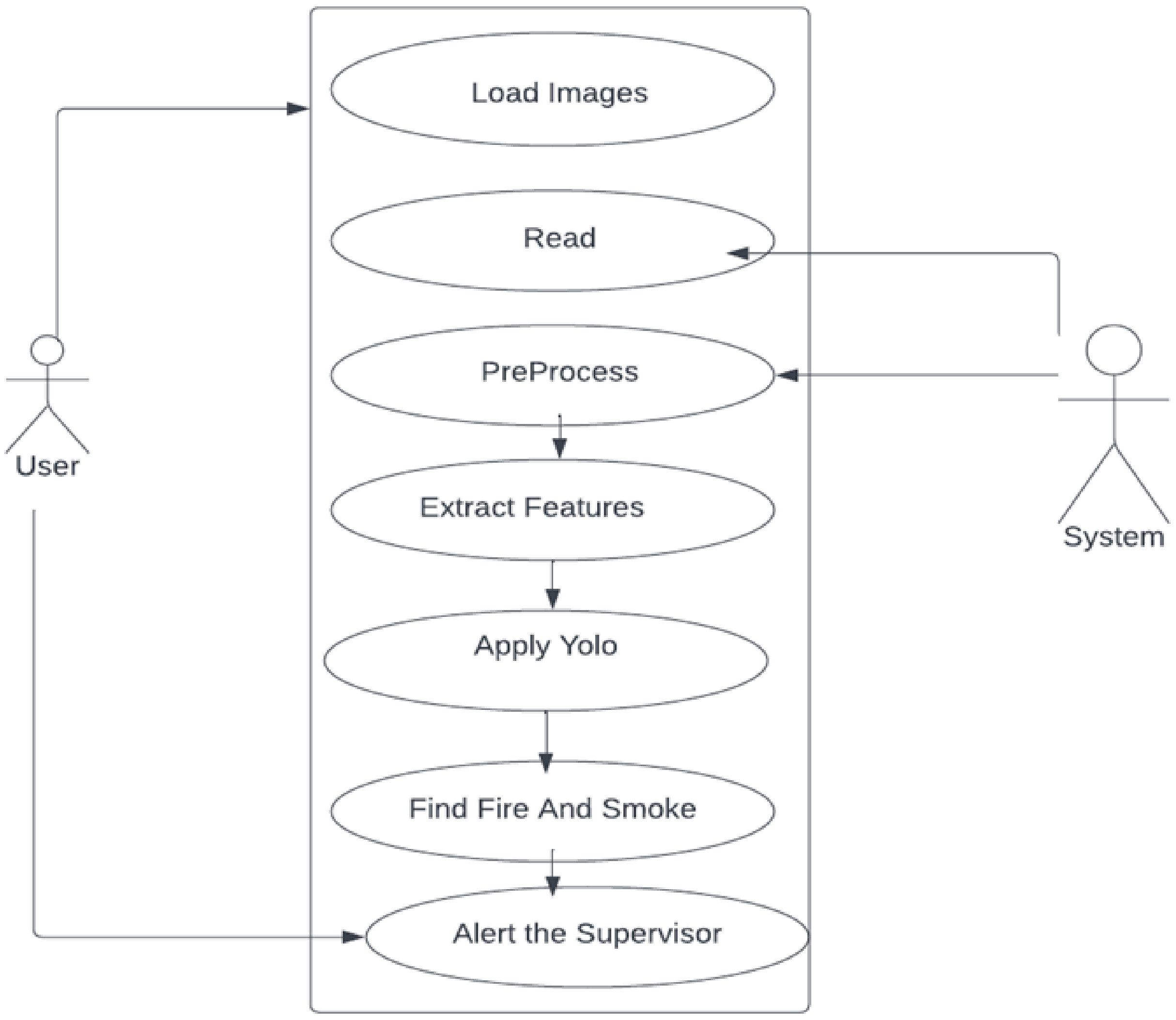


Figure 6.1: Use Case diagram

7. Style Transfer Transferring the style of one image onto another while preserving the content of the original image.

Example: Applying the artistic style of a famous painting to a photograph.

8. Super-Resolution Enhancing the resolution of an image by predicting and adding high-frequency details.

Example: Improving the quality of low-resolution images in satellite imagery.

9. Medical Diagnosis Analyzing medical images to diagnose diseases or conditions.

Example: Detecting tumors in X-ray or MRI images.

10. Autonomous Driving Processing images from vehicle-mounted cameras to make driving decisions.

Example: Lane detection, traffic sign recognition, and obstacle avoidance.

11. Agricultural Monitoring Analyzing images from drones or satellites for crop monitoring and disease detection.

Example: Identifying stressed plants and estimating crop yields.

12. Remote Sensing Processing satellite or aerial imagery for various applications.

Example: Land cover classification, environmental monitoring, and disaster assessment.

13. Retail and E-commerce Improving customer experience through visual search and recommendation systems.

Example: Allowing customers to search for products using images instead of text.

14. Anomaly Detection Detecting unusual patterns or anomalies in images or video streams.

Example: Identifying defective products on a production line.

Chapter 7

Data Flow Diagram

7.1 Level 0 DFD

A Data Flow Diagram (DFD) is a visual representation of how data flows within a system. It typically consists of processes, data stores, data flows,and external entities. Processes represent activities or transformationsthat occur within the system, such as calculations or data manipulations.

Data stores are repositories where data is stored within the system, like databases or files. Data flows depict the movement of data between processes, data stores, and external entities. External entities are sources or destinations of data outside the system boundary, like users or other systems.

In a DFD, processes are usually represented as circles or rectangles, data stores as squares, data flows as arrows, and external entities as rectangleswith rounded corners. The level of detail in a DFD can vary, ranging from a high-level overview of the entire system to more detailed diagrams focusing on specific aspects or subsystems.

DFDs are valuable tools for understanding the flow of data within a system, identifying potential bottlenecks or inefficiencies, and communicating system requirements to stakeholders. They serve as a

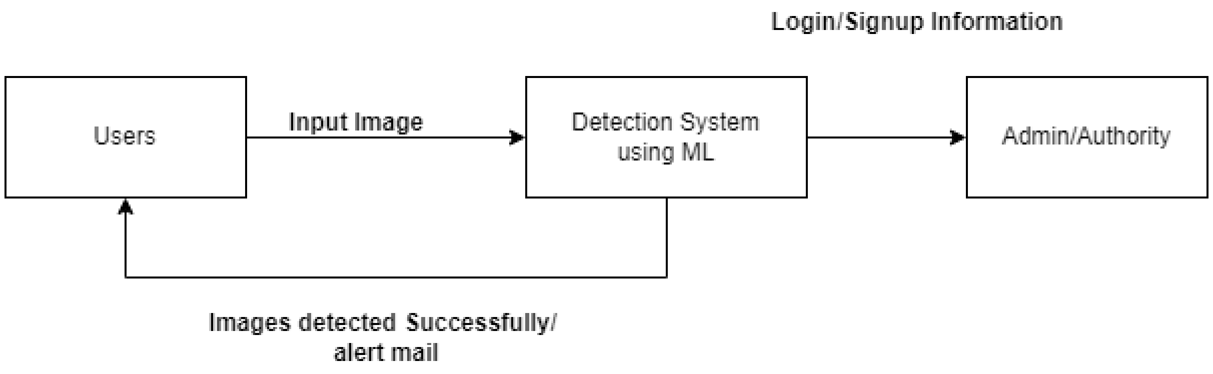

Login/Signup Information
Users
Input Image
Detection System using ML
Admin/Authority
Images detected Successfully/ alert mail

Figure 7.1: DFD 0

blueprint for designing or improving systems, helping to ensure that data is processed and managed effectively throughout its lifecycle.

It is often referred to as a context diagram. This diagram offers a high level view, showcasing the system as a singular process and its interactions with external entities. The entire voting system is depicted as one cohesive unit, with arrows indicating the flow of input and output data.

One of the methodologies employed in the development of a fire and smoke detction system is the DFD (data flow diagram). This diagram visualizes the primary processes of the system and the alternative pathways that facilitate the internal movement of data

7.2 Level 1 DFD

At the Level 1 Data Flow Diagram (DFD) stage, the focus is on providing a high-level overview of the entire system, breaking it down into major processes and depicting the flow of data between them. Typically, a Level 1 DFD will show the main processes, data stores, and external entities involved in the system without delving into detailed subprocesses.

For example, in a simple online shopping system, the Level 1 DFD

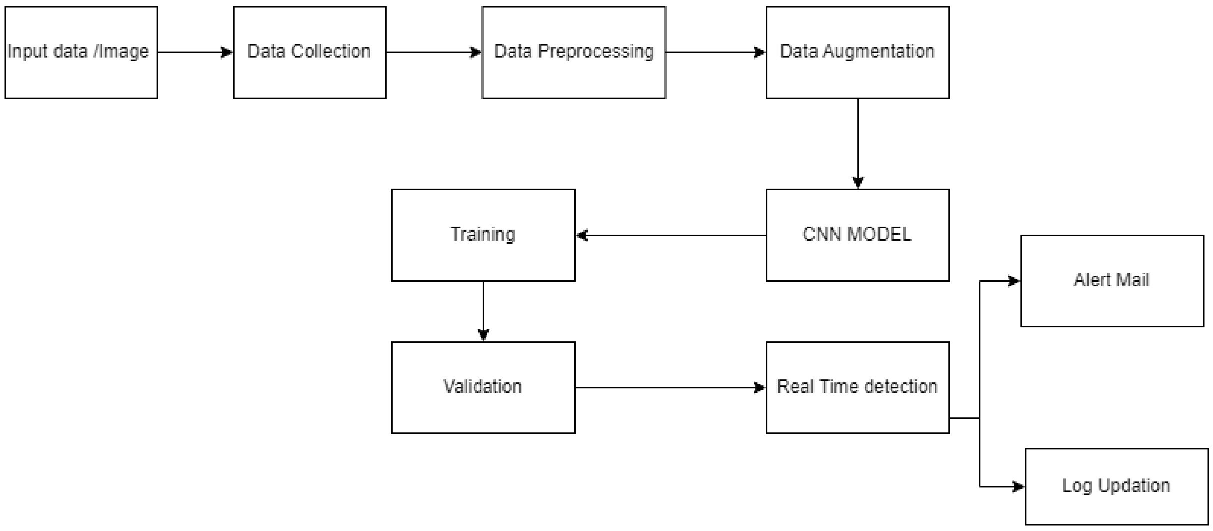


Figure 7.2: DFD 1

might include processes such as "Customer Order Processing," "Inventory Management," and "Payment Processing." Data stores could represent databases where customer information and product inventory are stored.External entities might include "Customer" and "Payment Gateway."

Arrows would illustrate how data flows between these elements. For instance, data might flow from the "Customer" entity to the "Customer Order Processing" process when an order is placed. Then, data could flow from "Customer Order Processing" to "Inventory Management" to update product availability and to "Payment Processing" to process payment details. Finally, confirmation or updates might flow back to the "Customer"entity.

Overall, a Level 1 DFD provides a simplified yet comprehensive view of the system's major components and their interactions, laying the groundwork for more detailed analysis and design in subseque

Chapter 8

Database, Schema and Test Cases

8.1 Database Design

Database: Adatabase is a structured collection of data that is organized and stored in a way that facilitates efficient retrieval, updating, and man agement. It serves as a central repository for storing and managing vast amounts of data, making it accessible to users and applications as needed. Databases are fundamental components of modern information systems, used across various industries and domains, including finance, healthcare, e-commerce, and more.

Types of Databases:

1. Relational Databases: Relational databases organize data into tables consisting of rows and columns, with each row representing a record and each column representing a data attribute. They use structured query language (SQL) to manage and manipulate data, enabling com plex queries and transactions.

2. NoSQL Databases: NoSQL databases, or "Not Only SQL," are de signed to handle unstructured or semi-structured data and offer flex ible schemas. They are well-suited for handling large volumes of data and supporting distributed architectures.

3. Object-Oriented Databases: Object-oriented databases store data as objects, combining data and behavior (methods) into a single entity. They are ideal for applications with complex data structures and relationships.

4. Graph Databases: Graph databases model data as nodes, edges, and properties, making them suitable for representing and querying highly interconnected data, such as social networks or recommendation sys tems.

Schema: Aschemaisalogical description of the structure of a database, defining its organization, relationships, constraints, and integrity rules. It serves as a blueprint for designing and managing the database, ensuring consistency and coherence in data storage and retrieval.

Types of Schema:

1. Physical Schema: The physical schema defines how data is stored on the underlying storage devices, including details such as file orga nization, indexing methods, and data compression techniques. It is concerned with optimizing data storage and retrieval performance.

2. Logical Schema: The logical schema describes the logical structure of the database, including tables, columns, relationships, and con straints. It abstracts away the physical implementation details, focus ing on data modeling and organization from a conceptual perspective.

3. Conceptual Schema: The conceptual schema provides a high-level, abstract view of the entire database, representing entities, attributes, and relationships without specifying implementation details. It serves as a conceptual framework for understanding the overall structure and purpose of the database.

Importance of Schema:

1. Data Integrity: A well-defined schema enforces data integrity constraints, ensuring that data is accurate, consistent, and reliable. Constraints such as primary keys, foreign keys, and check constraints help maintain data quality and prevent inconsistencies.

2. Data Independence: Schema abstraction separates the logical view of the data from its physical representation, enabling changes to the database structure without affecting application programs or data access methods. This promotes data independence and facilitates system evolution and maintenance.

3. Interoperability: A standardized schema promotes interoperability by enabling data exchange and integration across different systems and platforms.It facilitates data sharing and collaboration between disparate applications and environments.

In summary, databases and schemas are foundational concepts in the field of data management, providing the framework for organizing, storing, and accessing data effectively. Understanding their principles and practices is essential for designing, implementing, and maintaining robust and scalable information systems.

8.2 Database Schema

The database schema for a fire and smoke detection system defines the structure of the database, including the tables, fields, data types, and relationships between the tables. Below is a detailed description of each table and the overall structure.

8.2.1 Tables and Fields

1. Location Table

- **LocationID** (Primary Key): A unique identifier for each location. This is an integer that auto-increments with each new location.

- **Description:** A textual description of the location, such as "Building A, Floor 1, Room 101". This is a string field with a maximum length of 100 characters.
- **Address:** The physical address of the location. This is a string field with a maximum length of 200 characters.

2.Sensor Table

- **SensorID** (Primary Key): A unique identifier for each sensor. This is an integer that auto-increments with each new sensor.
- **Type**: The type of sensor, such as "Smoke", "Heat", or "Flame". This is a string field with a maximum length of 50 characters.
- **LocationID** (Foreign Key): A reference to the LocationID in the Location table, indicating where the sensor is installed. This enforces a relationship between the Sensor and Location tables.
- **Status:** The current status of the sensor, such as "Active" or "Inactive". This is a string field with a maximum length of 10 characters.
- **LastMaintenanceDate:** The date when the sensor was last maintained. This is a date field.
- **InstallationDate:** The date when the sensor was installed. This is a date field.

2.User Table

- **UserID** (Primary Key): A unique identifier for each user. This is an integer that auto-increments with each new user.
- **Name:** The name of the user. This is a string field with a maximum length of 100 characters.

– **Role:** The role of the user within the system, such as ”Admin”, ”Firefighter”, or ”Maintenance”. This is a string field with a maximum length of 50 characters.

– ContactInfo: The contact information for the user, such as an email address or phone number. This is a string field with a maximum length of 100 characters.

3.Alarm Table

– **AlarmID** (Primary Key): A unique identifier for each alarm. This is an integer that auto-increments with each new alarm.

– **SensorID** (Foreign Key): A reference to the SensorID in the Sensor table, indicating which sensor triggered the alarm. This enforces a relationship between the Alarm and Sensor tables.

– **AlarmType:** The type of alarm, such as ”Fire” or ”Smoke”. This is a string field with a maximum length of 50 characters.

– **TriggeredAt:** The date and time when the alarm was triggered. This is a datetime field.

– **IsActive:** A boolean field indicating whether the alarm is currently active.

– **AcknowledgedBy** (Foreign Key): A reference to the UserID in the User table, indicating which user acknowledged the alarm. This enforces a relationship between the Alarm and User tables.

4. MaintenanceRecord Table

– **MaintenanceID** (Primary Key): A unique identifier for each maintenance record. This is an integer that auto-increments with each new maintenance record.

– **SensorID** (Foreign Key): A reference to the SensorID in the Sensor table, indicating which sensor was maintained. This enforces a relationship between the MaintenanceRecord and Sensor tables.

– **PerformedBy** (Foreign Key): A reference to the UserID in the User table, indicating which user performed the maintenance. This enforces a relationship between the MaintenanceRecord and User tables.

8.3 Indexes and Constraints

Indexes

Indexes are special data structures associated with tables in a database that improve the speed of data retrieval operations. An index can be created on one or more columns of a table, and it allows the database to find rows much faster than it could without the index.

Types of Indexes:

1.Primary Index: This type of index is automatically created when a primary key is defined for a table. Its main purpose is to enforce the uniqueness and non-null constraints of the primary key column(s). The primary index ensures that each row in the table can be uniquely identified by the primary key and facilitates fast access to specific rows based on this key. This is essential for maintaining data integrity and efficient query performance in relational databases.

2.Unique Index: Unlike a primary index, a unique index ensures that all values in the indexed column(s) are unique across the table. It allows for fast lookup of rows based on these unique values and enforces uniqueness constraints on columns that are not part of the primary key. A table can have multiple unique indexes, and these can be created on single or multiple columns. Unique indexes are particularly useful for maintaining data integrity by preventing duplicate values in specified columns.

3.Composite Index: Also known as a composite key, this type of index is created on multiple columns of a table. It is used when queries often involve conditions on multiple columns, allowing the database to retrieve data more efficiently. A composite index can include up to 32 columns and is created by concatenating the values of the indexed columns into a single index key. This enables the database to search and retrieve data based on the combined values of these columns.

4.Full-text Index: This specialized type of index is designed for large text fields, such as VARCHAR or TEXT columns, and enables efficient text search and retrieval operations. Unlike standard indexes, which work well with exact matches, full-text indexes support searches for keywords and phrases within text data. They use advanced algorithms to tokenize and index the words within the text, allowing for fast and accurate text searches. Full-text indexes are particularly useful in applications requiring extensive search capabilities, such as content management systems and search engines.

5.Clustered Index: Unlike other indexes, a clustered index directly affects the physical order of the table data. It reorders the rows of the table to match the order of the index key. A table can have only one clustered index because the data rows themselves are stored in the order specified by the clustered index key. This means that the clustered index determines the physical order of the data pages on disk. Clustered indexes are beneficial for queries that retrieve ranges of data or require ordered results, as they minimize the need for sorting and provide efficient access to data.

Constraints Constraints are rules that the database enforces to improve data integrity. You can specify all of the following constraints at either the column level or at the table level in the PointBase RDBMS.

1. Unique Constraint A unique constraint forces a column to contain only unique values.

2. Referential Constraint You can use a referential constraint to link foreign key columns with primary key columns.

3. Check constraint The body of a check constraint is a search condition. You can use a check constraint to make sure that a value going into a column meets the criteria of the search condition.

8.3.1 Data Migration

Data migration is the process of transferring data from one storage system, format, or environment to another. It involves moving data from legacy systems, outdated formats, or disparate sources to newer systems,updated formats, or consolidated platforms. Data migration is a critical aspect of system upgrades, technology modernization initiatives, cloud adoption, and organizational restructuring efforts.

Steps for Data Migration

1. Planning and Analysis

- **Assessment of Current System:** Evaluate the existing fire and smoke detection system to understand its data structure, types of data stored, and the overall architecture.

- **Define Requirements:** Identify what needs to be migrated, including sensor data, historical logs, user data, system configurations, and any other relevant information.

- **Risk Assessment:** Identify potential risks and develop mitigation strategies. Consider data integrity, downtime, and security issues.

2. Data Mapping and Transformation

- **Data Mapping:** Map data fields from the source system to the destination system. Ensure that each piece of data has a corresponding field in the new system.
- **Data Transformation:** Convert data into the required format for the new system. This may involve changing data types, normalizing values, and ensuring compatibility.

3. **Data Extraction**
   - **Backup Data:** Always back up the data before starting the extraction process to prevent data loss.
   - **Extract Data:** Use ETL (Extract, Transform, Load) tools to pull data from the source system. Ensure that the data extraction process is thorough and captures all necessary information.
4. **Data Loading**
   - **Load Data into the New System:** Carefully load the transformed data into the new system. Use batch processing or real-time data migration methods based on the requirements and system capabilities.
   - **Validation:** Check for data integrity, completeness, and accuracy during and after the loading process.
5. **Testing and Validation**
   - **Functional Testing:** Ensure that the new system correctly interprets the migrated data and that all functionalities work as expected.
   - Performance Testing: Validate that the system performs well under load and that the data migration hasn't introduced performance bottlenecks.

- **User Acceptance Testing (UAT):** Have end-users verify that the system meets their needs and expectations.

6. Go-Live and Monitoring

- **Cutover Plan:** Develop a detailed cutover plan to transition from the old system to the new system with minimal downtime.
- **Monitoring:** Continuously monitor the new system for any issues, errors, or performance issues post-migration.
- **Support:** Provide support to users and address any problems that arise after going live.

8.3.2 Data import

Data import in a fire and smoke detection system involves integrating external data into the system to enhance its functionality, improve decision-making, or update existing information. This process is crucial when integrating new sensors, updating configurations, or importing historical data for analysis.

Data backup and recovery are crucial components of maintaining a reliable fire and smoke detection system. Effective backup strategies ensure that data is protected against loss, while recovery procedures enable quick restoration of system functionality after an incident. Here's a detailed guide on implementing data backup and recovery for such systems:

8.4 Data Backup

1. Planning and Strategy

- **Identify Critical Data:** Determine which data is essential to back up. This includes sensor data, configuration settings, user data, logs, and any other critical information.

- **Backup Frequency:** Decide how often backups should be performed (e.g., daily, weekly, hourly). The frequency depends on the criticality of the data and the acceptable data loss window.
- **Backup Types:**
  * **Full Backup:** A complete backup of all data.
  * **Incremental Backup:** Only the data that has changed since the last backup.
  * **Differential Backup:** Only the data that has changed since the last full backup.

2. **Backup Methods**
   - **On-Site Backups:** Store backups on local storage devices such as external hard drives, NAS (Network Attached Storage), or dedicated backup servers.
   - **Off-Site Backups:** Store backups in a different physical location to protect against site-specific disasters. This can include cloud storage solutions.
   - **Cloud Backups:** Utilize cloud services (e.g., AWS, Google Cloud, Microsoft Azure) for remote, scalable, and easily accessible backups.
3. **Automation and Scheduling**
   - **Automate Backups:** Use backup software or scripts to automate the backup process, reducing the risk of human error.
   - **Schedule Backups:** Configure backup schedules to ensure regular and timely backups, minimizing data loss in case of an incident.
4. **Encryption and Security**

- **Encrypt Backups:** Ensure that backup data is encrypted to protect sensitive information from unauthorized access.
- **Access Control:** Restrict access to backup data to authorized personnel only.

5. **Verification and Testing**
   - **Verify Backups:** Regularly verify backups to ensure they are complete and not corrupted.
   - **Test Restorations:** Periodically perform test restorations to verify that backups can be successfully restored and that data integrity is maintained.

8.5 Data Recovery

1. **Recovery Planning**
   - **Develop a Recovery Plan:** Create a detailed plan outlining the steps to be taken in the event of data loss. This should include contact information for key personnel, recovery procedures, and priorities.
   - **Define Recovery Objectives:**
     * **Recovery Time Objective (RTO):** The maximum acceptable time to restore the system after a disruption.
     * **Recovery Point Objective (RPO):** The maximum acceptable amount of data loss measured in time.
2. **Recovery Procedures**
   - **Initiate Recovery Process:** When data loss occurs, initiate the recovery process as per the plan.

  - **Restore Data:** Use backup data to restore the system to its last known good state. Depending on the type of backup, this may involve restoring a full backup first, followed by incremental or differential backups.
  - **System Verification:** Verify that the system is fully functional and that data integrity is maintained after restoration.

3. **Post-Recovery Steps**
   - **Incident Analysis:** Analyze the cause of data loss and implement measures to prevent future occurrences.
   - **Update Recovery Plan:** Review and update the recovery plan based on lessons learned during the recovery process.
   - **Communication:** Inform stakeholders of the recovery status and any actions taken to prevent future incidents.

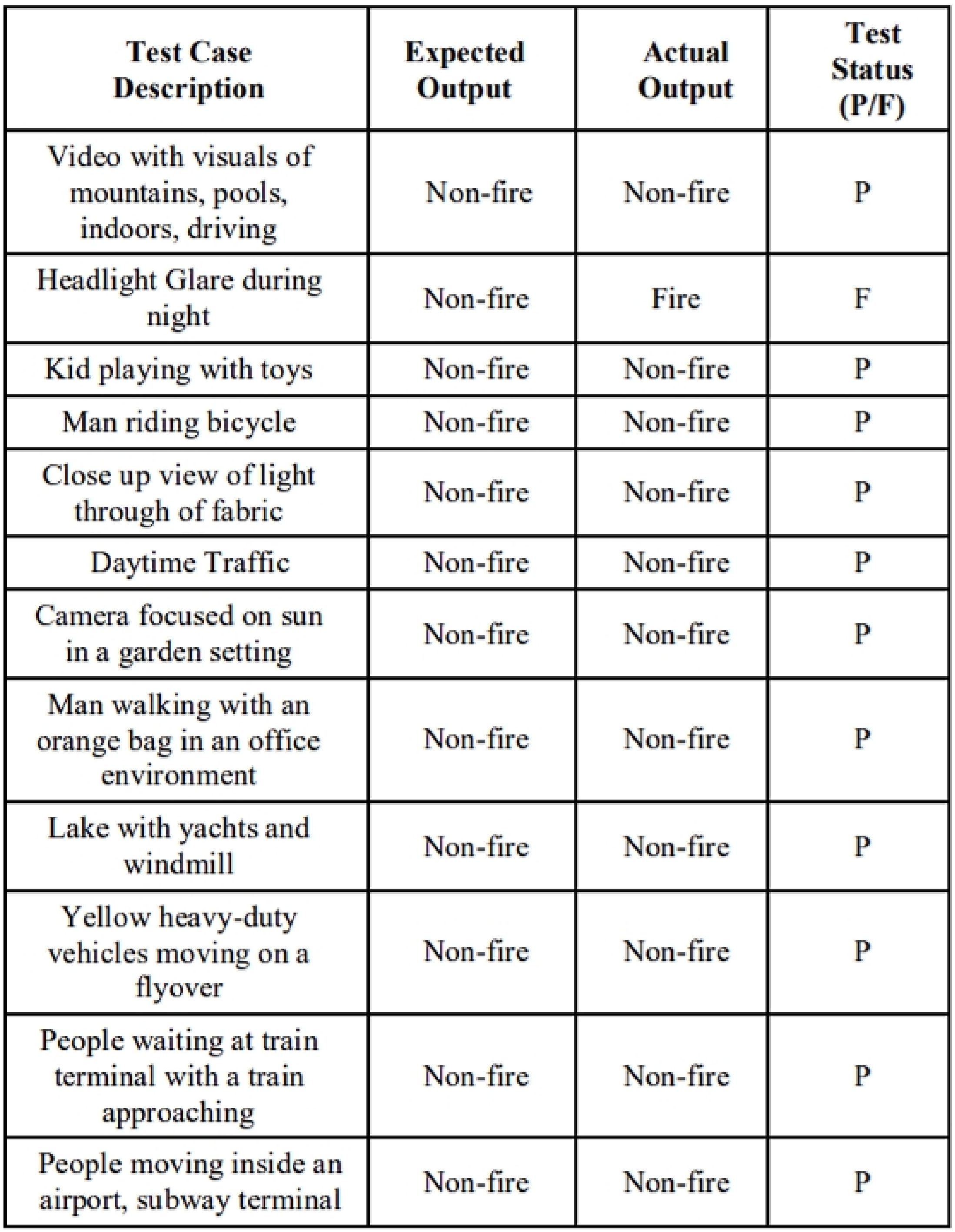
8.6 Test cases

Here are the generated test cases from the fire and non-fire images.

In Figures 8.1 and 8.2, the tables provide a detailed comparison between the expected output and the actual output generated by a machine learning model across a range of scenarios. These tables are crucial tools for evaluating the model's performance in tasks such as fire and smoke detection from images or videos.

For each scenario listed in the table, the expected output represents the ground truth or the correct label that the model should ideally predict. This is based on human annotations or other established methods that define what constitutes the presence or absence of fire and smoke.

In contrast, the actual output indicates what the model predicted for each scenario based on the features it extracted and processed



Test Case Description	Expected Output	Actual Output	Test Status (P/F)
Video with visuals of mountains, pools, indoors, driving	Non-fire	Non-fire	P
Headlight Glare during night	Non-fire	Fire	F
Kid playing with toys	Non-fire	Non-fire	P
Man riding bicycle	Non-fire	Non-fire	P
Close up view of light through of fabric	Non-fire	Non-fire	P
Daytime Traffic	Non-fire	Non-fire	P
Camera focused on sun in a garden setting	Non-fire	Non-fire	P
Man walking with an orange bag in an office environment	Non-fire	Non-fire	P
Lake with yachts and windmill	Non-fire	Non-fire	P
Yellow heavy-duty vehicles moving on a flyover	Non-fire	Non-fire	P
People waiting at train terminal with a train approaching	Non-fire	Non-fire	P
People moving inside an airport, subway terminal	Non-fire	Non-fire	P

Fig. 8.1 Non- fire Videos Testing Result.

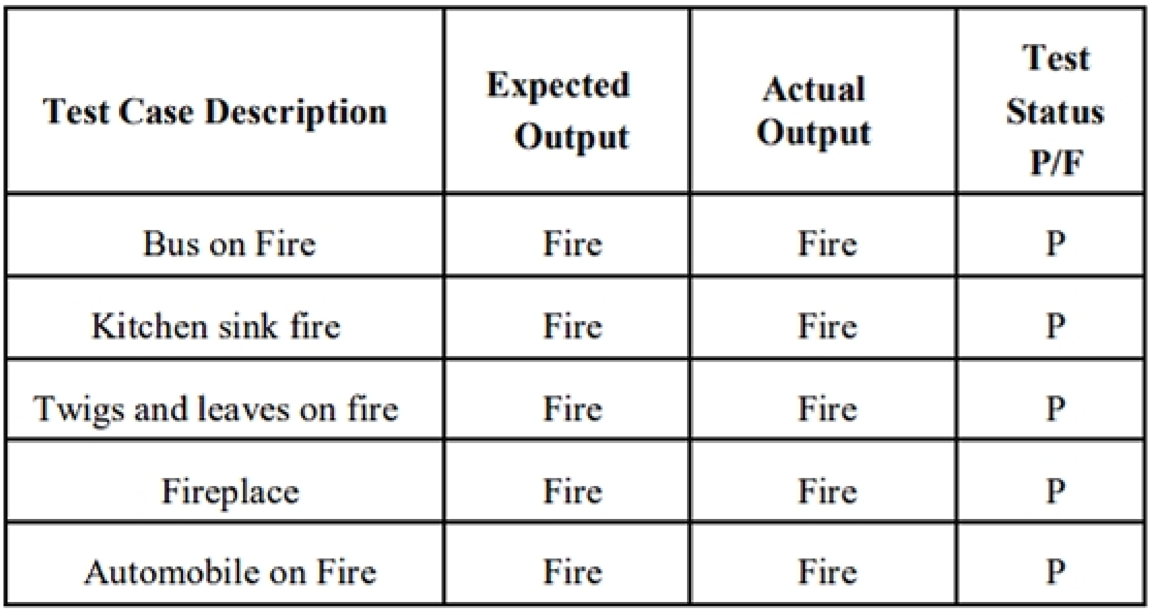
from the input data. This comparison allows researchers to assess how accurately the model performs in identifying fire and smoke in different situations.

The tables play a pivotal role in evaluating the model's accuracy and effectiveness. They offer a succinct summary of where the model succeeded in correctly identifying fire or smoke and where it may have struggled or made errors. These insights are crucial for pinpointing areas where the model needs improvement.

Moreover, the tables facilitate the calculation of performance metrics such as accuracy, precision, recall, and F1-score. Accuracy measures how often the model is correct in its predictions, while precision and recall provide insights into the model's ability to avoid false positives and false negatives, respectively. The F1-score balances precision and recall, offering a single metric to evaluate the model's overall performance.

In the realm of machine learning, such evaluations are indispensable for refining models and enhancing their capabilities over time. By carefully analyzing the results presented in Figures 8.1 and 8.2, researchers and engineers can make informed decisions about adjustments to the model architecture, the optimization of feature extraction methods, or modifications to the training dataset itself.

Overall, these tables serve as comprehensive visual representations of the model's performance in distinguishing between different situations related to fire and smoke. They provide valuable insights into the model's strengths and weaknesses, offering a roadmap for further improvements and advancements in the field of computer vision and machine learning.



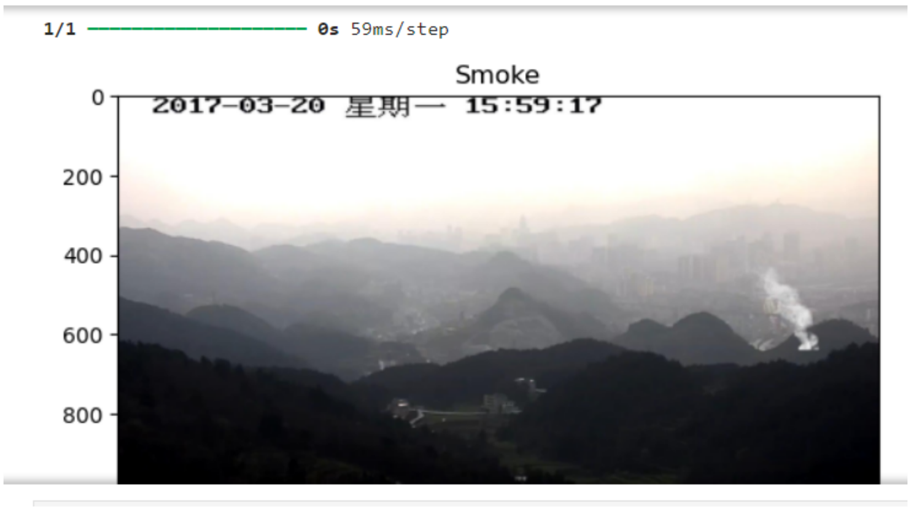
Test Case Description	Expected Output	Actual Output	Test Status P/F
Bus on Fire	Fire	Fire	P
Kitchen sink fire	Fire	Fire	P
Twigs and leaves on fire	Fire	Fire	P
Fireplace	Fire	Fire	P
Automobile on Fire	Fire	Fire	P

Fig. 8.2 Fire Videos Testing Result

Chapter 9

Snapshots

Here are the snapshots of the output



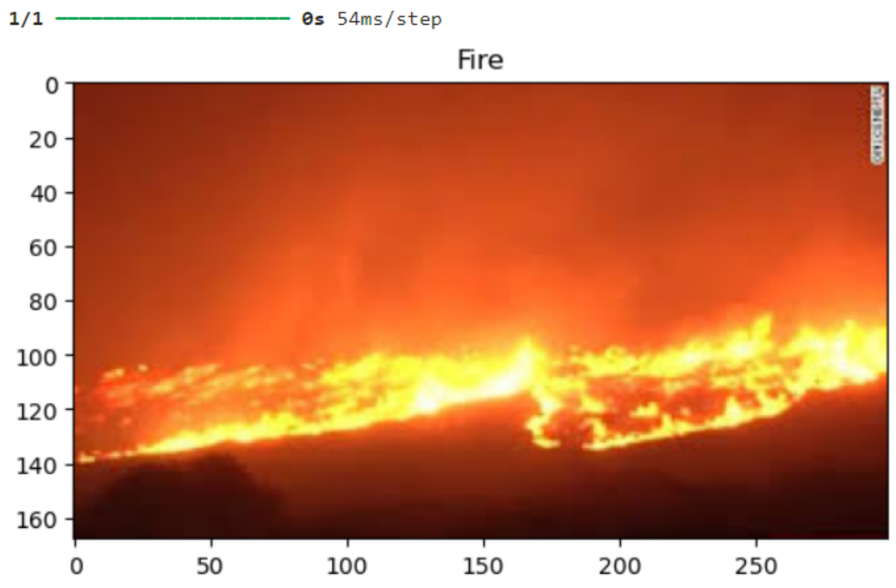
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 59ms/step
```

Figure 9.1: Smoke Detected

This is the smoke generated detected images formed on detection

Figure 9.2: Non-Fire Image

This is the image showing non-fire in the presence of normal lights



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 54ms/step
```

Figure 9.3: Fire Detected

This is the fire image detected from the detection

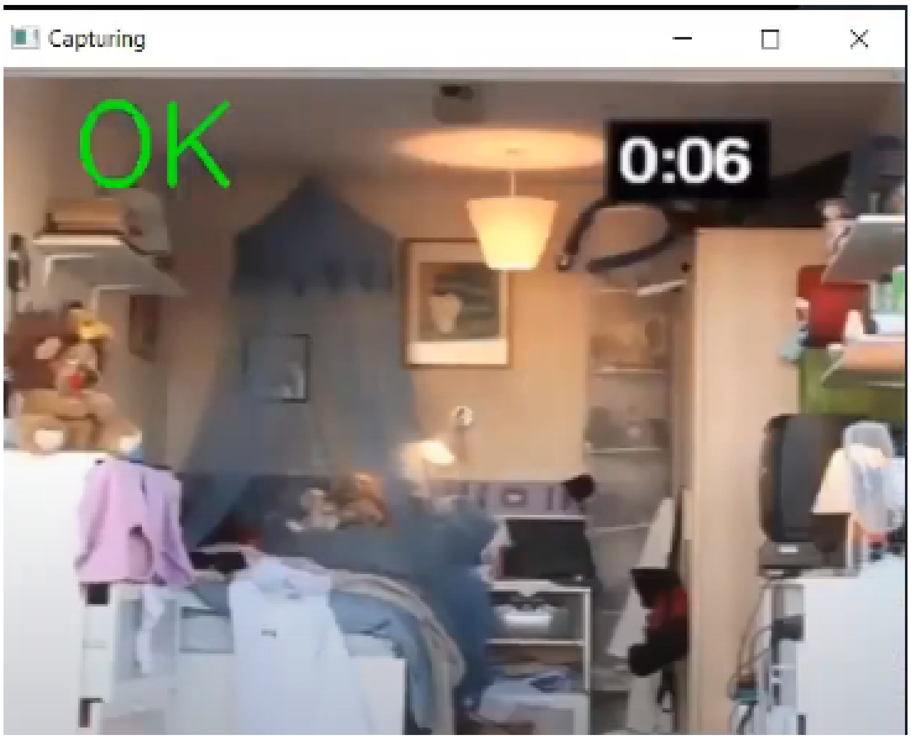


Figure 9.4: Before Fire Snap

This is the image just before the fire was occured.

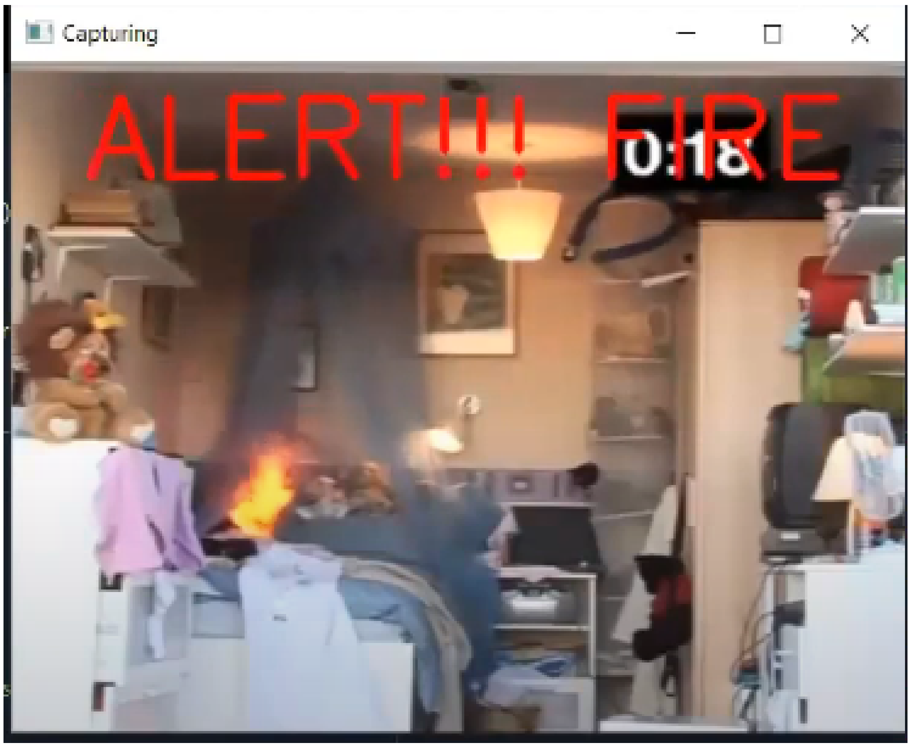


Figure 9.5: After Fire Occurred Snap

This the image after just the fire occurs.

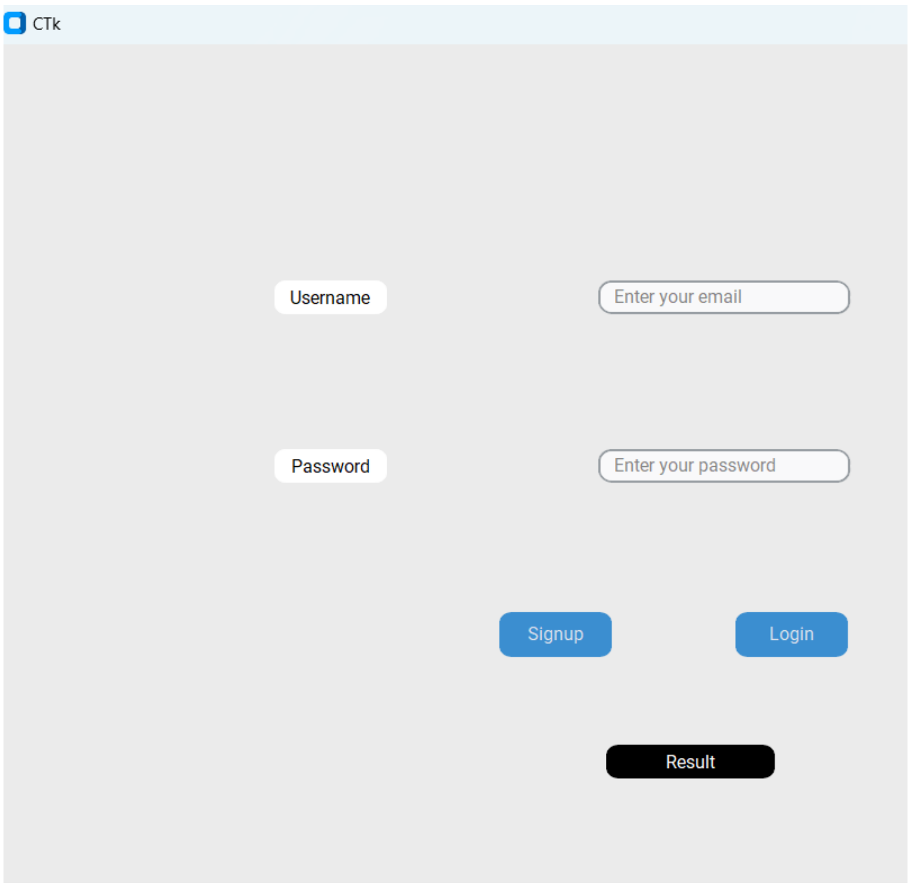
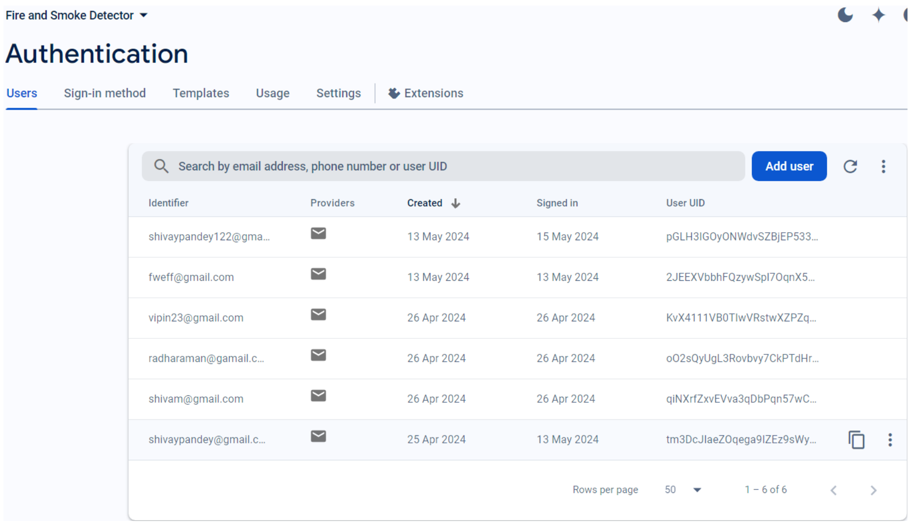


Figure 9.6: User Login

This the User interface where user can login or signup accordingly.



Identifier	Providers	Created	Signed in	User UID
shivaypandey122@gma...	✉	13 May 2024	15 May 2024	pGLH3IGOyONWdvSZBjEP533...
fweff@gmail.com	✉	13 May 2024	13 May 2024	2JEEXVbbhFQzywSpI7OqnX5...
vipin23@gmail.com	✉	26 Apr 2024	26 Apr 2024	KvX4111VB0TIwVRstwXZPZq...
radharaman@gamail.c...	✉	26 Apr 2024	26 Apr 2024	oO2sQyUgL3Rovbvy7CkPTdHr...
shivam@gmail.com	✉	26 Apr 2024	26 Apr 2024	qiNXrfZxvEVva3qDbPqn57wC...
shivaypandey@gmail.c...	✉	25 Apr 2024	13 May 2024	tm3DcJlaeZOqega9IZEz9sWy...

Rows per page 50 1 – 6 of 6

Figure 9.7: Firebase Stored Data

This is the login credential stored on the database using firebase.

Chapter 10

Conclusion

The present decade is marked by huge strides in areas of processing, computation and algorithms. This has enabled great progress in many fields including processing of surveillance video streams for recognizing abnormal or unusual events and actions. Fire accidents have caused death and destruction all over the world, consuming countless lives and causing billions in damages. This implies that developing an accurate, early, affordable fire-detection system is imperative Therefore, we have proposed a fire detection model for videos/video frames using transfer learning for deep learning. The models make use of ResNet-50, InceptionV3 and Inception-ResNet-V2 models to extract the features and various ML algorithms such as SVM, Logistic Regression, Naive Bayes and Decision Tree on the extracted features to detect fire in video frames. Coming to the application on the whole, it works in real-time and has the ability to send alertemails along with offering a user-friendly graphical interface. It's cost-effective, reliable, robust, accurate compared to existing opto-electronic hardware and software-based systems in the market.

In this paper, we showed that very high classification performance can be achieved using deep CNNs, even when limited data is provided. The overfitting problem, caused by a limited set of image data for training, leads to poor performance in neural network models. To solve this problem, we enlarge our training sets using various data augmentation techniques

Future Scope

The application can be enhanced by training the model with a larger dataset consisting of fires at various stages and dimensions. With higher GPU memory, we could use two deep learning models for feature extraction, whose output feature vectors are concatenated and classified to offer more robustness. An R-CNN model can be used to implement fire localization along with classification. We can also expect better deep learning architectures to emerge in the future, offering better feature extraction. The application will also offer a considerably better performance when run on machines having better processing power compared to existing one of which it has been developed.

References

[**1**] Chenebert, Audrey & Breckon, Toby & Gaszczak, Anna. (2011). “A nontemporal texture driven approach to realtime fire detection”.

[**2**] Seebamrungsat, Jareerat et al. “Fire detection in the buildings using image processing.” 2014 Third ICT International Student Project Conference (ICTISPC) (2014): 95-98.

[**3**] K. Muhammad, J. Ahmad, Z. Lv, P. Bellavista, P. Yang and S. W. Baik, “Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications,” in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 7, pp. 1419-1434, July 2019.

[**4**] Z. Jiao et al., “A Deep Learning Based Forest Fire Detection Approach Using UAV and YOLOv3,” 2019 1st International Conference on Industrial Artificial Intelligence (IAI), Shenyang, China, 2019.

[**5**] Faming Gong ,1 Chuantao Li,1 Wenjuan Gong ,1 Xin Li,1 Xiangbing Yuan,2 Yuhui Ma and Tao Song. “A RealTime Fire Detection Method from Video with Multifeature Fusion”. Hindawi, Computational Intelligence and Neuroscience, Volume 2019.

[**6**] L. Shao, F. Zhu and X. Li, “Transfer Learning for Visual Categorization: A Survey,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no. 5, pp. 1019-1034, May 2015.

[**7**] S. Mohd Razmi, N. Saad and V. S. Asirvadam, “Visionbased flame detection: Motion detection & fire analysis,” 2010 IEEE Student Con-

ference on Research and Development (SCOReD), Putrajaya, 2010, pp. 187-191.

[8] T. Qiu, Y. Yan and G. Lu, “A new edge detection algorithm for flame image processing,” 2011 IEEE International Instrumentation and Measurement Technology Conference, Binjiang, 2011, pp. 1-4.

[9] M. Ligang, C. Yanjun and W. Aizhong, “Flame region detection using color and motion features in video sequences,” The 26th Chinese Control and Decision Conference (2014 CCDC), Changsha, 2014, pp. 3005-3009.

[10] Toulouse, Tom & Rossi, Lucile & Celik, Turgay & Akhloufi, Moulay. (2015). “Automatic fire pixel detection using image processing: A comparative analysis of Rulebased and Machine Learning-based methods. Signal, Image and Video Processing”.

[11] K. He, X. Zhang, S. Ren and J. Sun, “Deep Residual Learning for Image Recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.

[12] S. Wilson, S. P. Varghese, G. A. Nikhil, I. Manolekshmi and P. G. Raji, “A Comprehensive Study on Fire Detection,” 2018 Conference on Emerging Devices and Smart Systems (ICEDSS), Tiruchengode, 2018, pp. 242-246.

[13] X. Wu, X. Lu and H. Leung, “An adaptive threshold deep learning method for fire and smoke detection,” 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, 2017, pp. 1954-1959.

[14] Chenebert, Audrey & Breckon, Toby & Gaszczak, Anna. (2011). “A nontemporal texture driven approach to real-time fire detection”.

[15] Seebamrungsat, Jareerat et al. “Fire detection in the buildings using image processing.” 2014 Third ICT International Student Project

Conference (ICTISPC) (2014): 95-98.

[16] K. He, X. Zhang, S. Ren and J. Sun, “Deep Residual Learning for Image Recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.

[17] S. Wilson, S. P. Varghese, G. A. Nikhil, I. Manolekshmi and P. G. Raji, “A Comprehensive Study on Fire Detection,” 2018 Conference on Emerging Devices and Smart Systems (ICEDSS), Tiruchengode, 2018, pp. 242- 246.

[18] X. Wu, X. Lu and H. Leung, “An adaptive threshold deep learning method for fire and smoke detection,” 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, 2017, pp. 1954-1959.

[19] Chenebert, Audrey Breckon, Toby Gaszczak, Anna, (2011). “A nontemporal texture driven approach to realtime fire detection”.

[20] Seebamrungsat, Jareerat et al. “Fire detection in the buildings using image processing.” 2014 Third ICT International Student Project Conference (ICTISPC) (2014): 95-98.

Appendix A

Installation instructions for an fire and smoke detection using machine laerning may vary based on the specific technology stack and platform used. However, here's a brief and general set of instructions that you can adapt to your system:

1. Pre-requisites:

Ensure that the hosting environment meets the system requirements, including the necessary server specifications, database support, and any required software dependencies. Verify that the web server (e.g., Apache, Nginx) and database server (e.g., SQLite, PostgreSQL) are installed and configured.

2. Download and Extract:

Download the system package from the official source or repository. Extract the files to the root directory of your web server.

3. Database Setup:

Create a new database for the system and configure the database connection settings. Import the provided SQL script to set up the necessary tables and initial data.

4. Configuration:

Update the configuration files with essential settings such as database credentials, base URLs, and any environmentspecific configurations. Ensure that file permissions are correctly set for security.

5. Web Server Configuration:

Configure the web server to point to 67 the installation directory. Set up any necessary URL rewriting rules to ensure proper navigation within the application.

6. Initial Setup and Admin Account:

Access the system through a web browser. Complete the initial setup process, which may include creating the first administrator account. Ensure secure login credentials are used.

7. Backup and Recovery:

Implement a regular backup strategy for both the application files and the database. Ensure that recovery procedures are documented and readily available in case of data loss or system failure.

8. Security Measures:

Implement security best practices, including using HTTPS, configuring secure file permissions, and regularly updating the system

Appendix B

Maintenance procedures for a fire and smoke detction system integrated with machine laerning are essential to ensure its continued functionality, security, and adaptability over time. These procedures typically involve a combination of routine tasks, updates, and monitoring activities. Maintenance procedures include regular checks for system health, identifying and addressing issues promptly, updating software components, and implementing security patches. Routine tasks may include database optimization, data backup, and system performance monitoring. Software updates and enhancements are applied to keep the system aligned with technological advancements and to address any identified bugs or vulnerabilities. Security measures involve periodic security audits, ensuring that the system complies with the latest security standards, and implementing measures to protect against evolving threats. Data integrity checks are performed to ensure that applicant data is accurate and secure. In summary, maintenance procedures for an campus recruitment system integrated with machine learning encompass a proactive approach to system health, security, and user satisfaction, ensuring that the system remains robust, up-to-date, and responsive to the needs of both adminis69 trators and applicants. Regular updates, security measures, and ongoing monitoring contribute to the long-term success and efficiency of the online management system.

Troubleshooting Guidelines

Troubleshooting is a critical aspect of maintaining and resolving issues in an campus recruitment system integrated with predictive intellegence. Here are brief guidelines for troubleshooting common issues:

1. Define the Problem:

Clearly identify and define the problem or issue reported by users or system monitoring.

2. Check System Status:

Verify the overall status of the system, including server availability, database connectivity, and essential services.

3. Review Logs and Error Messages:

Examine system logs and error messages to pinpoint the source of the issue and gather relevant information for diagnosis.

4. User Reports:

Gather detailed information from users regarding the issue, including the steps leading to the problem, any error messages received, and the browser or device used.

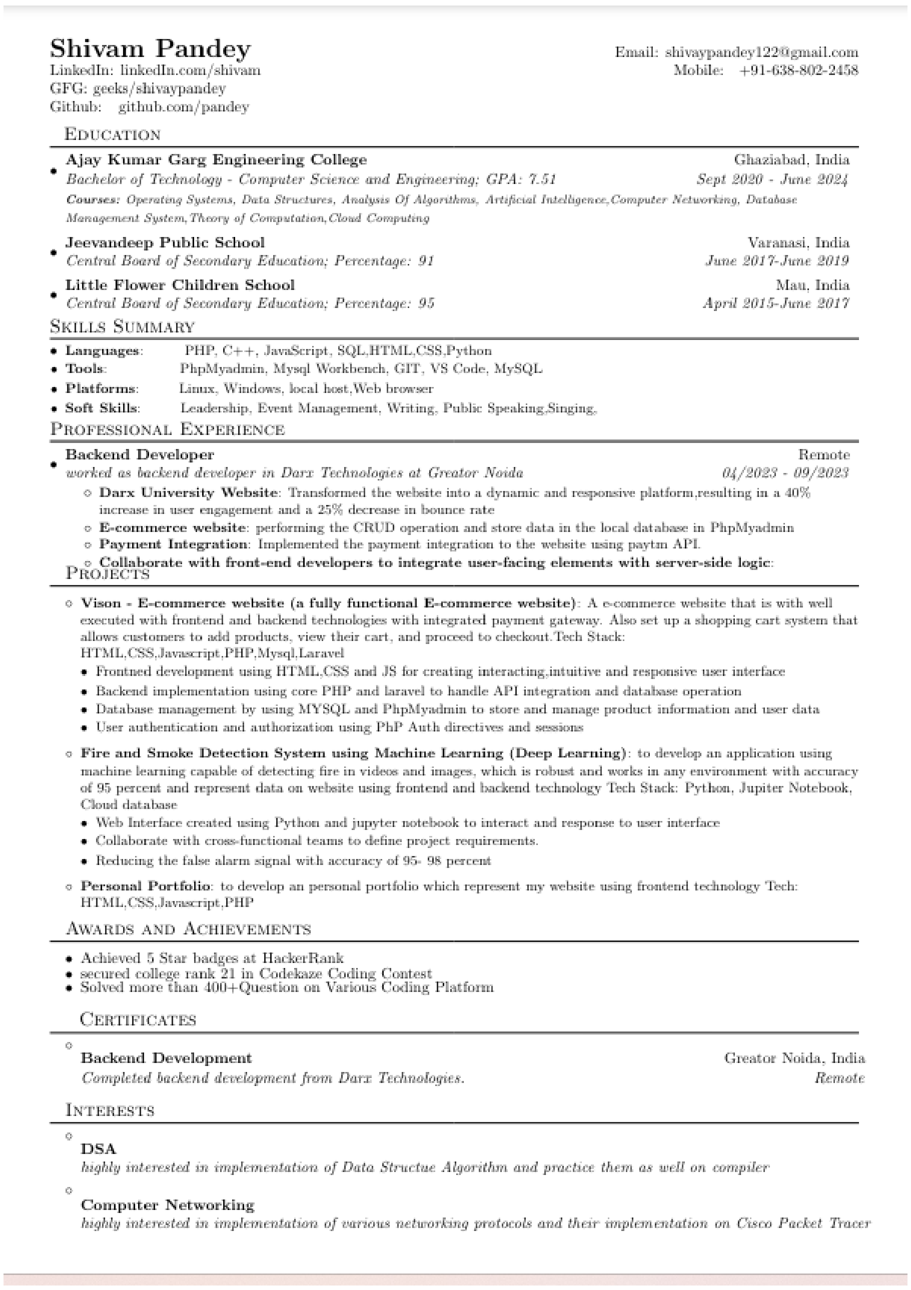
5. Isolate the Issue:

Determine whether the issue is system-wide or specific to certain modules, features, or users.

6. Collaborate with Development Team:

Engage with the development team to discuss the issue, share findings, and determine if a software bug or code-related problem exists

Appendix C



Shivam Pandey
LinkedIn: linkedIn.com/shivam
GFG: geeks/shivaypandey
Github: github.com/pandey

Email: shivaypandey122@gmail.com
Mobile: +91-638-802-2458

EDUCATION

- **Ajay Kumar Garg Engineering College** — Ghaziabad, India
  Bachelor of Technology - Computer Science and Engineering; GPA: 7.51 — *Sept 2020 - June 2024*
  ***Courses:** Operating Systems, Data Structures, Analysis Of Algorithms, Artificial Intelligence,Computer Networking, Database Management System,Theory of Computation,Cloud Computing*
- **Jeevandeep Public School** — Varanasi, India
  Central Board of Secondary Education; Percentage: 91 — *June 2017-June 2019*
- **Little Flower Children School** — Mau, India
  Central Board of Secondary Education; Percentage: 95 — *April 2015-June 2017*

SKILLS SUMMARY

- **Languages**: PHP, C++, JavaScript, SQL,HTML,CSS,Python
- **Tools**: PhpMyadmin, Mysql Workbench, GIT, VS Code, MySQL
- **Platforms**: Linux, Windows, local host,Web browser
- **Soft Skills**: Leadership, Event Management, Writing, Public Speaking,Singing,

PROFESSIONAL EXPERIENCE

- **Backend Developer** — Remote
  worked as backend developer in Darx Technologies at Greator Noida — *04/2023 - 09/2023*
  - **Darx University Website**: Transformed the website into a dynamic and responsive platform,resulting in a 40% increase in user engagement and a 25% decrease in bounce rate
  - **E-commerce website**: performing the CRUD operation and store data in the local database in PhpMyadmin
  - **Payment Integration**: Implemented the payment integration to the website using paytm API.
  - **Collaborate with front-end developers to integrate user-facing elements with server-side logic**:

PROJECTS

- **Vison - E-commerce website (a fully functional E-commerce website)**: A e-commerce website that is with well executed with frontend and backend technologies with integrated payment gateway. Also set up a shopping cart system that allows customers to add products, view their cart, and proceed to checkout.Tech Stack: HTML,CSS,Javascript,PHP,Mysql,Laravel
  - Frontned development using HTML,CSS and JS for creating interacting,intuitive and responsive user interface
  - Backend implementation using core PHP and laravel to handle API integration and database operation
  - Database management by using MYSQL and PhpMyadmin to store and manage product information and user data
  - User authentication and authorization using PhP Auth directives and sessions
- **Fire and Smoke Detection System using Machine Learning (Deep Learning)**: to develop an application using machine learning capable of detecting fire in videos and images, which is robust and works in any environment with accuracy of 95 percent and represent data on website using frontend and backend technology Tech Stack: Python, Jupiter Notebook, Cloud database
  - Web Interface created using Python and jupyter notebook to interact and response to user interface
  - Collaborate with cross-functional teams to define project requirements.
  - Reducing the false alarm signal with accuracy of 95- 98 percent
- **Personal Portfolio**: to develop an personal portfolio which represent my website using frontend technology Tech: HTML,CSS,Javascript,PHP

AWARDS AND ACHIEVEMENTS

- Achieved 5 Star badges at HackerRank
- secured college rank 21 in Codekaze Coding Contest
- Solved more than 400+Question on Various Coding Platform

CERTIFICATES

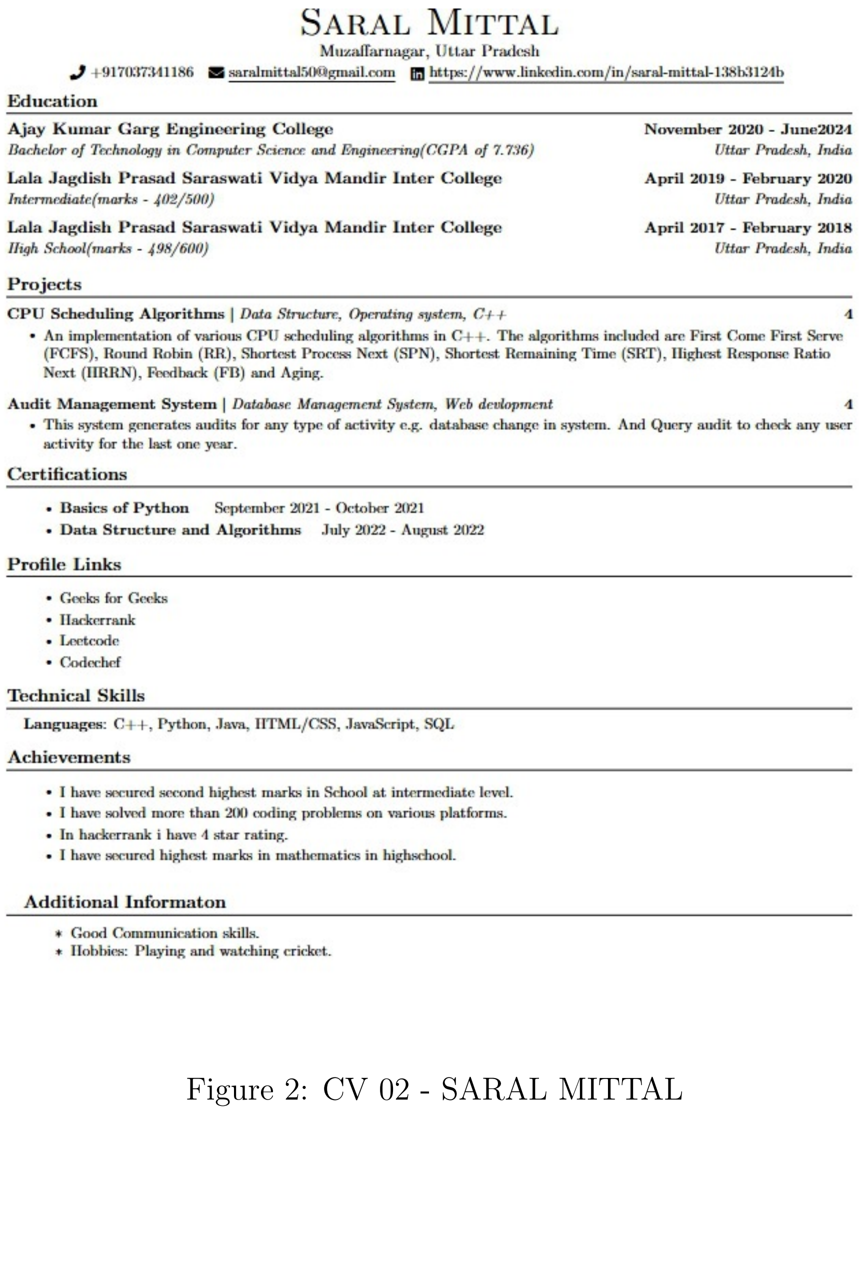
- **Backend Development** — Greator Noida, India
  Completed backend development from Darx Technologies. — *Remote*

INTERESTS

- **DSA**
  highly interested in implementation of Data Structue Algorithm and practice them as well on compiler
- **Computer Networking**
  highly interested in implementation of various networking protocols and their implementation on Cisco Packet Tracer

Figure 1: CV 01 - SHIVAM PANDEY

Appendix D



SARAL MITTAL

Muzaffarnagar, Uttar Pradesh

+917037341186 saralmittal50@gmail.com https://www.linkedin.com/in/saral-mittal-138b3124b

Education

Ajay Kumar Garg Engineering College **November 2020 - June2024**
Bachelor of Technology in Computer Science and Engineering(CGPA of 7.736) *Uttar Pradesh, India*

Lala Jagdish Prasad Saraswati Vidya Mandir Inter College **April 2019 - February 2020**
Intermediate(marks - 402/500) *Uttar Pradesh, India*

Lala Jagdish Prasad Saraswati Vidya Mandir Inter College **April 2017 - February 2018**
High School(marks - 498/600) *Uttar Pradesh, India*

Projects

CPU Scheduling Algorithms | *Data Structure, Operating system, C++* 4

- An implementation of various CPU scheduling algorithms in C++. The algorithms included are First Come First Serve (FCFS), Round Robin (RR), Shortest Process Next (SPN), Shortest Remaining Time (SRT), Highest Response Ratio Next (HRRN), Feedback (FB) and Aging.

Audit Management System | *Database Management System, Web devlopment* 4

- This system generates audits for any type of activity e.g. database change in system. And Query audit to check any user activity for the last one year.

Certifications

- **Basics of Python** September 2021 - October 2021
- **Data Structure and Algorithms** July 2022 - August 2022

Profile Links

- Geeks for Geeks
- Hackerrank
- Leetcode
- Codechef

Technical Skills

Languages: C++, Python, Java, HTML/CSS, JavaScript, SQL

Achievements

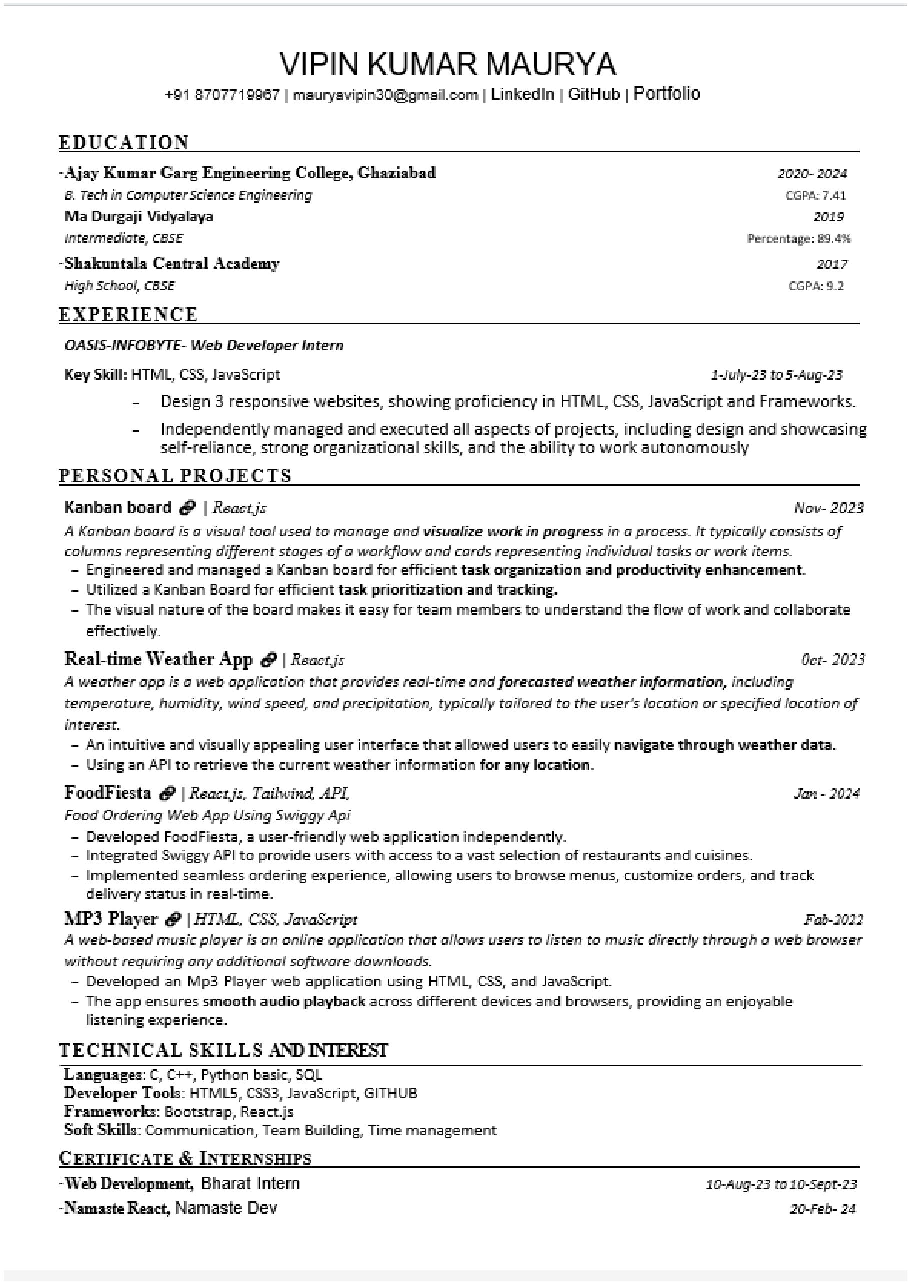
- I have secured second highest marks in School at intermediate level.
- I have solved more than 200 coding problems on various platforms.
- In hackerrank i have 4 star rating.
- I have secured highest marks in mathematics in highschool.

Additional Informaton

* Good Communication skills.
* Hobbies: Playing and watching cricket.

Figure 2: CV 02 - SARAL MITTAL

Appendix E



VIPIN KUMAR MAURYA

+91 8707719967 | mauryavipin30@gmail.com | LinkedIn | GitHub | Portfolio

EDUCATION

-**Ajay Kumar Garg Engineering College, Ghaziabad**	*2020- 2024*
B. Tech in Computer Science Engineering	CGPA: 7.41
Ma Durgaji Vidyalaya	*2019*
Intermediate, CBSE	Percentage: 89.4%
-**Shakuntala Central Academy**	*2017*
High School, CBSE	CGPA: 9.2

EXPERIENCE

OASIS-INFOBYTE- Web Developer Intern

Key Skill: HTML, CSS, JavaScript *1-July-23 to 5-Aug-23*

- Design 3 responsive websites, showing proficiency in HTML, CSS, JavaScript and Frameworks.
- Independently managed and executed all aspects of projects, including design and showcasing self-reliance, strong organizational skills, and the ability to work autonomously

PERSONAL PROJECTS

Kanban board | *React.js* *Nov- 2023*

*A Kanban board is a visual tool used to manage and **visualize work in progress** in a process. It typically consists of columns representing different stages of a workflow and cards representing individual tasks or work items.*

- Engineered and managed a Kanban board for efficient **task organization and productivity enhancement**.
- Utilized a Kanban Board for efficient **task prioritization and tracking.**
- The visual nature of the board makes it easy for team members to understand the flow of work and collaborate effectively.

Real-time Weather App | *React.js* *Oct- 2023*

*A weather app is a web application that provides real-time and **forecasted weather information**, including temperature, humidity, wind speed, and precipitation, typically tailored to the user's location or specified location of interest.*

- An intuitive and visually appealing user interface that allowed users to easily **navigate through weather data.**
- Using an API to retrieve the current weather information **for any location**.

FoodFiesta | *React.js, Tailwind, API,* *Jan - 2024*

Food Ordering Web App Using Swiggy Api

- Developed FoodFiesta, a user-friendly web application independently.
- Integrated Swiggy API to provide users with access to a vast selection of restaurants and cuisines.
- Implemented seamless ordering experience, allowing users to browse menus, customize orders, and track delivery status in real-time.

MP3 Player | *HTML, CSS, JavaScript* *Fab-2022*

A web-based music player is an online application that allows users to listen to music directly through a web browser without requiring any additional software downloads.

- Developed an Mp3 Player web application using HTML, CSS, and JavaScript.
- The app ensures **smooth audio playback** across different devices and browsers, providing an enjoyable listening experience.

TECHNICAL SKILLS AND INTEREST

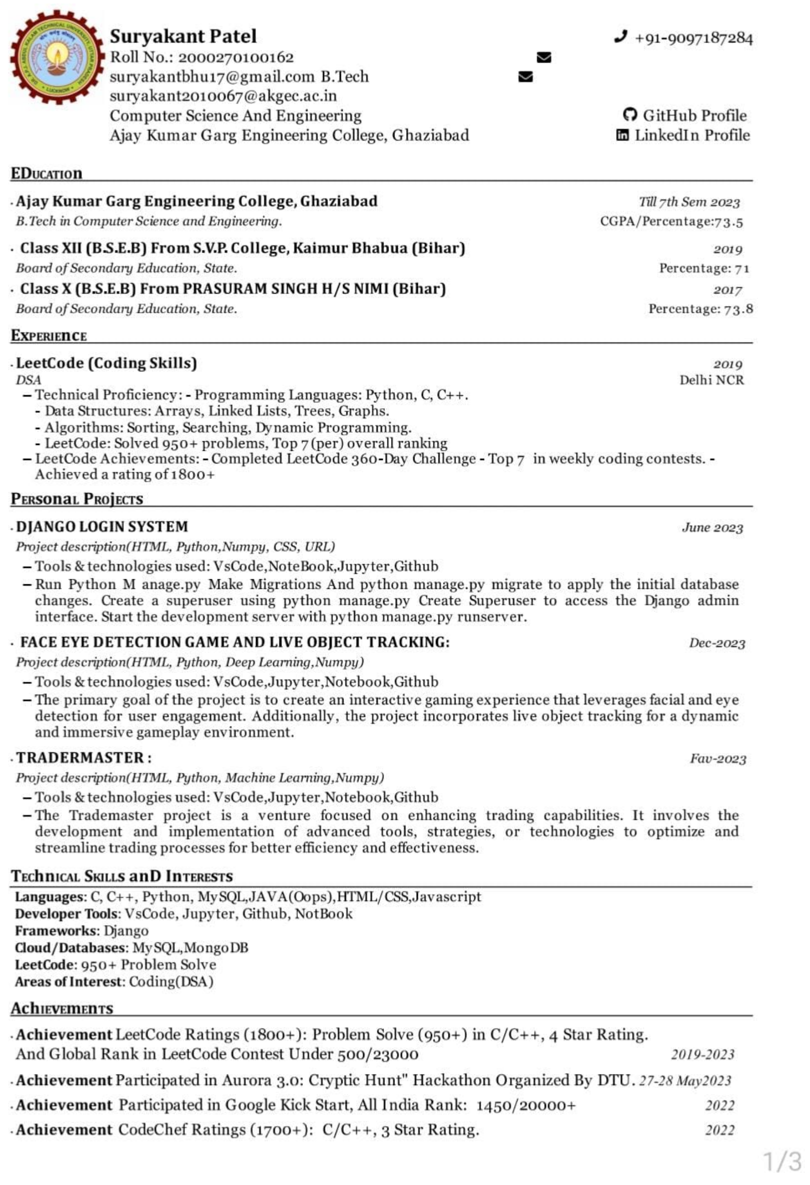
Languages: C, C++, Python basic, SQL
Developer Tools: HTML5, CSS3, JavaScript, GITHUB
Frameworks: Bootstrap, React.js
Soft Skills: Communication, Team Building, Time management

CERTIFICATE & INTERNSHIPS

-**Web Development,** Bharat Intern	*10-Aug-23 to 10-Sept-23*
-**Namaste React,** Namaste Dev	*20-Feb- 24*

Figure 3: CV-03 VIPIN KUMAR MAURYA

Appendix F



Suryakant Patel
Roll No.: 2000270100162
suryakantbhu17@gmail.com B.Tech
suryakant2010067@akgec.ac.in
Computer Science And Engineering
Ajay Kumar Garg Engineering College, Ghaziabad

+91-9097187284
GitHub Profile
LinkedIn Profile

EDucation

· **Ajay Kumar Garg Engineering College, Ghaziabad** *Till 7th Sem 2023*
B.Tech in Computer Science and Engineering. CGPA/Percentage:73.5

· **Class XII (B.S.E.B) From S.V.P. College, Kaimur Bhabua (Bihar)** *2019*
Board of Secondary Education, State. Percentage: 71

· **Class X (B.S.E.B) From PRASURAM SINGH H/S NIMI (Bihar)** *2017*
Board of Secondary Education, State. Percentage: 73.8

Experience

· **LeetCode (Coding Skills)** *2019*
DSA Delhi NCR
- Technical Proficiency: - Programming Languages: Python, C, C++.
  - Data Structures: Arrays, Linked Lists, Trees, Graphs.
  - Algorithms: Sorting, Searching, Dynamic Programming.
  - LeetCode: Solved 950+ problems, Top 7 (per) overall ranking
- LeetCode Achievements: - Completed LeetCode 360-Day Challenge - Top 7 in weekly coding contests. - Achieved a rating of 1800+

Personal Projects

· **DJANGO LOGIN SYSTEM** *June 2023*
Project description(HTML, Python,Numpy, CSS, URL)
- Tools & technologies used: VsCode,NoteBook,Jupyter,Github
- Run Python M anage.py Make Migrations And python manage.py migrate to apply the initial database changes. Create a superuser using python manage.py Create Superuser to access the Django admin interface. Start the development server with python manage.py runserver.

· **FACE EYE DETECTION GAME AND LIVE OBJECT TRACKING:** *Dec-2023*
Project description(HTML, Python, Deep Learning,Numpy)
- Tools & technologies used: VsCode,Jupyter,Notebook,Github
- The primary goal of the project is to create an interactive gaming experience that leverages facial and eye detection for user engagement. Additionally, the project incorporates live object tracking for a dynamic and immersive gameplay environment.

· **TRADERMASTER :** *Fav-2023*
Project description(HTML, Python, Machine Learning,Numpy)
- Tools & technologies used: VsCode,Jupyter,Notebook,Github
- The Trademaster project is a venture focused on enhancing trading capabilities. It involves the development and implementation of advanced tools, strategies, or technologies to optimize and streamline trading processes for better efficiency and effectiveness.

Technical Skills anD Interests

Languages: C, C++, Python, MySQL,JAVA(Oops),HTML/CSS,Javascript
Developer Tools: VsCode, Jupyter, Github, NotBook
Frameworks: Django
Cloud/Databases: MySQL,MongoDB
LeetCode: 950+ Problem Solve
Areas of Interest: Coding(DSA)

Achievements

· **Achievement** LeetCode Ratings (1800+): Problem Solve (950+) in C/C++, 4 Star Rating.
And Global Rank in LeetCode Contest Under 500/23000 *2019-2023*

· **Achievement** Participated in Aurora 3.0: Cryptic Hunt" Hackathon Organized By DTU. *27-28 May2023*

· **Achievement** Participated in Google Kick Start, All India Rank: 1450/20000+ *2022*

· **Achievement** CodeChef Ratings (1700+): C/C++, 3 Star Rating. *2022*

1/3

Figure 4: CV 04 - SURYAKANT PATEL