**Phase 1**

We have Collected total 94 images.

In these 94 images we have covered

<>Old CS building Front and back side,

<>New CS building Front and back side,

<>EE building Front and back side,

<>Civil building Front and back side,

<>Main Building,

<>Business Department Building,

<>Library building Front and Corner,

By using python and pandas libraries we have made annotations.csv file which have following two columns:

<>image\_path - It contains path all images that we have captured

<>Building name - It contains name of corresponding building that image is presenting

We have faced following challenges while collecting our datasets of images:

<>First Lightning issue as due to lightning some images have bad resolution not much clear.

<>Some images have some noise

<>There are people in some images

**Phase 2**

**1. Introduction**  
This report summarizes a landmark recognition model that uses transfer learning with MobileNetV2 for building classification. The objective is to identify buildings from images by training a classifier on features extracted from a pre-trained CNN.

**2. Dataset**

* **Source:** Custom dataset containing images of different buildings.
* **Structure:**
  + data/images/: Contains images for each building (e.g., Library, MainBuilding, etc.)
  + data/annotations/annotations.csv: Contains image file paths and corresponding building labels.
* **Preprocessing:** Images are resized to 224×224 and normalized using MobileNetV2’s preprocessing function.

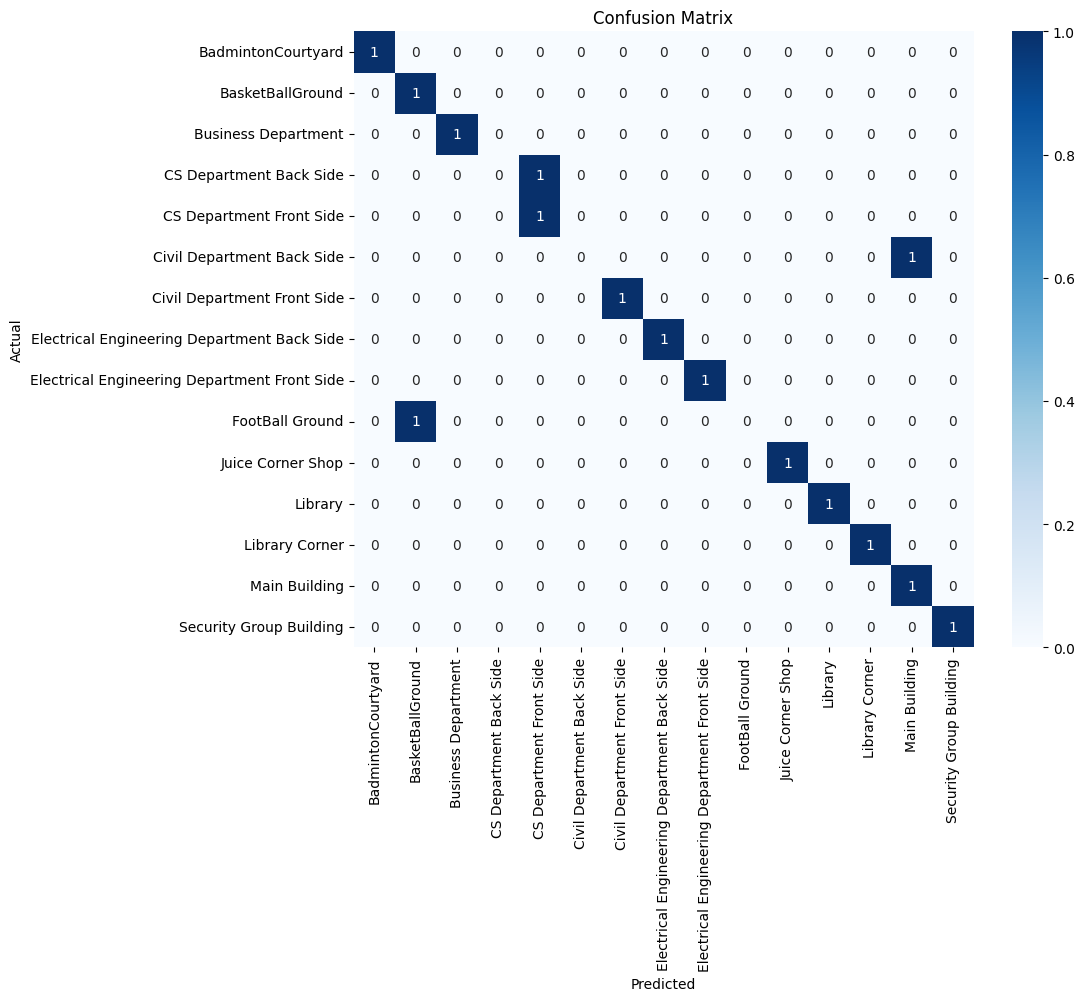
**3. Model Architecture**

* **Base Model:** MobileNetV2 (pre-trained on ImageNet) without the top classification layer.
* **Feature Extraction:** Using pooling='avg', MobileNetV2 outputs a 1280-dimensional feature vector.
* **Classifier:** A custom fully connected neural network appended to the base model:
  + Dense layer with 1024 neurons (ReLU activation)
  + Dropout layer (rate=0.5)
  + Dense output layer with softmax activation (number of neurons = number of building classes)

**4. Training Details**

* **Loss Function:** Sparse Categorical Crossentropy
* **Optimizer:** Adam
* **Batch Size:** 32
* **Epochs:** 10 (with early stopping and validation split used during training)

**5. Evaluation**

* **Metrics:** Accuracy, precision, recall, F1-score
* **Test Accuracy:** *0.8*
* **Confusion Matrix:**
* 
  + The confusion matrix was generated using sklearn's confusion\_matrix function and visualized with seaborn's heatmap.

**6. Model Conversion**

* The trained model was saved in multiple formats:
  + **Keras:** Mobile-BasedCampusNavigationAssistant.h5

**7. Conclusion**  
The landmark recognition model effectively classifies building images based on the extracted features. Although the dataset was limited (one image per class initially), the model demonstrates the feasibility of using transfer learning for landmark recognition. Future work will include expanding the dataset and experimenting with fine-tuning more layers.

**8. References**

* Keras & TensorFlow Documentation
* MobileNetV2 Research Paper
* Scikit-learn Documentation

**Phase 3**

In this phase we have used reference size array for each department, and we take bearing angle from the user to estimate his distance from the building and we have also implemented the triangulate method. We have set building coordinates as well to show each building on map. Basically, we take image from the user then preprocess the image and then we draw boundaries around the image then pass it to the model then model predict the image and tell its confidence value and then we show the user and all the departments on a map visually.



A graph with a dotted line

AI-generated content may be incorrect.

**Phase 4**

In this phase we simply designed a mobile app in java that takes pictures at runtime, or we can load images from gallery and then we make an Api of a model that runs locally on our laptop(server). Then this image will go to the model and the model will return the result and visual map as well.  
The images of mobile app attached as well.

A screenshot of a phone

AI-generated content may be incorrect.

A screenshot of a cellphone

AI-generated content may be incorrect.