Report-Project Topic 3 Camera Based Occupancy Sensing

by User System

Submission date: 29-Aug-2024 11:14PM (UTC+0800)
Submission ID: 2440530189
File name: Report-Project_Topic_3_Camera_Based_Occupancy_Sensing.docx (3.19M)
Word count: 8980
Character count: 53910

List of Tables

List of Figures

Chapter 03

Chapter 05

Chapter 06

Chapter 07

Chapter 08

Camera-Based Occupancy Sensing Using AI for Smart Home Applications

Chapter 2

2.1 Introduction:

Since the onset of modern housing technology, it has undergone serious changes that facilitate innovation compliant with user's application, safety, and other factors. One example is that AI-based Camera-based Occupancy Sensing leads the way in capturing data on the presence and absence of individuals within homes. Our project utilizes state-of-the-art Convolutional Neural Networks (CNNs) such as ResNet50, MobileNet V3, EfficientNet B0, MobileNet V2, and MobileNet V3 Small to come up with an advanced real-time occupancy detection system. [1]

To develop this system, we employed various data types of videos taken from different sources including treating hospitals and some other spaces to train the model comprehensively. Our computational configuration was furnished with robust processing and graphics capabilities that supplied adequate facilities for demanding image processing operations. A complex optimization technique was used for optimizing our models that improved their precision and functioning dramatically.

A sophisticated simulation environment was created to replicate real-life conditions, allowing our system to undergo rigorous testing and refinement. Throughout this process, we encountered several challenges. To address inconsistencies in performance under varying lighting conditions, we implemented advanced pre-processing techniques and employed a diverse range of training datasets. Additionally, we optimized our code for parallel processing and utilized GPU acceleration to overcome computational constraints. To manage the variability in occupancy scenarios, we expanded the dataset and applied data augmentation methods, ensuring the system's robustness and adaptability.

Therefore, some future improvements lie in integrating advanced sensors like thermal imaging in order to improve accuracy; developing scalable solutions for larger datasets and complex environments as well as introducing continuous learning frameworks that will help adjust to changing occupancy patterns. With these improvements, our Camera-Based Occupancy Sensing system will become even more effective and adaptable thus fulfilling the requirements of contemporary smart homes.

Chapter 3

3.1 Resource List:

Hardware Resources: Here, the hardware components support the infrastructure required for training the models.

Table 1: Hardware components utilized in the project

SL	Product Description	Qty	Unit Price ($)	Amount ($)
1.	Intel 13th Gen Core i5-13500 Raptor Lake Processor, 14 Cores, 20 Threads, 2.5GHz-4.8GHz, 24MB Cache, Intel UHD 770	1	239.13	239.13
2.	MSI MAG B760M Mortar DDR5 LGA1700 Gaming Motherboard	1	183.62	183.62
3.	Team T-Force Delta TUF Gaming RGB 16GB 6000MHz DDR5 RAM #FF5D516G6000HC38A01	2	67.47	134.94
4.	MaxGreen 1200VA Offline UPS (Plastic Body)	1	52.10	52.10
5.	Gigabyte C301 Glass E-ATX Gaming Case (Black)	1	57.22	57.22
6.	MSI GeForce RTX 4060 Ti Ventus 3X 16G OC GDDR6 Graphics Card	1	555.13	555.13
7.	Samsung 980 Pro 500GB PCIe Gen4 M.2 NVMe SSD	1	73.45	73.45
8.	CoolerMaster MWE 750-Watt Gold V2 Power Supply #MPE-7501-AFAAG-IN	1	999.23	999.23
9.	DeepCool LE520 240mm All-in-One ARGB Liquid CPU Cooler	1	56.37	56.37
			Total =	2351.19 $

For advanced computing, various hardware components were used in this project. Among them is a 13th-generation multi-core processor that can do intensive work due to its advanced threading and cache capabilities. As for the motherboard, it supports DDR5 memory for quick and efficient data processing. On the other hand, RAM sticks with high speeds improve the reactivity of the system while offline UPS provides backup power so that one does not suffer power outages. Components fit in a spacious gaming case which also contains a high-end graphics card responsible for complex visual processing. To access information swiftly, there is an NVMe SSD installed in addition to other storage requirements and high-quality UPS gives rise to consistent power supply. A liquid CPU cooler helps maintain optimum temperatures by managing heat during peak operations. [2]

Software Resources:

Table 2: Software components employed in the project

Tool	Functions	Why we selected this tool
Python Language	Programming language for project implementation.	Widely adopted in AI/ML projects with a rich ecosystem of libraries and resources.
PyTorch Deep Learning Framework	Framework for developing and training neural networks.	Offers flexibility, ease of use, and strong community support for deep learning.
Pandas	Library for data manipulation and analysis.	Efficiently manages large datasets and supports comprehensive data preprocessing.
Matplotlib/Seaborn	Libraries for visualizing data.	Provides tools for creating clear and insightful visual representations of data and results.
Jupyter Notebook	Interactive coding environment.	Enables quick prototyping, testing, and easy sharing of code and findings.
Adam Optimizer	Algorithm for optimizing neural network training.	Recognized for fast convergence and adaptive learning rates, enhancing model performance.
CrossEntropyLoss	Loss function for classification tasks.	Ideal for handling multi-class classification problems, such as in waste sorting.

Various software tools have been incorporated in the project because they are very important in implementation and optimization of machine learning models. The project has been coded in a programming language that is popular about AI and machine learning projects due to its large support for libraries. Concerning its wide adaptability and ease of handling, a deep learning framework is used to develop and train neural networks. A robust library has been deployed for data manipulation that can efficiently deal with big datasets while enabling comprehensive data pre-processing. Visualization libraries have been integrated to develop clear visual representations of data including results. To accommodate rapid prototyping, testing as well as seamless sharing of code; the interactive coding environment is utilized. The process of optimizing training neural networks employs a special algorithm which is characterized by fast convergence and adaptive learning rates. A specific loss function is lastly applied when dealing with the classification tasks because it works well even in multi-class classification problems like waste sorting. [3]

3.2 Specifications of Software Block Diagram using ML (Python Programming Language)

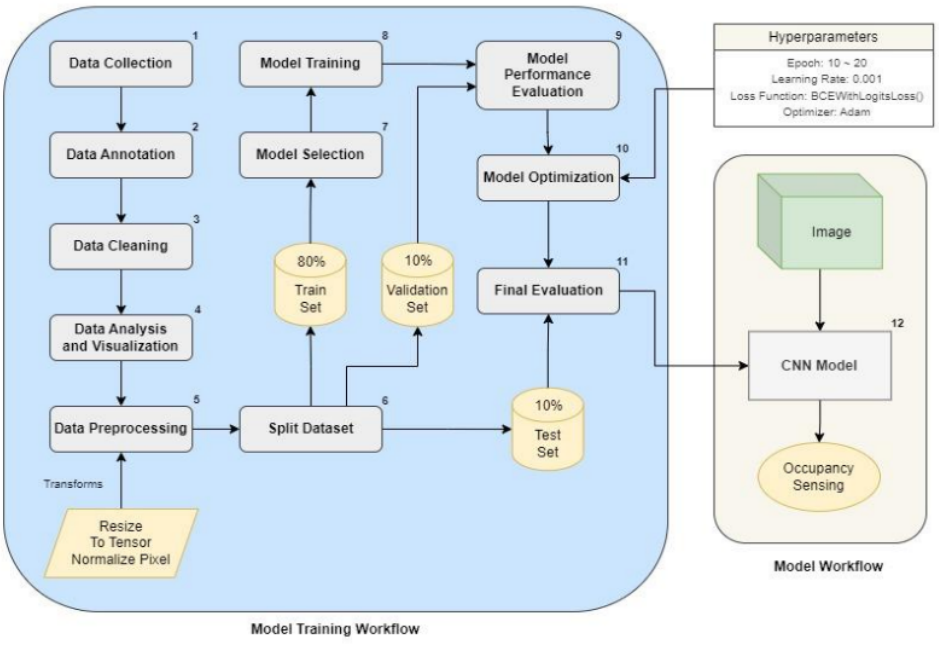

Figure 1: Software Block Diagram Specifications for ML

In terms of designing a camera-based system that senses occupancy using AI for smart homes, figure 1 represents how such systems can be built. The process starts with collection of relevant pictures that are later annotated so they can be labeled properly. To make them ready for analysis, the collected data goes through cleaning and preprocessing steps like resizing and normalizing. After breaking down the dataset into three parts which are training, validation and test sets, certain AI models are put to test through model training to determine their effectiveness in selection; one is eventually chosen as it is improved upon and assessed with regards to performance. Some hyperparameters such as learning rate optimization algorithms modify themselves in search of increasing accuracy levels. Last but not least, an artificial intelligence program gets implemented in the home automation system able to track human motion dynamics consistently thus operating so much on its own than before enabling power-saving measures within residential premises while still being safe against criminals' attacks and adding convenience to those using those facilities similar like these. [4]

Chapter 4

Chapter 5

5.1 Data Sources:

The data used in this project is sourced from two different online datasets containing images and information about human detection and medical staff tracking using CCTV footage. Now the two datasets that were used in the project are discussed below:

Dataset 1: Human Detection Dataset

The data set utilized in this project consists of CCTV images from both indoor and outdoor settings with an equal distribution of scenes containing human beings as well as those that do not. The set contains 362 images labeled "0_n.png" meaning those without humans and 559 labeled "1_n.png", meaning they have humans in them. Whereby, "n" indicates the specific number of every picture in that data set. Pictures were collected by combining several publically available YouTube CCTV footage downloads, an open-access indoor images bank, and recordings made by the users themselves. This heterogeneous data is critical to train and test the artificial intelligence models supporting occupancy sensing systems to be strong within all types of environments. Moreover, it is licensed through CC0: Public Domain License which allows anyone to use it or make changes to it at any time – this would greatly help enhance smart home applications continuously. [5]

Dataset 2: Medical Staff People Tracking

This project employs a dataset dedicated to the tracking of medical staff people to build and improve AI models capable of effectively recognizing and tracking humans in a healthcare setting. With various settings in hospitals, the dataset has many images that show some people especially those working in the medical field inside boxes that are outlined. The system aims at creating systems that will help track movements of health practitioners, monitor patient movements, measure wait times as well as evaluate overall performance in hospitals and other medical facilities.

The dataset is structured in two main folders: “images” which contain original video frames and “boxes” which contain annotation files that visually represent data related to the images in question. Additionally, there’s a .csv file that lists frame IDs together with their paths from the “images” folder, whereas the “annotations.xml” file outlines the precise coordinates of each bounding box making it possible to accurately follow individuals. This important annotation data is vital for models that separate or distinguish between classes: "doctor", and "nurse", while "others" refer to people not classified as healthcare professionals. The dataset includes 137 photos that can be found in the "images" and the "boxes" folders distributed under the Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND) License, which restricts its usage to non-commercial purposes with the original contents remaining unaltered. Within this context, this dataset will be used by this project to improve smart home applications related to health monitoring to ensure precise detection of medical staff or any other individual thereby enhancing efficient energy usage, safety, and home automation systems based on actual user presence activities. [6]

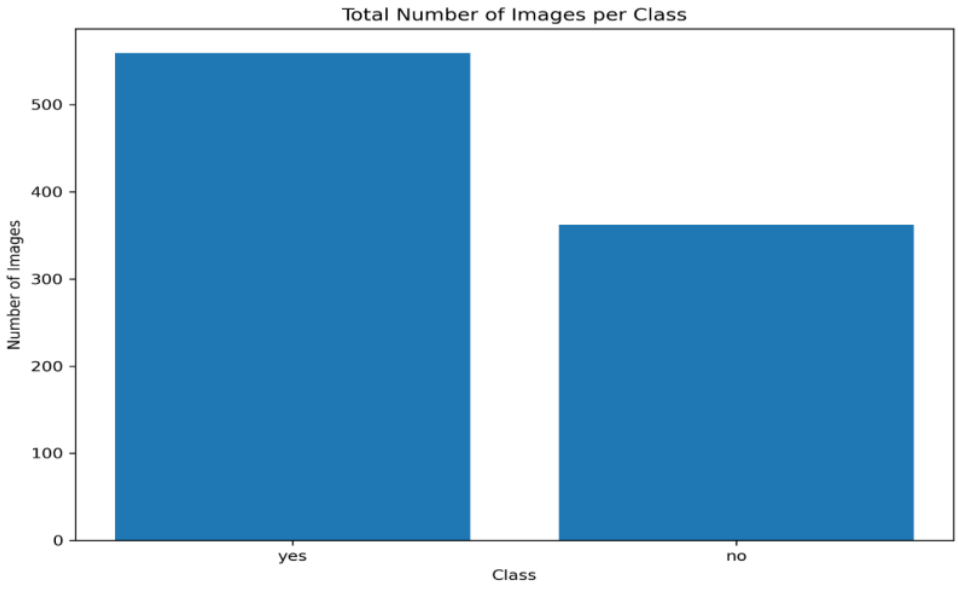

Figure 2: Total image count by category

Figure 2 depicts how photographs are split into two classes namely scenes that contain human beings (denoted by "yes") and those that do not have them (marked as "no"). When it comes to this dataset, there is a larger volume of pictures with individuals than without them. For this reason, it is important for the occupancy sensing project using camera-based technologies because it plays a huge role in training artificial intelligence models that are supposed to identify human presence within smart houses. The huge number of images containing persons leads to an extensive database for creating precise identification algorithms necessary for making proper use of power, increasing safety measures, and automating houses according to people's habits.

5.2 Data Annotation:

Data annotation is very important in making a dataset of AI models that will aid in detecting human presence in smart homes in this project. By joining two datasets from Kaggle, one possible way of making them more understandable is employing a human annotator. Each image of the dataset has to be examined and labeled in such a way that scenes that have human beings should be differentiated from those that do not have them. Wherever humans are detected, there will be their locations clearly outlined with boxes that precisely join them within each frame. Such labeling entails more than just binary classifications since they also mark particular spaces where some actions performed by humans can be seen thus enabling the model to learn refined patterns associated with human beings' occupancy. Therefore, it becomes important for an artificial intelligence system to grasp and acknowledge humanity hence enhancing energy conservation, protection, and individually controlled houses based on human beings' presence. The quality level including the correctness of these notes relates directly to how well the model can operate in practical settings hence this forms the basic groundwork of dependable as well as effective smart home solutions. [7]

5.3 Data Preprocessing:

All the datasets are prepared for AI training through several steps of data preprocessing. Image resizing, pixel normalization for better model performance and transforming data into neural network processable format are aspects of this. Because of these measures, inconsistencies are eliminated, computation load is reduced and occupancy sensing models are more accurate. Dividing data into training, validation as well as test sets is also done at this stage to make sure there are effective models trained and their evaluation is done in a very detailed manner. Now the steps are described below: [8]

- **Resizing Images:** The process begins with resizing all images in the dataset to a standardized dimension. This step ensures that the images meet the input specifications required by the neural networks utilized in the project.
- **Pixel Value Normalization:** Following resizing, the pixel values of the images are adjusted, typically scaled to fall within the range of 0 to 1. This normalization is key to improving the models' convergence speed during training, as it mitigates the effects of variations in lighting and contrast across the dataset.
- **Data Transformation:** Next, the images are converted into tensors, which is the format necessary for deep learning model computations. Additionally, data

augmentation techniques such as cropping or rotating images are applied, which improves the model's robustness by exposing it to a broader range of scenarios.

- **Dataset Splitting:** The dataset is then partitioned into three key subsets: training, validation, and test sets. The training set is used for model training, the validation set is utilized for tuning hyperparameters, and the test set is reserved for assessing the final performance of the model.
- **Consistency Assurance:** Throughout all these preprocessing steps, particular care is taken to ensure the dataset remains consistent. This involves addressing any potential biases or inconsistencies that could negatively impact the model's learning and overall accuracy.

Chapter 6

Design & Development:

6.1 Algorithm Development:

The project harnesses a convolutional neural network (CNN) model that allows transforming raw image data into useful categories that drive the automated waste sorting system. This makes waste management more efficient and accurate. CNNs have been known to yield the best results in various computer vision tasks including object detection, image classification, and segmentation. They can independently learn features from images enabling them to execute these operations with high precision which is amazing. A typical CNN has many different types of layers: convolutional layers, pooling layers, fully connected layers, input layers, and output layers. Below are the functions of these layers. [9]

- **Input Layer:** The CNN processes an input image represented as a grid of pixels. For color images, each pixel includes three color channels (red, green, and blue), whereas grayscale images contain a single channel. Note: ImageNet is a widely used dataset that provides millions of labeled images across thousands of categories for computer vision tasks.
- **Convolutional Layer:** The convolutional layer applies convolution operations using learnable filters or kernels. These kernels move across the input image, multiplying their weights with the corresponding pixels in their receptive field and summing the results to create a feature map. Multiple kernels allow the network to learn a diverse set of features.
- **Activation Function:** Following convolution, an activation function (e.g., ReLU) is applied element-wise to introduce nonlinearity into the network. This enables the CNN to learn complex patterns and relationships within the extracted features.
- **Pooling Layers:** Pooling layers reduce the spatial dimensions of the feature maps generated by the convolutional layers, which helps preserve essential information while decreasing the computational load.

- **Repetition of Convolution, Activation, and Pooling:** Subsequent layers learn increasingly abstract features by building on the representations obtained from previous layers.
- **Flattening:** After several convolutional and pooling layers, the feature maps are flattened into a one-dimensional vector, collapsing the spatial structure into a linear format.
- **Fully Connected Layers:** This vector is then passed through fully connected layers, where traditional neural network operations occur. These layers learn to classify the input based on the extracted features. The final fully connected layer provides the output, representing either class probabilities or specific values for the task.
- **Training and Optimization:** During training, the CNN's parameters (e.g., filter weights, biases) are optimized to minimize a loss function. This is achieved through backpropagation, where gradients are calculated and used to update the parameters using optimization algorithms like gradient descent. The training process refines the network's weights to enhance prediction accuracy. Note: ResNet50 is a commonly used pre-trained model for computer vision tasks, trained on the ImageNet dataset.
- **Inference:** After training, the CNN can be used to make predictions on new, unseen images. The network's forward pass generates class probabilities or other outputs based on the input image.

Different varieties of CNNs can be used for categorizing images. This project employs six CNN frameworks to develop the algorithm. Presented below is a table showing the different names of models along with their respective workflows:

Table 3: Overview of CNN Models and Associated Workflow for Image Classification.

Model Name	Model Architecture
ResNet50	~25.6 million parameters
MobileNet V3 Large	~5.4 million parameters
EfficientNet B0	~5.3 million parameters
MobileNet V2	~3.4 million parameters
MobileNet V3 Small	~2.5 million parameters

To train and develop CNN models that will enable camera-based occupancy sensing for smart homes, this project employs PyTorch. The flexibility of PyTorch's interface and the availability of libraries like torchvision make it easier to create and evaluate various

architectures of neural networks. Images from home cameras are processed by the CNN in order to find out whether or not there are occupants in the rooms and their types. By using convolutional layers followed by max-pooling layers, the images are processed in such a way that only important features are retained and others discarded. These features are then classified by fully connected layers to determine if individuals are present. The final output indicates the room's occupancy status, which demonstrates how effective CNNs can be in improving smart homes through accurate and fast detection of occupancy.

In this project, different CNN models implemented with PyTorch are employed for image classification in camera-based occupancy sensing for smart home applications. Below is a summary of each model and its advantages:

1. **ResNet50**
    - **Description**: ResNet50 is a deep CNN with 50 layers that utilizes residual connections to address the vanishing gradient problem in deep networks. These shortcut connections allow the model to maintain performance despite its depth.
    - **Benefits**: ResNet50 is effective at extracting intricate features from images, providing high accuracy for detecting and classifying occupancy-related details in various scenes. [10]

2. **MobileNet V3 Large**
    - **Description**: MobileNet V3 Large is optimized for mobile and embedded devices, using depth-wise separable convolutions and Squeeze-and-Excite modules to achieve a balance between performance and computational efficiency.
    - **Benefits**: This model is well-suited for real-time occupancy detection in smart homes due to its efficiency and low computational requirements, making it ideal for on-device processing. [11]

3. **MobileNet V3 Small**
    - **Description**: MobileNet V3 Small is a compact version designed for environments with very limited computational resources, retaining the core features of MobileNet V3 while minimizing power consumption.
    - **Benefits**: It offers fast and lightweight processing, making it suitable for deployment in smaller, resource-constrained smart home devices. [12]

4. **EfficientNet B0**
    - **Description**: EfficientNet B0 is the foundational model of the EfficientNet series, utilizing a compound scaling method to optimize network depth, width, and resolution, resulting in an efficient model with fewer parameters.

- **Benefits**: EfficientNet B0 strikes a strong balance between accuracy and computational efficiency, which is advantageous for smart home systems that require both high precision and manageable resource usage. [13]

5. **MobileNet V2**

   - **Description**: MobileNet V2 features an inverted residual structure and linear bottlenecks, enhancing efficiency and performance, especially on devices with limited computing power.
   - **Benefits**: It effectively reduces memory usage and computational cost, making it a robust choice for real-time occupancy sensing tasks in smart home applications. [14]

6.2 Simulation Environment Development

6.2 (a). Modelling:

To create a simulation environment to replicate the camera-based occupancy sensing process a diagram is shown below:

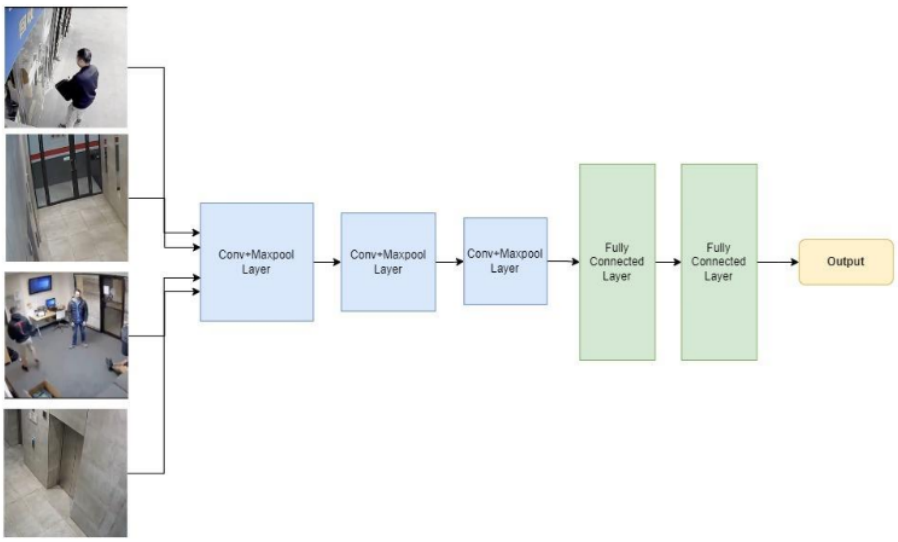

Figure 3: Simulation to replicate the camera-based occupancy sensing process

The architecture of a deep neural network for camera-based occupancy sensing in smart homes is featured in Figure 3. Relevant features are extracted from input images taken from different camera angles using convolutional and pooling layers before classifying them into occupied or non-occupied through fully connected layers. This method can accurately detect human beings' presence and automate various home functions like controlling lights, HVAC systems, and security systems. Some possible improvements are data augmentation, transfer learning, and including temporal information for better performance.

6.2 (b). Parameters:

The hyperparameters table for training is given below:

Table 4: Hyperparameters for training

Hyperparameter	Value
Number of Epochs	10
Batch Size	32
Learning Rate	0.001
Optimizer	Adam
Loss Function	BCEWithLogitsLoss()
Train Size	736 images
Test Size	93 images

In this project's CNN models, the hyperparameters used for training are shown in Table 4. The training procedures consist of 10 epochs, using a batch size of 32 to allow for efficient processing and updating of weights by the model. Moreover, a learning rate of 0.001 was selected to ensure stable learning while the Adam optimizer is applied due to its adaptability and efficiency towards different gradient scenarios. Additionally, the BCEWithLogitsLoss() loss function aids binary classification tasks by integrating the sigmoid layer with binary cross-entropy to manage the model's outputs. A total of 736 images were included in the training dataset and performance evaluation of the model was done on a separate test set comprising 93 images to ensure it generalizes well on new data.

6.2. (c). Scenarios:

To simulate different scenarios a diagram is given below:

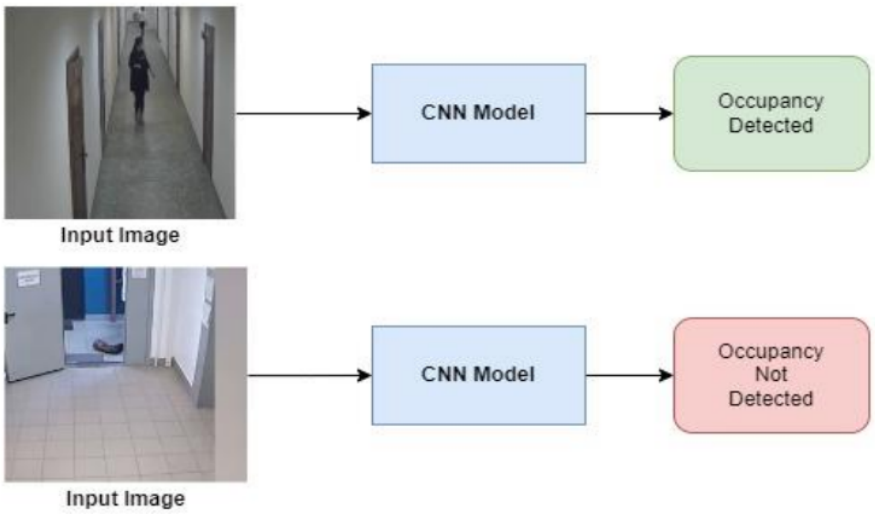

Figure 4: Different scenarios of the camera-based occupancy sensing process

Figure 4 outlines the process flow in the project titled "Camera-Based Occupancy Sensing Using AI for Smart Home Applications." This is done by a convolutional neural network (CNN) model which like the way humans perceive and recognize things from different angles regulates input images to find out whether there are any people. In the first scenario, one of many images depicting a corridor with some individual is analyzed and it is termed as having occupancy due to the "occupancy detected" tag written inside the green box. Meanwhile, in the second scenario, another image shows an empty hallway where there's no presence of anyone indicated by the "occupancy not detected" sign engraved on a red square. This visual representation delivers an accurate indication that occupancy is successfully detected thereby enabling automating home systems that include automatic lighting management and security measures suggested by Princeton University's Office of Public Safety.

6.3 Optimization Techniques Development

6.3. (a). Optimization Algorithms:

The optimization algorithms are very important when it comes to the enhancement of performance of Convolutional Neural Network (CNN) models that are used in camera-based occupancy sensing systems. Thus, these algorithms are meant to cause systematic adjustment of the parameters of these models to reduce errors, boost predictive accuracy, and improve the effectiveness of the entire organization. Some common optimization techniques that are mostly applied include Stochastic Gradient Descent (SGD), Adam, and RMSprop.

SGD allows for weight updates to be done iteratively on the model meaning that it will eventually approach an optimal solution although this process may take time. Adam is an adjustable learning rate optimization algorithm that merges properties of two other extensions (named RMSprop and Momentum) from SGD hence resulting into faster convergence with better performance especially in non-linear models with a lot of complexities. On its side RMSprop individually modifies learning rates for each parameter making it suitable in case there are differing data distributions at hand. [15]

These optimization algorithms are essential in smart home applications where the system has to operate in real-time and in different lighting and environmental conditions under which it operates. They not only help in reducing computation costs and training times for CNN models, but they also ensure that the models remain robust and responsive. This results in better detection of occupancy, which is important for the efficient operation of automated systems like lighting controls, HVACs, and monitoring security. The project employs these advanced optimization techniques to make sure the occupancy sensing system is reliable as well as scalable, and able to adjust to the changing demands of smart homes.

The project on Camera-Based Occupancy Sensing Using AI for Smart Home Applications benefits greatly from the Adam optimizer when it comes to determining how well CNN models can learn over time. The adaptive learning rate optimization algorithm is an adjustment made by Adam throughout training such that the process becomes faster and smoother enabling it to converge a lot more quickly. With this adaptability, there is no way that a good learning curve can be established without reducing error through time. Therefore, by using Adam, it becomes possible for the CNNs used in this project to easily separate between occupied and unoccupied premises even when presented with various images that are complex and heterogeneous. Consequently, this helps to reduce the loss function much faster and stabilizes the learning curve sooner resulting in a highly reliable and very accurate occupancy detection system that works instantly with smart homes.

Now the learning curves for the project are shown below:

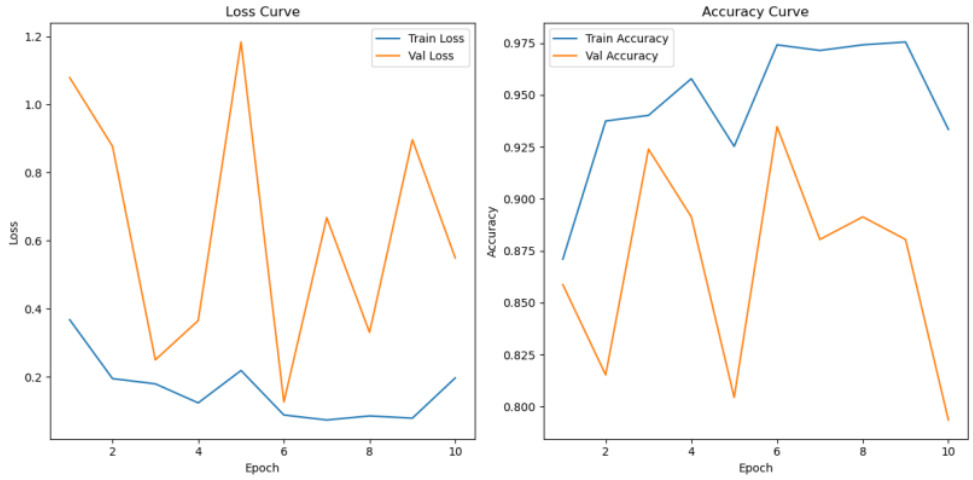

Figure 5: Learning Curve for ResNet50

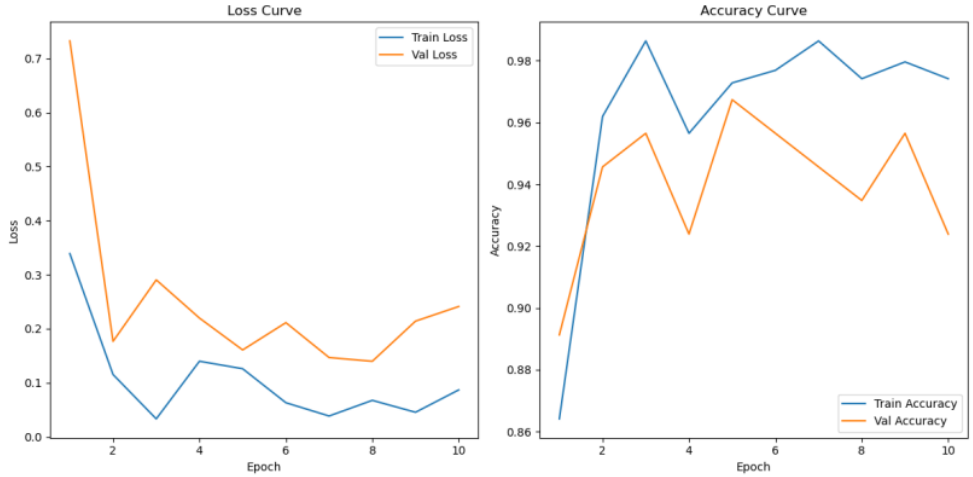

Figure 6: Learning Curve for EfficientNet B0

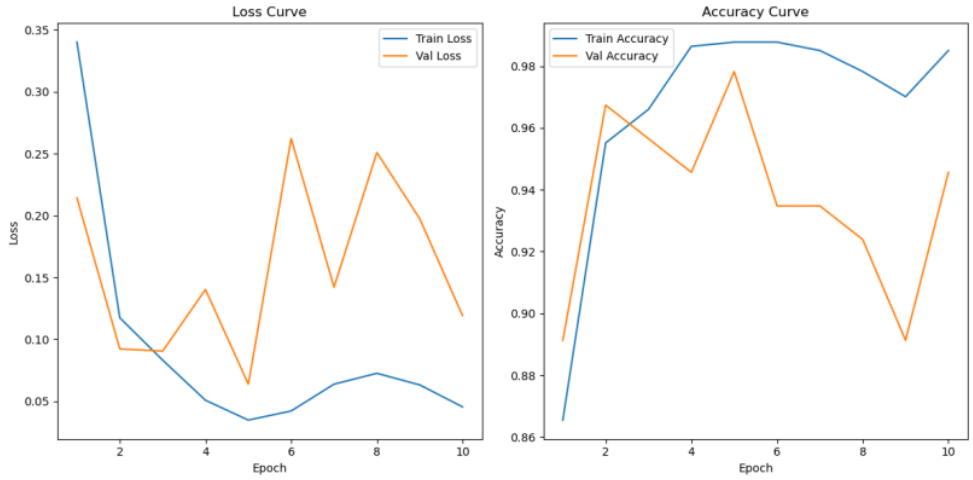

Figure 7: Learning Curve for MobileNet V2

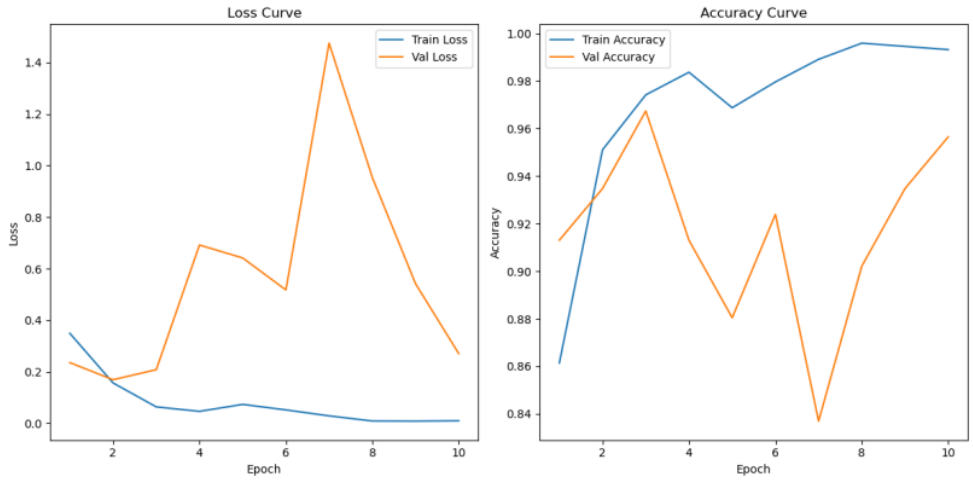

Figure 8: Learning Curve for MobileNet V3 Large

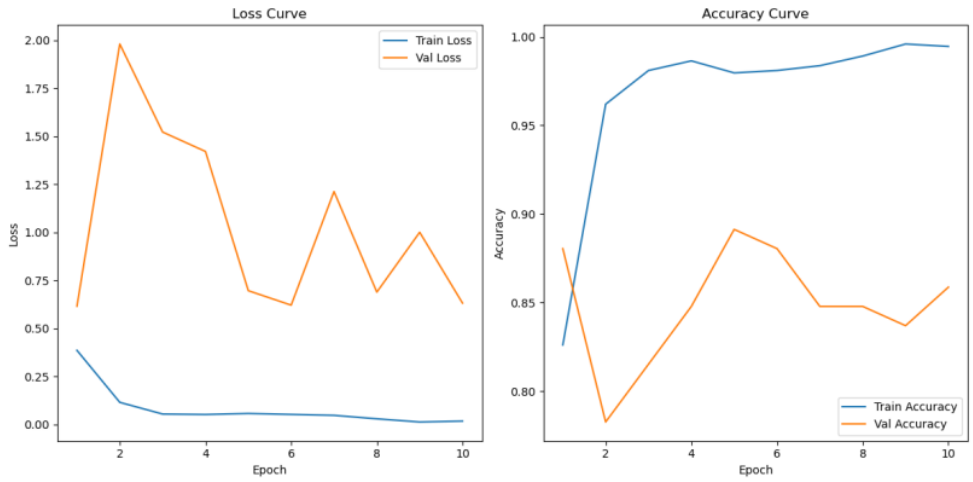

Figure 9: Learning Curve for MobileNet V3 Small

Through the project is about Camera-Based Occupancy Sensing Using AI for Smart Home Applications, an increase in the ability to accurately detect occupancy is achieved by minimizing loss during model training. A reduction in the loss indicates that the learning has improved, thus leading to an increased accuracy when it comes to distinguishing between spaces that are occupied from those that are not occupied. This increased accuracy confirms that CNN model can be trusted for use in everyday homes and its uses include but are not limited to automated lighting, security monitoring, and energy management.

6.3. (b). Performance Metrics:

Model Accuracy:

In the Camera-Based Occupancy Sensing Using AI for Smart Home Applications project, accuracy is a key metric for evaluating the performance of the CNN models. Accuracy measures the proportion of correctly identified occupancy states (whether a space is occupied or not) out of the total number of predictions made by the model. It is calculated using the formula:

Accuracy = (Number of Correct Predictions × 100) / (Total Number of Predictions)

This percentage reflects the model's effectiveness in distinguishing between occupied and unoccupied spaces. A higher accuracy indicates that the model is effectively learning to recognize different scenarios and conditions within a smart home environment, which is crucial for reliable automation of tasks such as lighting control, security monitoring, and energy management. By tracking accuracy during the training and validation phases, the performance of the CNN models can be assessed, and adjustments can be made to improve their precision and reliability in real-world applications.

Learning Curve:

For the project, Camera Occupancy Sensing Using AI for Smart Home Applications the learning curve has several phases: first, it is necessary to gather and pre-process data that consists of labeling images and creating bounding boxes; second, the selection of the CNN models is made by training them e.g. ResNet50, MobileNet V3 along with hyperparameter tuning and performance evaluation, the third stage is a refinement of the models through test results; thereafter comes the integration of the trained models into the smart home system enabling real-time occupancy detection; lastly; monitoring system performance iteratively using feedbacks for improved accuracy as well as reliability.

Here, for example, the learning curve for the Model ResNet50 is shown below with explanation:

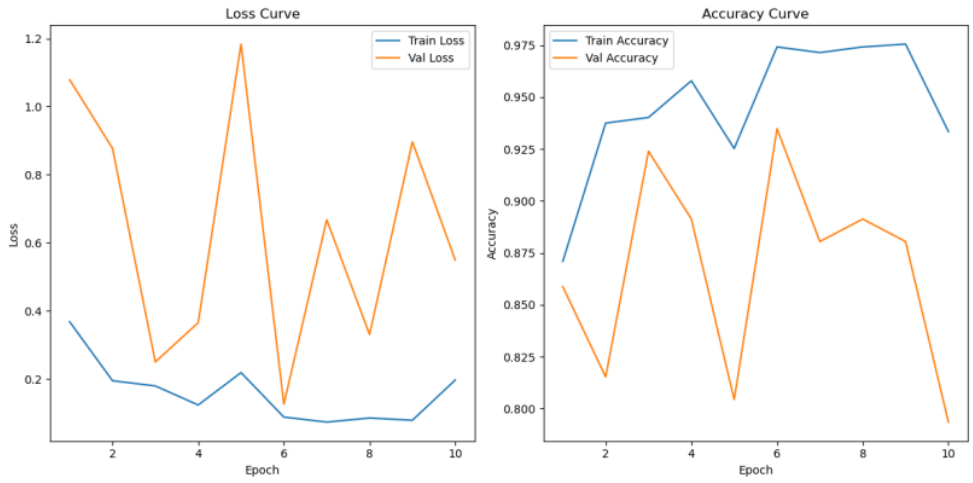

Figure 10: Learning Curve of ResNet50 model to explain the performance metrics

Here, the learning curves for the ResNet50 model show decreasing training loss and increasing accuracy, which are positive. However, if validation loss rises or validation accuracy plateaus while training metrics improve, it may indicate overfitting. To counter this, use regularization or data augmentation. Analyzing these curves helps you assess and optimize the model's performance effectively.

Confusion Matrix:

A confusion matrix is a way that helps to evaluate how classification models perform by providing a summary of prediction results. The Camera-Based Occupancy Sensing project, it shows how many instances were either accurately or inaccurately categorized across different categories like 'occupancy' and 'no occupation'. The matrix contains true positives (correctly classified occupancy), false positives (misspecified occupancy), true negatives (correctly classified non-occurrence), and false negatives (missed cases). By analyzing the confusion matrix; it provides avenues for identifying strong points as well as weaknesses of a given model thereby guiding in its upcoming revisions to [16] achieve better accuracy and lessen misclassifications while detecting occupancy.

Now some diagrams are given below to show the confusion matrix for the project:

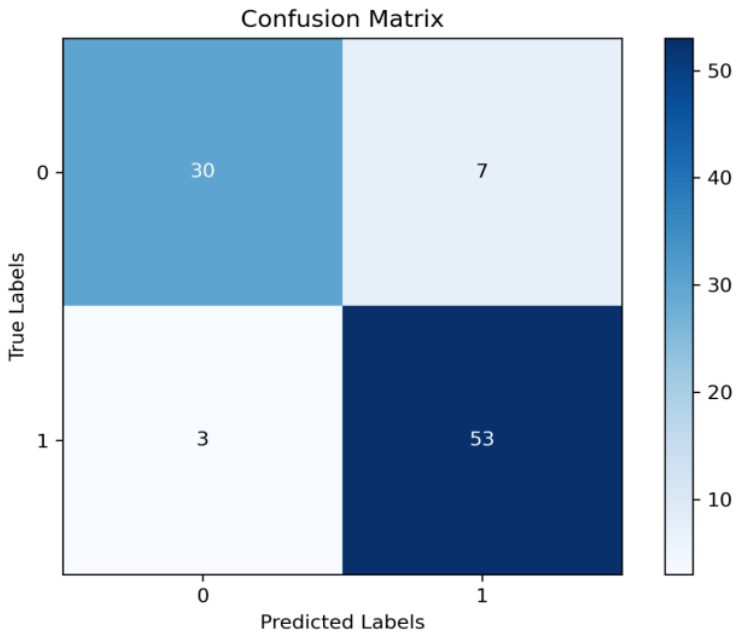

Figure 11: Confusion matrix graph for ResNet50 model

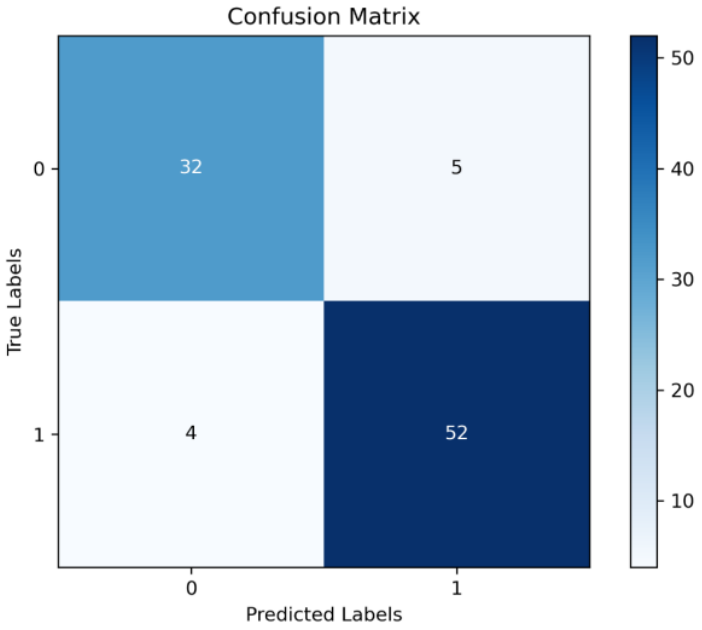

Figure 12: Confusion matrix graph for EfficientNet B0 model

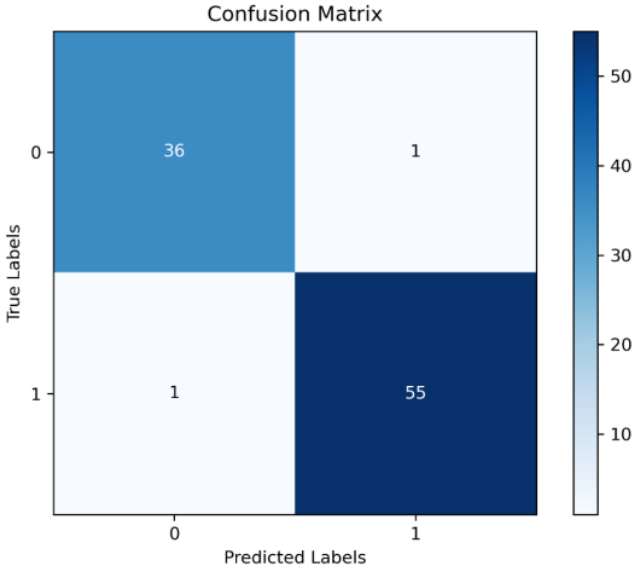

Figure 13: Confusion matrix graph for MobileNet V2model

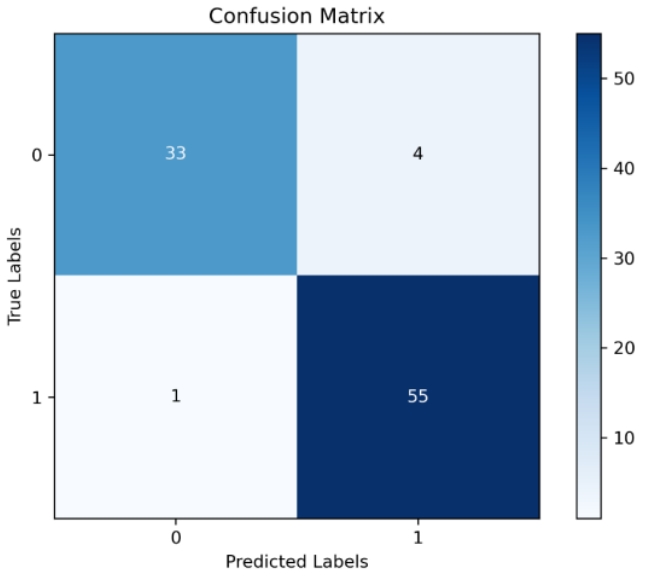

Figure 14: Confusion matrix graph for MobileNet V3 large model

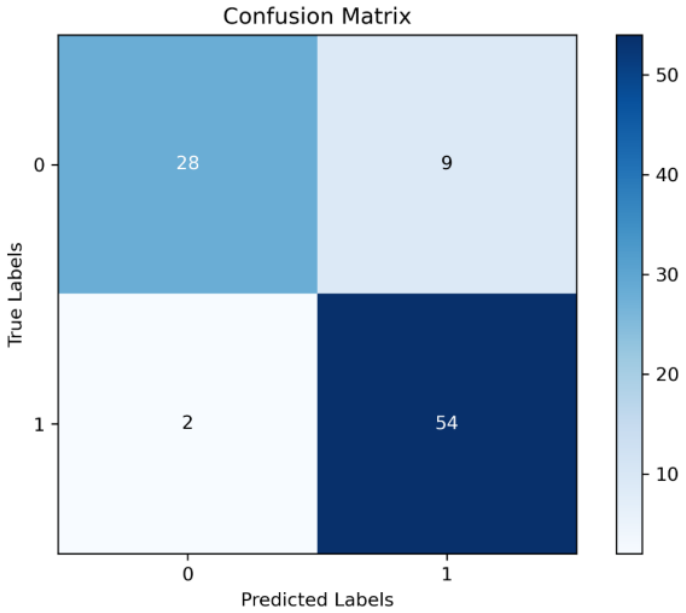

Figure 15: Confusion matrix graph for MobileNet V3 small model

Chapter 7

Testing:

To test the project in a controlled environment with real-world samples a table of testing and a screenshot of model training is given below:

Table 5: Testing results

Model Name	Test Loss	Test Accuracy
ResNet50	0.2154	89.25%
MobileNet V3 Large	0.3350	94.62%
EfficientNet B0	0.1918	90.32%
MobileNet V2	0.1205	97.85%
MobileNet V3 Small	0.8295	88.17%

Screen Shot:

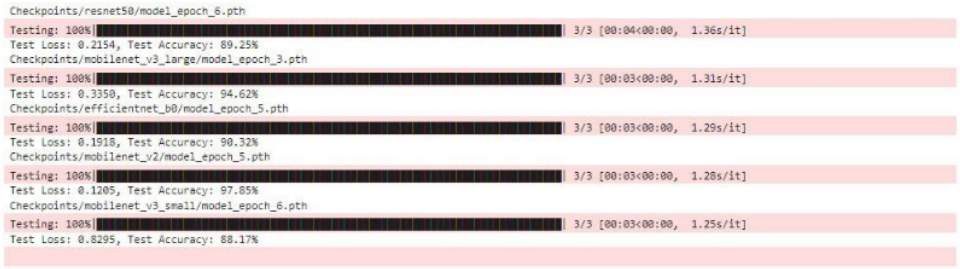

Figure 16: Testing Results from the training of the models

Table 5 provides testing results for various models used in the Camera-Based Occupancy Sensing project, showing their test loss and accuracy:

- **ResNet50** achieved a test loss of 0.2154 and a test accuracy of 89.25%.
- **MobileNet V3 Large** had a test loss of 0.3350 and a test accuracy of 94.62%.
- **EfficientNet B0** recorded a test loss of 0.1918 and a test accuracy of 90.32%.
- **MobileNet V2** demonstrated the highest performance with a test loss of 0.1205 and a test accuracy of 97.85%.
- **MobileNet V3 Small** had a test loss of 0.8295 and a test accuracy of 88.17%.

These results highlight MobileNet V2 as the most accurate model with the lowest loss, indicating it performs best for occupancy sensing. In contrast, MobileNet V3 Small shows higher loss and lower accuracy, suggesting it may be less suitable for this task.

Chapter 8

Analysis:

8.1 Validation Result Analysis:

To validate the model's accuracy and reliability, a result table and a diagram of the model's result analysis are given below:

Table 6: A result table after model training

Model Name	Train loss	Train accuracy	Validation loss	Validation accuracy	Test loss	Test accuracy
ResNet50	0.0875	97.42%	0.1258	93.48%	0.2154	89.25%
MobileNet V3 Large	0.0640	97.42%	0.2084	96.74%	0.3350	94.62%
EfficientNet B0	0.1259	97.28%	0.1607	96.74%	0.1918	90.32%
MobileNet V2	0.0349	98.78%	0.0639	97.83%	0.1205	97.85%
MobileNet V3 Small	0.0520	98.10%	0.6209	88.04%	0.8295	88.17%

Table 6 presents performance metrics for different models after training, showing train loss, train accuracy, validation loss, validation accuracy, test loss, and test accuracy:

- **ResNet50**: Achieved a train loss of 0.0875 and train accuracy of 97.42%, with validation loss of 0.1258 and validation accuracy of 93.48%, and test loss of 0.2154 with test accuracy of 89.25%. This indicates strong training performance but some decrease in accuracy and an increase in loss on validation and test sets.
- **MobileNet V3 Large**: Had a train loss of 0.0640 and train accuracy of 97.42%, with a validation loss of 0.2084 and validation accuracy of 96.74%, and a test loss of 0.3350 with test accuracy of 94.62%. It shows good performance on both training and validation sets but a higher test loss.
- **EfficientNet B0**: Recorded a train loss of 0.1259 and train accuracy of 97.28%, with validation loss of 0.1607 and validation accuracy of 96.74%, and test loss of 0.1918 with test accuracy of 90.32%. It performs consistently across training, validation, and testing phases.
- **MobileNet V2**: Demonstrated the best results with a train loss of 0.0349 and train accuracy of 98.78%, validation loss of 0.0639 and validation accuracy of 97.83%, and a test loss of 0.1205 with test accuracy of 97.85%. It shows superior performance overall with the lowest loss and highest accuracy.

- **MobileNet V3 Small**: Achieved a train loss of 0.0520 and train accuracy of 98.10%, but with a significantly higher validation loss of 0.6209 and validation accuracy of 88.04%, and a test loss of 0.8295 with test accuracy of 88.17%. This indicates potential overfitting or issues with generalization to new data.

Overall, MobileNet V2 stands out with the highest accuracy and lowest loss across all metrics, while MobileNet V3 Small shows poorer performance on validation and test data despite strong training results.

Now from the table, a graph is plotted below to show the analysis more efficiently:

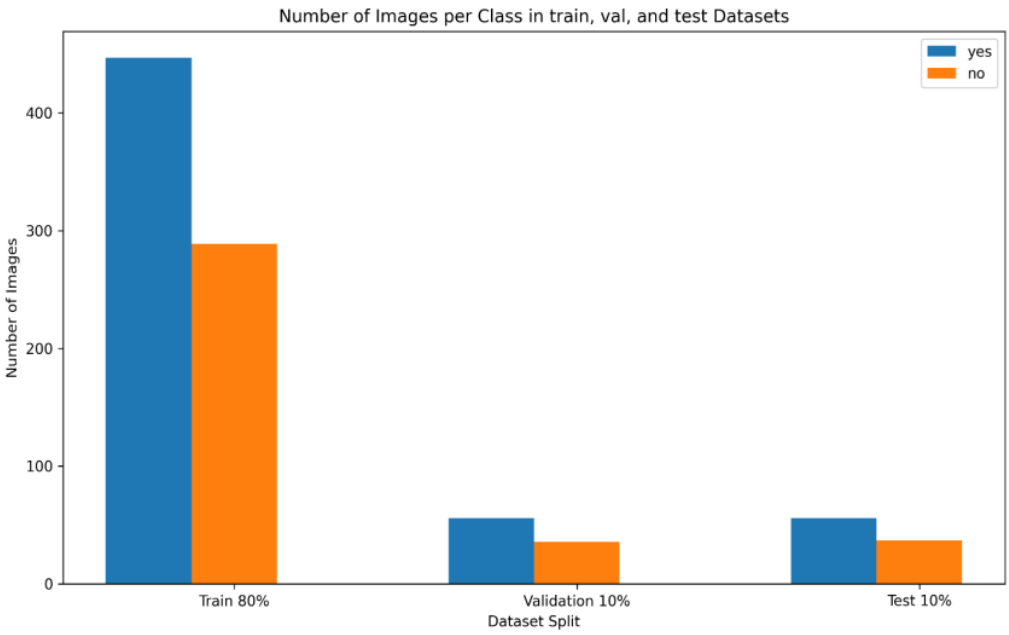

Figure 17: A bar chart for result analysis of the trained datasets

In Figure 17, the training, validation, and test datasets can be classified into either “yes” or “no” categories as illustrated in the diagram. Out of the total data, 80% is for training purpose with numerous more images belonging to the “yes” compared with “no”. Validation set and test set each hold 10% of the data but have an identical class distribution disproportionality like that observed in training set. This discrepancy should be corrected during the modeling process to avoid any kind of favoritism during model building thereby leading to equal performance in both classes.

8.2 Benchmarking:

To compare the simulation results with real-world data a table is given below:

Table 7: A table for showing the benchmarking results to compare the simulation results with real-world data

	Model Name	Test Loss	Test Accuracy
Benchmarking Dataset	**ResNet50**	0.0669	97.18%
	MobileNet V3 Large	0.1144	97.72%
	EfficientNet B0	0.0603	97.94%
	MobileNet V2	0.0251	99.35%
	MobileNet V3 Small	0.3331	93.70%

Screenshot of results from which Table 7 was derived:

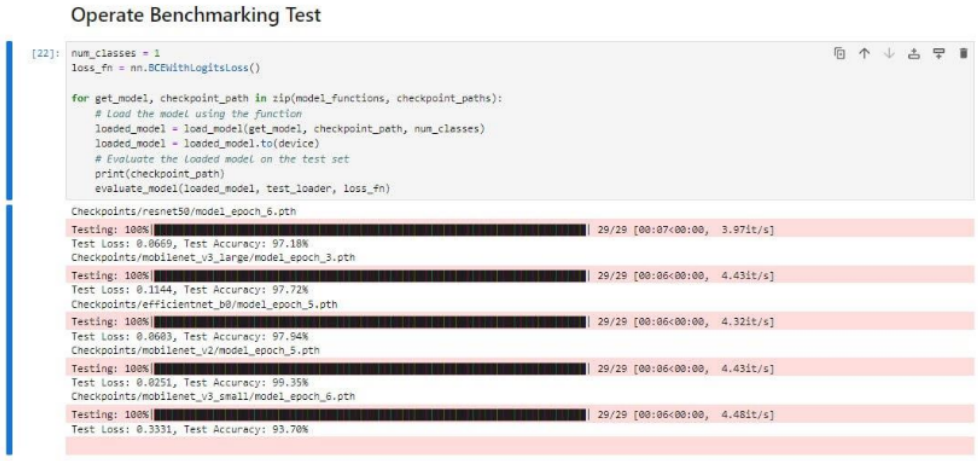

Figure 18: Results of Benchmarking Test

The table of benchmarking compares performance tests losses and accuracies of several CNN models using a certain dataset. With test loss equaling to 0.0669 while on the other hand obtaining; a test accuracy rate standing at 97.18% ResNet50 emerges with high accuracy but a slightly high loss. Message these are just numbers that can sound meaningless… The difference between them might not mean anything on their own because they do not tell you how good or bad the system worked overall However in comparison MobileNet V3 Large had similar balance with 0.1144. 49 along with that it

had accuracy of 97.72%. That was also done well by EfficientNet B0 as it got lowest loss 0.0603 and highest quality 97.94%, which means it is well-balanced model out of all five compared here so far! At#5 in performance evaluation is MobileNet V2 having great results; test accuracy equal to 99.35% and a low loss of 0.0251 indicating extraordinary results for this case study conducted under certain circumstances! On the contrary MobileNet V3 Small incurred loss rate of 0.3331 about 7 times more than last one and test accuracy rate stood at 93.70% showing difficulties which may appear Here are just some highlights from our summary: so far there are two most promising solutions for practical applications related to automatic detection of residents' presence in smart houses delivered by MobileNet V2 and EfficientNet B0 models respectively.

8.3 Analysis of Camera-Based Occupancy Sensing Using AI for Smart Home Applications efficiency in industry:

The efficiency of Camera-Based Occupancy Sensing Using AI for Smart Home Applications is significant in enhancing both operational performance and user experience in the industry. This advanced system leverages AI algorithms to analyze camera feeds, enabling precise detection of room occupancy and activity levels. Such capabilities offer several key advantages:

1. **Optimized Energy Management**: By accurately determining occupancy, the system can automate lighting, heating, and cooling adjustments based on room usage. This leads to substantial energy savings by reducing unnecessary power consumption in unoccupied spaces. The result is a reduction in utility costs and a contribution to environmental sustainability.
2. **Enhanced Security**: AI-driven occupancy sensing improves security by monitoring unusual behavior or movements within the home. For instance, it can alert homeowners to potential intrusions or irregular activities, providing an additional layer of protection and ensuring a safer living environment.
3. **Improved User Experience**: The system enhances user convenience by integrating seamlessly with home automation systems. It enables features such as automatic lighting adjustments and climate control, which adapt to the presence of occupants. This leads to a more intuitive and comfortable living experience.
4. **Data-Driven Insights**: The system can also provide valuable data on occupancy patterns and usage trends. This information can be used to further refine energy management strategies and tailor home automation features to better meet the needs of residents.

Overall, the Camera-Based Occupancy Sensing System demonstrates significant industry efficiency by delivering practical benefits, including cost savings, increased security, and improved user satisfaction. Its ability to integrate with smart home technologies and adapt to real-time conditions underscores its value in modernizing residential environments. [17]

8.4 Reduction in Contamination: evaluate how improved sorting reduces contamination in recycling streams for the industry:

In the recycling industry, effective sorting is crucial for reducing contamination in recycling streams, which enhances the quality and value of recyclable materials. Improved sorting technologies, particularly those integrated with AI and advanced sensors, offer several benefits in this context:

1. **Enhanced Material Separation**: Advanced sorting systems use sophisticated algorithms and high-resolution imaging to precisely identify and separate different types of materials. This granularity in sorting reduces the likelihood of cross-contamination between recyclables, ensuring that materials such as plastics, metals, and paper are kept in their respective streams.
2. **Increased Purity of Recycled Materials**: By reducing contamination, improved sorting leads to higher purity levels in recycled materials. Purity is critical for producing high-quality recycled products that meet industry standards and are more suitable for reuse in manufacturing. This, in turn, increases the market value of recycled materials and makes them more attractive to buyers.
3. **Reduction in Waste Processing Costs**: Contaminated recycling streams often require additional processing steps to remove impurities, which increases operational costs. By minimizing contamination through improved sorting, these additional processing steps can be reduced, leading to cost savings for recycling facilities. Efficient sorting systems also streamline the recycling process, making it more cost-effective overall.
4. **Improved Environmental Impact**: Effective sorting reduces the volume of recyclable materials that end up in landfills due to contamination. Ensuring that a higher percentage of materials are correctly sorted and recycled reduces the environmental impact of waste disposal. This supports sustainability goals and reduces the carbon footprint associated with waste management.
5. **Compliance with Regulations**: Stringent environmental regulations often require high standards for recycling practices. Improved sorting technologies help recycling facilities comply with these regulations by maintaining the quality and integrity of recycling streams. Compliance not only avoids potential fines but also enhances the facility's reputation and credibility in the industry.

In summary, improved sorting technologies play a pivotal role in reducing contamination in recycling streams. They enhance material separation, increase the purity of recycled products, lower waste processing costs, and support environmental sustainability. By addressing contamination effectively, these technologies contribute to a more efficient and environmentally responsible recycling industry. [18]

8.5 Consumer Benefits: analyze the overall benefits of Camera-Based Occupancy Sensing Using AI for Smart Home Applications:

The integration of Camera-Based Occupancy Sensing Using AI in smart home applications provides a range of significant benefits for consumers, enhancing both comfort and efficiency in residential environments. The following points highlight the key advantages:

1. **Energy Efficiency and Cost Savings**: AI-driven camera systems enable precise detection of room occupancy, allowing for automated control of lighting, heating, and cooling systems based on actual usage. This targeted approach minimizes energy waste by ensuring that these systems are only active when needed. As a result, homeowners can experience substantial savings on their utility bills.
2. **Enhanced Security and Safety**: The AI-powered cameras continuously monitor for unusual activity or potential security breaches, providing an added layer of protection for the home. Features such as motion detection and activity alerts can notify homeowners of potential intrusions or emergencies, thereby enhancing overall home security and ensuring a safer living environment.
3. **Increased Convenience and Comfort**: By automating responses to occupancy changes, the system offers increased convenience and comfort. For example, lighting and climate control can adjust automatically as residents move through different areas of the home. This creates a more intuitive and personalized living experience, where the home adapts to the residents' needs without requiring manual adjustments.
4. **Optimized Home Management**: The system provides valuable insights into occupancy patterns, allowing homeowners to manage their living environment more effectively. For instance, it can help identify which rooms are used most frequently, enabling better planning for energy usage and home maintenance. This data-driven approach enhances the ability to optimize home resources and improve overall efficiency.
5. **Seamless Integration with Smart Home Ecosystems**: Camera-based occupancy sensing integrates smoothly with existing smart home devices and systems, such as voice assistants and home automation hubs. This interoperability allows for a cohesive smart home experience, where various devices and systems work together harmoniously to create a connected and responsive living environment.
6. **Personalized User Experience**: AI-driven systems can learn and adapt to individual preferences and routines over time. This personalization ensures that the home environment is continuously optimized to suit the specific needs and habits of its residents, enhancing overall satisfaction and comfort.

In summary, Camera-Based Occupancy Sensing Using AI offers substantial consumer benefits by improving energy efficiency, enhancing security, increasing convenience, and providing valuable insights for optimized home management. Its seamless integration with smart home ecosystems and ability to deliver a personalized user experience further contribute to its appeal and effectiveness in modern residential settings. [19]

Chapter 9

9.1 Monitoring Procedures for the Model:

Effective monitoring procedures are crucial for maintaining the performance and reliability of the AI models used in Camera-Based Occupancy Sensing for smart home applications. These procedures ensure that the models operate optimally over time and adapt to any changes in the environment or data. The following points outline a comprehensive approach to monitoring these models:

1. **Performance Tracking**: Continuously track key performance metrics such as accuracy, precision, recall, and F1 score. This involves regularly evaluating the model's predictions against a validation dataset to ensure it consistently performs well. Performance tracking helps identify any deviations from expected outcomes and provides insights into potential issues or areas for improvement.
2. **Real-Time Monitoring**: Implement real-time monitoring to assess how the model performs under live conditions. This includes checking for any discrepancies between predicted and actual occupancy data and monitoring the model's response to dynamic changes in the environment. Real-time monitoring ensures that the model remains effective in detecting occupancy and responding to varying conditions.
3. **Data Drift Detection**: Monitor for data drift, which occurs when the characteristics of input data change over time. This can affect the model's accuracy and reliability. Techniques such as statistical tests and visualization tools can be used to detect shifts in data distribution, allowing for timely adjustments or retraining of the model.
4. **Model Retraining and Updates**: Establish a routine for retraining the model with new data to maintain its accuracy and relevance. This involves periodically updating the training dataset with recent samples and re-evaluating the model's performance to ensure it continues to meet the desired standards. Retraining helps the model adapt to changes in user behavior and environmental conditions.
5. **Error Analysis**: Conduct regular error analysis to understand the types and causes of errors made by the model. By examining false positives, false negatives, and other misclassifications, you can identify patterns or specific scenarios where the model may be underperforming. This analysis informs targeted adjustments to improve model performance.
6. **User Feedback Integration**: Collect and incorporate feedback from users to gain insights into the model's real-world performance and user experience. Feedback can highlight practical issues, such as missed detections or false alarms, and guide improvements to enhance the model's effectiveness and user satisfaction.
7. **System Health Checks**: Perform regular system health checks to ensure that all components of the model deployment, including hardware and software, are functioning correctly. This includes verifying data flow, computational resources, and system stability to prevent and address any technical issues that could impact model performance.

In summary, robust monitoring procedures are essential for ensuring the ongoing effectiveness of AI models in Camera-Based Occupancy Sensing systems. By implementing performance tracking, real-time monitoring, data drift detection, model retraining, error analysis, user feedback integration, and system health checks, you can maintain high model performance, adapt to changes, and provide reliable occupancy sensing in smart home applications.

9.2 Risk Assessment of the Model:

Here is a risk assessment table tailored to the Camera-Based Occupancy Sensing Using AI for Smart Home Applications project: [20]

Table 8: A table showing the risk assessment of the project

Risk No.	Description of the Risk	Probability of the Risk	Effect on the Project	Contingencies to Mitigate the Risk
GR1	Technical Risks	MEDIUM TO HIGH	Technical issues may delay model development, impact real-time performance, or reduce system accuracy.	Conduct regular technical reviews, perform rigorous testing, and have a backup plan for technical issues.
GR2	Data Quality Risks	MEDIUM	Poor data quality could lead to inaccurate occupancy detection and reduced system performance.	Implement thorough data preprocessing and validation techniques to ensure high-quality input data.
GR3	Resource Risks	MEDIUM	Insufficient computational resources or expertise may delay model training and deployment.	Ensure access to adequate computing resources and skilled personnel, and allocate sufficient project time.
GR4	Schedule Risks	MEDIUM	Delays in development phases could lead to missed deadlines and additional costs.	Use project management tools to monitor progress, adjust timelines as necessary, and reallocate resources if needed.
SR1	Model Performance Risk	HIGH	Poor model performance could result in inaccurate occupancy sensing,	Continuously monitor model performance and conduct retraining

			affecting overall system reliability.	with updated data to improve accuracy.
SR2	Algorithm Complexity Risk	MEDIUM TO HIGH	High complexity in algorithms may lead to difficulties in model interpretation and maintenance.	Simplify algorithms where possible, modularize complex components, and maintain thorough documentation.
SR3	Data Drift Risk	MEDIUM	Changes in occupancy patterns or environments may reduce model accuracy over time.	Implement data drift detection mechanisms and periodically retrain the model with current data.
SR4	Testing and Validation Risk	MEDIUM	Inadequate testing might result in undetected errors, leading to system failures in real-world scenarios.	Develop a comprehensive testing plan, including edge cases and real-world scenarios, and conduct thorough validation.
SR5	Integration Risk	MEDIUM	Challenges integrating the model with existing smart home systems may affect overall system functionality.	Plan integration strategies early, conduct incremental testing, and ensure compatibility with existing systems.
SR6	User Adoption Risk	LOW TO MEDIUM	Users may be reluctant to adopt the new system due to unfamiliarity or perceived complexity.	Provide comprehensive user training, and support, and involve users early in the development process to address their needs.

Table 8 outlines the potential risks, their likelihood, impact on the project, and the strategies to mitigate them, ensuring a successful and effective deployment of the Camera-Based Occupancy Sensing system in smart home applications.

Chapter 10

10.1 The work carried out to date:

How We Designed Our Project:

A top-of-the-line Camera-Based Occupancy Sensing system has been developed using advanced Convolutional Neural Networks (CNNs), such as ResNet50, MobileNet V3, EfficientNet B0, MobileNet V2 and MobileNet V3 Small for effective monitoring and analysis of occupancy in smart home applications.

• A comprehensive dataset consisting of frames taken from hospital video footage and several other sources was used to train CNN models covering different scenarios with or without humans in order to achieve a robust model performance.

• For our project hardware configuration we had an Intel 13th Gen Core i5 processor alongside an NVIDIA GeForce RTX 4060 Ti Graphics Card which were sufficient for demanding image processing computations.

• Adam Optimizer was employed to adjust the model parameters where necessary so as to improve accuracy and performance in terms of our occupancy sensing models.

• To replicate real-world situations we constructed a complex simulation environment that mimicked various home settings thus allowing us to test it through diverse scenarios.

Scenarios: Problems Encountered and How We Solved Them:

Scenario 1 – Inconsistent Occupancy Detection across Different Lighting Conditions

Problem: No matter the various lighting conditions, the system's accuracy within occupancy detection was greatly affected.

Solution: Advanced pre-processing techniques were applied to adjust the light conditions in input images. Furthermore, we trained our models on a large dataset that captured various light levels to enhance their consistency and make them more robust.

Scenario 2 – Computational Constraints Affecting Real-Time Performance

Problem: Computational bottlenecks prevented real-time processing of video streams hence causing delays in occupancy monitoring.

Solution: Our code was optimized for parallel processing and we made use of GPU acceleration to increase the processing speed. By simplifying the structure of our neural networks and optimizing resource allocation, we managed to find a compromise between speed and accuracy.

Scenario 3: Variability of Model Performance in Occupational Scenarios

Problem: The models had inconsistent efficiency when classifying occupancy under different conditions like room layout differences or number of occupants variations.

Solution: We increased the training dataset size to include a wider range of scenarios and room types, as well as using data augmentation techniques in order to enhance the model's ability to generalize. Thus, the system remained accurate under various conditions through regular performance evaluations and adjustments.

Future Improvement Suggestions
Advanced Sensors Integration:

Integration of extra sensors such as thermal imaging and depth sensors will improve the accuracy of occupancy detection. Thermal sensors can help differentiate between different types of activities and enhance their ability to tell from each other multiple occupants occupying various places with differing circumstances.

Scalability Enhancements:
To handle larger datasets and more complex occupancy scenarios, a distributed version of the system should be developed. This might involve deploying the system across different processing units or integrating it into extensive smart home networks to manage big environments, similar to implementations done in commercial or multi-residential buildings.

AI-Driven Continuous Learning:
A continuous learning framework that enables the system to learn from new data and changing occupancy patterns should be implemented. This may comprise automatic model updates utilizing fresh labeled data, hence keeping pace with user behavior changes about housing designs and layouts.

By focusing on these aspects of improvement, we could even enhance further effectiveness and adaptability of our Camera-Based Occupancy Sensing system thereby ensuring that it remains at the cutting edge of smart home technology.

10.2 Conclusion:

Our Camera-Based Occupancy Sensing system represents a significant milestone in merging artificial intelligence with smart home technology. By leveraging advanced Convolutional Neural Networks (CNNs) and a diverse array of data sources, we have developed a sophisticated system capable of accurately detecting occupancy in real-time. Through innovation and optimization, we have successfully addressed challenges such as performance inconsistencies and computational limitations.

The successful implementation and testing of our system underscore its potential to greatly enhance smart home applications, offering users improved automation and monitoring capabilities. Looking ahead, the system's evolution will involve integrating advanced sensors, enhancing scalability, and incorporating continuous learning mechanisms. These advancements will ensure the system stays aligned with the latest technological developments and meets the evolving demands of modern living environments.

Chapter 9

Keep Blank

References

[1] "Image Classification using CNN," geeksforgeeks, 21 May 2024. [Online]. Available: https://www.geeksforgeeks.org/image-classifier-using-cnn/.

[2] "Everything you Need to Know About Hardware Requirements for Machine Learning," einfochips, 24 April 2024. [Online].

[3] "Software Requirement Classification Using Machine Learning Algorithms," ieeexplore, 21 April 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10169588.

[4] "What is block diagram in software engineering?," geeksforgeeks, 26 July 2024. [Online]. Available: https://www.geeksforgeeks.org/what-is-block-diagram-in-software-engineering/.

[5] K. Verner, "Human Detection Dataset," Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/constantinwerner/human-detection-dataset/data.

[6] T. Data, "Medical Staff People Tracking," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/trainingdatapro/medical-staff-people-tracking/data.

[7] "DataForce accelerates your data labeling processes with our range of human-annotation services at scale," dataforce.ai, [Online]. Available: https://www.dataforce.ai/services/data-annotation.

[8] "Data Preprocessing in Data Mining," geeksforgeeks, 06 May 2023. [Online]. Available: https://www.geeksforgeeks.org/data-preprocessing-in-data-mining/.

[9] Sarka, "Image classification using CNN," medium, 25 February 2024. [Online]. Available: https://medium.com/@sarka.pribylova/image-classification-using-cnn-0fad8367acfd.

[10] N. Kundu, "Exploring ResNet50: An In-Depth Look at the Model Architecture and Code Implementation," Medium, 23 January 2023. [Online].

[11] "mobilenet_v3_large," pytorch, [Online]. Available: https://pytorch.org/vision/main/models/generated/torchvision.models.mobilenet_v3_large.html.

[12] "mobilenet_v3_small," pytorch, [Online]. Available: https://pytorch.org/vision/main/models/generated/torchvision.models.mobilenet_v3_small.html.

[13] "EfficientNet," paperswithcode, [Online]. Available: https://paperswithcode.com/method/efficientnet.

[14] "What is MobileNetV2? Features, Architecture, Application and More," analyticsvidhya, [Online]. Available: https://www.analyticsvidhya.com/blog/2023/12/what-is-mobilenetv2/.

[15] "What is Adam Optimizer?," geeksforgeeks, 20 March 2024. [Online]. Available: https://www.geeksforgeeks.org/adam-optimizer/.

[16] "Understanding the Confusion Matrix in Machine Learning," geeksforgeeks, 08 July 2024. [Online]. Available: https://www.geeksforgeeks.org/confusion-matrix-machine-learning/.

[17] J. P. R. R. A. U. A. R. P. B. and S. V. Deeraj Krishna Kaki, "Development and implementation of novel sensor fusion algorithm for occupancy detection and automation in energy efficient buildings," *sciencedirect,* vol. 44, pp. 85-98, 2019.

[18] I. Z. T. Moreau and . J. Z. , "Toward zero waste events: Reducing contamination in waste streams with volunteer assistance," *sciencedirect,* vol. 76, pp. 39-45, 2018.

[19] "How AI-enabled Smart Home Cameras Can Help Enhance Overall Home Security," prolink2u, 28 May 2024. [Online].

[20] "Validation and Evaluation of Predictive Models in Hazard Assessment and Risk Management," link.springer, [Online]. Available: https://link.springer.com/article/10.1007/s11069-005-5182-6.

Report-Project Topic 3 Camera Based Occupancy Sensing

ORIGINALITY REPORT

9%	5%	4%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Syed Nisar Hussain Bukhari. "Data-Driven Farming - Harnessing the Power of AI and Machine Learning in Agriculture", CRC Press, 2024 Publication	1%
2	Submitted to Imperial College of Science, Technology and Medicine Student Paper	1%
3	Submitted to University of Glamorgan Student Paper	1%
4	fastercapital.com Internet Source	1%
5	Submitted to Manchester Metropolitan University Student Paper	<1%
6	www.fastercapital.com Internet Source	<1%
7	dspace.unive.it Internet Source	<1%

8	"Computer Vision and Image Processing", Springer Science and Business Media LLC, 2024 Publication	<1 %
9	assets.researchsquare.com Internet Source	<1 %
10	medium.com Internet Source	<1 %
11	ebin.pub Internet Source	<1 %
12	www.mdpi.com Internet Source	<1 %
13	Submitted to West University Of Timisoara Student Paper	<1 %
14	Submitted to SASTRA University Student Paper	<1 %
15	Submitted to Cranfield University Student Paper	<1 %
16	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023 Publication	<1 %
17	blog.enterprisedna.co Internet Source	<1 %
18	Submitted to Ganpat University Student Paper	<1 %

19	digitalcommons.unomaha.edu Internet Source	<1 %
20	www.ijraset.com Internet Source	<1 %
21	Submitted to Kingston University Student Paper	<1 %
22	Submitted to Mont Rose College Student Paper	<1 %
23	escholarship.org Internet Source	<1 %
24	"Proceedings of International Conference on Sustainable Expert Systems", Springer Science and Business Media LLC, 2021 Publication	<1 %
25	journal.hcsr.gov.sy Internet Source	<1 %
26	lambdagoogle.com Internet Source	<1 %
27	www.amazon.pl Internet Source	<1 %
28	www.findglocal.com Internet Source	<1 %
29	Alex Hultin, Manh-Toan Ho. "Sustaining Interdisciplinary Research: A Multilayer Perspective", Open Science Framework, 2019	<1 %

	Publication	
30	Amit Kumar Tyagi, Ajith Abraham. "Recurrent Neural Networks", CRC Press, 2022 Publication	<1 %
31	Submitted to The University of the West of Scotland Student Paper	<1 %
32	arxiv.org Internet Source	<1 %
33	dspace1.univ-tlemcen.dz Internet Source	<1 %
34	riunet.upv.es Internet Source	<1 %
35	www.amazon.com.au Internet Source	<1 %
36	www.ncbi.nlm.nih.gov Internet Source	<1 %
37	Evren Tuna, Asude Baykal, Alkan Soysal. "Multivariate and multistep mobile traffic prediction with SLA constraints: A comparative study", Ad Hoc Networks, 2024 Publication	<1 %
38	Yu-Ju Lin, H.A. Latchman, Minkyu Lee, S. Katar. "A power line communication network infrastructure for the smart home", IEEE Wireless Communications, 2002	<1 %

Publication

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		