NCE Journal of Science and Engineering (NJSE), Vol. 2, June 2025

AI Driven Monument Recognition: Developing a CNN-based Mobile App

Aayush Shrestha, Ishita Chalise, Pradish Tamrakar, Subash Panday[1,*]

[1]Department of Electronics and Computer Engineering, National College of Engineering, Lalitpur, Nepal

[*]Correspondence: subash@nce.edu.np

Manuscript received January 23, 2025; accepted April 8, 2025

***Abstract*—Monuments are historical and cultural artifacts that serve as evidence of a country's rich heritage. This system is developed to address the growing need for accessible information about Nepal's diverse monuments. Preserving the historical knowledge associated with heritage sites and monuments poses a significant challenge in modern Nepal. Many original writings and documents are deteriorating due to decay and loss, and the situation is worsened by the lack of focused preservation efforts. This mobile application seeks to bridge this historical gap by providing a platform for users to explore and engage with information about Nepal's monuments. The dataset used in this system comprises 2,000 images, with 100 photos taken from each side of five distinct monuments sourced from Patan Durbar Square. To enhance diversity and improve the dataset's robustness, augmentation techniques such as zooming, flipping, rotation, and shearing were applied, increasing the dataset size to 12,000 images. Additionally, 3,000 random images were added as a separate class to minimize the risk of misclassification. Each image was resized to 224 x 224 pixels and divided into training, validation, and test sets in a 7:2:1 ratio. Preprocessing steps included normalizing pixel values, introducing shading, adding noise, and grayscale conversion. The Convolutional Neural Network (CNN) architecture, specifically tailored for this application, achieved a training accuracy of 93.69.**

***Keywords*—Convolutional Neural Network, Deep Learning, Machine Learning, Mobile App, and Monument Recognition.**

I. INTRODUCTION

MONUMENTS are cultural and historical assets that serve as testimonies to a nation's heritage. In Nepal, a country rich in historical landmarks and UNESCO World Heritage sites, identifying and preserving these monuments is of great significance. However, physical records and descriptions of monuments are often stored on paper, which is prone to degradation over time. This creates challenges in disseminating accurate and comprehensive information about these structures to the general public and tourists.

Machine learning, particularly Convolutional Neural Networks (CNNs), has demonstrated remarkable success in image recognition tasks. MobileNet V2, known for its efficiency in lightweight applications, is particularly suitable for mobile environments [1]. Several studies, such as those by Crudge *et al.* [2] and Saini *et al.* [3], have explored landmark and monument recognition using CNNs, highlighting the potential of these methods in real-world applications.

The Monument Recognition System Using Mobile Application is developed to address the growing need for information about Nepal's diverse monuments. This mobile application provides a digital platform for exploring and engaging with Nepal's historical heritage, helping mitigate the loss of historical information caused by decaying and missing records. By focusing on five iconic monuments from Patan Durbar Square, the app preserves and enhances user interaction with Nepalese culture and history.

To develop this system, a custom dataset of 2,000 images was curated, including 100 images of each side of five distinct monuments in Patan Durbar Square. Dataset diversity was enhanced using augmentation techniques such as zooming, flipping, rotation, and shearing, expanding the dataset to 12,000 images. Furthermore, an additional 3,000 random im- ages were included as a separate class to improve classification robustness and avoid false positives.

All images were resized to 224 × 224 pixels and split into training, validation, and test sets in a 7:2:1 ratio. Preprocessing steps included normalizing pixel values, grayscale conversion, shading, adding noise, and enhancing features to simulate real-world conditions.

A refined MobileNet V2 architecture was employed for this task due to its balance of computational efficiency and accuracy. The model achieved a training accuracy of 93.69%, a validation accuracy of 98.9%, and an F1 score of 0.9783, demonstrating its robustness in classifying monuments.

The application, developed using Flutter and integrated with TensorFlow Lite, provides users with real-time monument recognition and detailed historical descriptions. It supports functionalities such as live camera recognition, image upload, and interactive descriptions, offering an intuitive and engaging user experience. This system bridges the gap between technology and cultural preservation, enabling accessibility for both locals and tourists.

III. Related Works

Numerous research papers have been published in the domain of image recognition, using techniques such as Histogram of Oriented Gradients (HOG) descriptors, Support Vector Machines (SVM), Deep Neural Networks (DNN), Random Forest algorithms, and Convolutional Neural Networks (CNN). Among these approaches, our work focuses on the use of CNN for image recognition tasks.

Andrew Crudge et al. proposed Landmark Recognition Using Machine Learning [2] which presents detection of build- ings in image, the image is cropped into multiple overlapping cells with identical aspect ratios and features are extracted from the cells using the HOG descriptor and classified using the SVM algorithm. The proposed model used a dataset consisting of 193 images of various buildings collected from Google Images. To improve the performance of model, each classifier was run 20 times varying the training and test sets by randomly permuting the dataset. The accuracy of the SVM classifier approaches 90%.

Siddhant Gada et al. presented Monument Recognition using Deep Neural Networks [4] which shows the concept of Transfer learning which has been used to prune the com- computational load, dataset of about 400 images per monument were used to retrain the final layer of the Inception model. The model is tested on a few arbitrary images and results with a training accuracy of 99.4% and corresponding testing accuracy ranging from 96-99%.

Mehdi Etaati et al. presented Cross Platform Web-based Smart Tourism Using Deep Monument Mining which uses a deep neural network (VGGNet model; transfer learning technique is used) for feature extraction from the images captured on a mobile phone and a classification algorithm: SVM and Random Forest are used for identification of monuments in the image. image uploaded to the web server detects the monument in the image, extracts its information from the database, and sends the results to the device [5].

Valerio Palma et al. proposed Towards Deep Learning for Architecture: A Monument Recognition Mobile App [6] that use data augmentation techniques on the dataset of roughly 50–100 images per monument to generate 500 training images per monument which is trained using CNN algorithm based on the MobileNet model implemented in Python using the open-source libraries TensorFlow and Keras. A commercial iOS app was developed with an overall accuracy of the trained models estimated to be over 95%.

Ramsha Fatima et al. proposed a Mobile Travel Guide using Image Recognition and GPS/Geo Tagging A Smart Way to Travel project that uses Open-CV for image recognition and GPS navigation system with Google Maps API to point the location of that monument on the map. The information associated with monuments is available in the JSON database. To identify the monument, the camera's captured image is compared to images stored in the database using Open- CV. The application compares two images using histogram comparison and feature detection using the ORB method [7]. Aradhya Saini et al. presented Image based Indian Monument Recognition using Convoluted Neural Networks where experiments on different method of extraction of features using hand crafted features like HOG, LBP and GIST and then CNN were performed. The manually acquired dataset comprises of 100 different monuments with 50 images per monument. HOG, LBP and GIST features extracted on training dataset classified by classification algorithms like SVM obtained very low accuracy. Finally, the DCNN model obtained accuracy of 92.7% using fc6 layer [3].

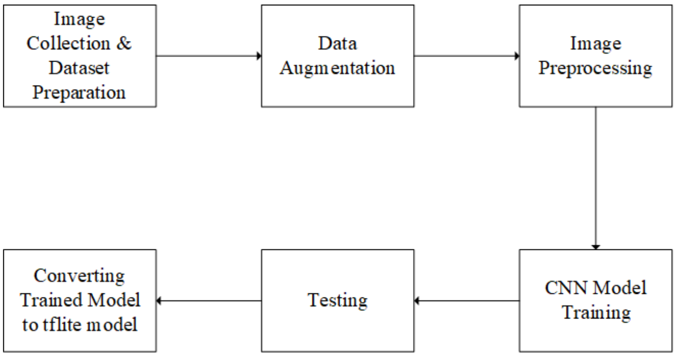


Image Collection & Dataset Preparation
Data Augmentation
Image Preprocessing
CNN Model Training
Testing
Converting Trained Model to tflite model

Fig. 1. Block diagram of model development process.

III. Methodology

The block diagram presents a systematic flow of tasks and components essential to our model development process. Beginning with data collection and preprocessing, the flow seamlessly Aradhya Saini et al. presented transitions to CNN model development and training stages. Subsequently, the model undergoes rigorous testing and validation procedures to ensure its robustness and accuracy. Following evaluation, the model is converted into a tflite model format, enhancing its compatibility and efficiency for deployment on mobile platform. The diagram serves as a visual representation of the journey from initial data processing to the implementation of advanced model applications, ultimately facilitating accurate monument classification and recognition.

A. Data Collection

The dataset consists of carefully selected images displaying the architectural wonders found in Patan Durbar Square. Each of the five renowned monuments - ”Chaar Narayan Mandir,” ”Chyasin Dega,” ”Krishna Mandir,” ”Shree Bhimsen Mandir,” and ”Vishwanath Mandir” - was photographed from all four sides (East, West, North, and South), resulting in a collection of 100 images per side and a total of 2000 images. These images were captured on-site to capture the unique features of each monument from various perspectives, forming the core of the dataset. Additionally, 3000 random images were added as a separate class to avoid misclassification of random images as monuments.

B. Data Augmentation

To enhance dataset diversity and resilience, sophisticated data augmentation techniques such as random rotations, flips, shear, and zoom were applied to images of 5 monuments, resulting in an expanded dataset of 15,000 images. This extensive dataset encapsulates the essence of Patan’s cultural heritage and serves as a strong foundation for monument classification task.

C. Preprocessing

In the preprocessing phase, several key steps were implemented to enhance the quality of the dataset and prepare it for effective model training. The dataset was split into training, validation, and test sets in the ratio of 7:2:1. This division ensures a substantial portion of the data is dedicated to training the model, while the validation set serves as a measure of the model’s performance on unseen data, and the test set evaluates the final model’s generalization.

To facilitate model training and evaluation, all images were resized to a standardized dimension of 224x224 pixels, promoting consistency in input dimensions across the dataset. Subsequently, normalization of pixel values was performed, scaling the data to a range of [0, 1]. This normalization aids in stabilizing the training process and preventing issues like exploding or vanishing gradients.

To simulate real-world lighting conditions, shading effects were introduced, adding a layer of complexity to the dataset. Additionally, a controlled amount of noise was deliberately added to a subset of images, replicating imperfections commonly found in photographs. This step is instrumental in training models to handle noisy input, contributing to improved overall performance.

Furthermore, a subset of photos underwent grayscale conversion, transforming images into black and white. This allows models to focus on essential features and patterns during training, especially useful when color information may not be critical for the task.

These preprocessing steps, combined with the careful dataset split, collectively contribute to the creation of a robust dataset, marking a crucial phase in the development of the models.

D. Preprocessing

In the model training phase, the prepared dataset takes center stage for training the model to comprehend and recognize patterns within the dataset. The images, normalized and augmented with shading effects and controlled noise, serve as inputs for the model, with grayscale conversion ensuring simplicity and focus on essential features. The CNN architecture was carefully chosen to facilitate effective learning. The hyperparameters, such as learning rate and batch size, were tuned to enhance the model’s performance. Over multiple epochs, the model iteratively refines its understanding of the dataset, progressively improving its ability to recognize complex patterns. The validation set, initially split from the dataset, plays a critical role in assessing the model’s ability to generalize to unseen data. The model based on Convolutional Neural Network (CNN) was employed for the training phase.

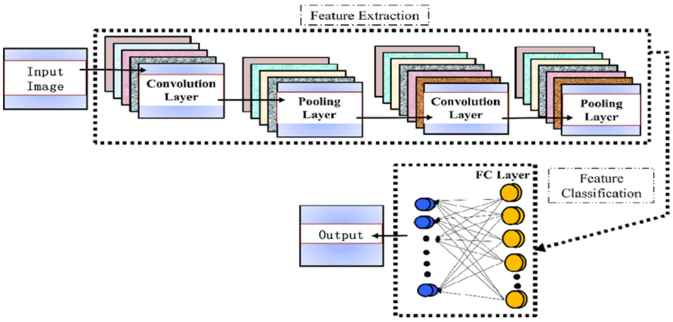


Fig. 2. Architecture of CNN [8]

The model consists of multiple layers designed to capture intricate patterns within the dataset. They are briefly described below:

- **Input Layer:** This layer defines the input shape for the network, which is an image with dimensions 224x224 pixels and 3 color channels (RGB).
- **MobileNetV2 Model:** This represents the MobileNetV2 model, which is a convolutional neural network designed for mobile and embedded vision applications. The model extracts feature from the input image and outputs a tensor with dimensions (None, 7, 7, 1280). The ’None’ dimension represents the batch size, which can vary during training and inference.
- **Global Max Pooling 2D Layer:** This layer performs global max pooling across the spatial dimensions of the feature maps output by the MobileNetV2 model. It reduces the dimensions from (None, 7, 7, 1280) to (None, 1280) by taking the maximum value along each channel.
- **Dropout Layer:** Dropout is a regularization technique used to prevent overfitting. It randomly sets a fraction of input units to zero during training to prevent them from co-adapting and forcing the network to learn more robust features.
- **Dense Layers (Dense):** These layers are fully connected layers used for classification. They transform the input data received from the previous layer into output data that represents the probability scores for each class. The number of units in the last dense layer matches the number of output classes (6 in this case), and it uses the softmax activation function to produce class probabilities.
- **Compilation:** The model is compiled using the ’RMSprop’ optimizer and ’categorical_cross entropy’ loss function. Categorical Cross-Entropy is suitable for multi-class classification tasks.

E. Performance Metrics

Accuracy: Accuracy measures the proportion of correctly classified instances among the total number of instances. Mathematically, it is expressed as:

$$Accuracy = \frac{TP+PN}{TP+TN+FP+FN} \quad (1)$$

where *TP* is True Positives, *TN* is True Negatives, *FP* is False Positives, and *FN* is False Negatives.

Precision: Precision evaluates the fraction of True Positive predictions among all positive predictions made by the model. It indicates the model's ability to minimize False Positives. The formula is:

$$Precision = \frac{TP}{TP+FP} \qquad (2)$$

Precision is particularly important in applications like spam detection, where minimizing False Positives is critical.

Recall (Sensitivity or True Positive Rate): Recall measures the proportion of actual positive instances that the model correctly identifies. It evaluates the model's ability to minimize False Negatives. Mathematically:

$$Recall = \frac{TP}{TP+FN} \qquad (3)$$

Recall is especially crucial in fields like medical diagnosis, where missing a True Positive can have severe consequences.

F. Flow Diagram of Application

The provided flowchart outlines the process of monument recognition and historical information display in a mobile application.

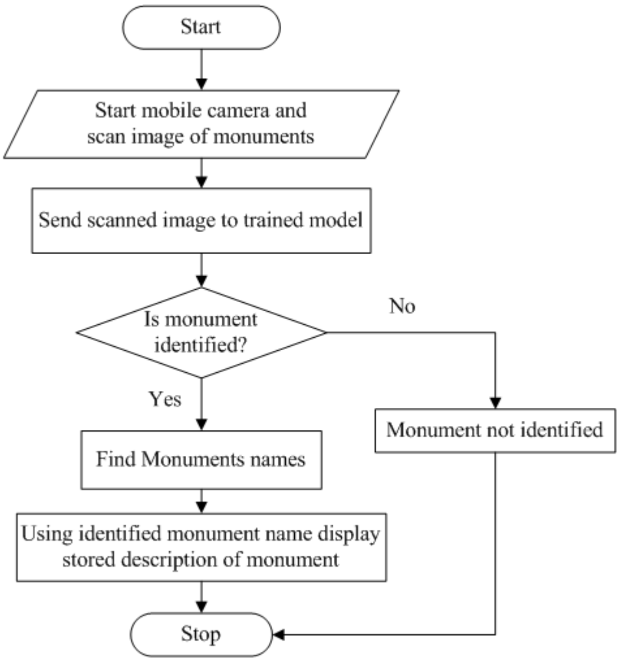
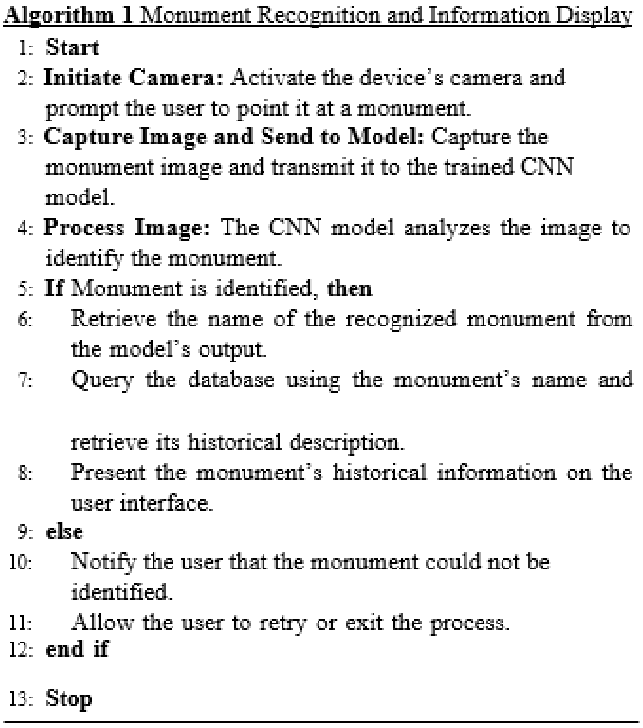

Start
Start mobile camera and scan image of monuments
Send scanned image to trained model
Is monument identified?
No
Yes
Monument not identified
Find Monuments names
Using identified monument name display stored description of monument
Stop

Fig. 3. Workflow of the System.



Algorithm 1 Monument Recognition and Information Display

```
Start
Initiate Camera: Activate the device's camera and prompt the user to point it at a monument.
Capture Image and Send to Model: Capture the monument image and transmit it to the trained CNN model.
Process Image: The CNN model analyzes the image to identify the monument.
If Monument is identified, then
    Retrieve the name of the recognized monument from the model's output.
    Query the database using the monument's name and

    retrieve its historical description.
    Present the monument's historical information on the user interface.
else
    Notify the user that the monument could not be identified.
    Allow the user to retry or exit the process.
end if
Stop
```

III. Results and Discussion

A. Data Collection

The dataset was compiled featuring monuments from Patan Durbar Square, with a total of 2000 photos. This involved capturing 100 images for each side (East, West, North, and South) at five distinct monuments: "Chaar Narayan Mandir," "Chyasin Dega," "Krishna Mandir," "Shree Bhimsen Mandir," and "Vishwanath Mandir". Additionally, 3000 images of random objects were added as a separate class. This dataset forms the foundation of the project, serving as the primary material for training the model.

B. Data Augmentation

Different data augmentation techniques were applied on images of each of the monuments to enhance the diversity of the dataset, incorporating random rotations, flips, shear, and zoom. Each image underwent augmentation five times, resulting in a total of six images, including the original. Following augmentation, the dataset expanded to 15,000 images. Additionally, all images are resized to 224x224 pixels.

C. Data Preprocessing

The augmented dataset was subsequently partitioned into 70% training set, 20% validation set and 10% test set. The train set underwent a series of transformations to enhance its readiness for training. Normalization ensured pixel values fell within a 0-1 range, shading effects simulated diverse lighting conditions,

underwent a series of transformations to enhance its readiness for training. Normalization ensured pixel values fell within a 0-1 range, shading effects simulated diverse lighting conditions,

controlled noise simulated real-world image imperfections, and

Fig. 4. Sample of augmented images.

Fig. 5. Sample images after shading.

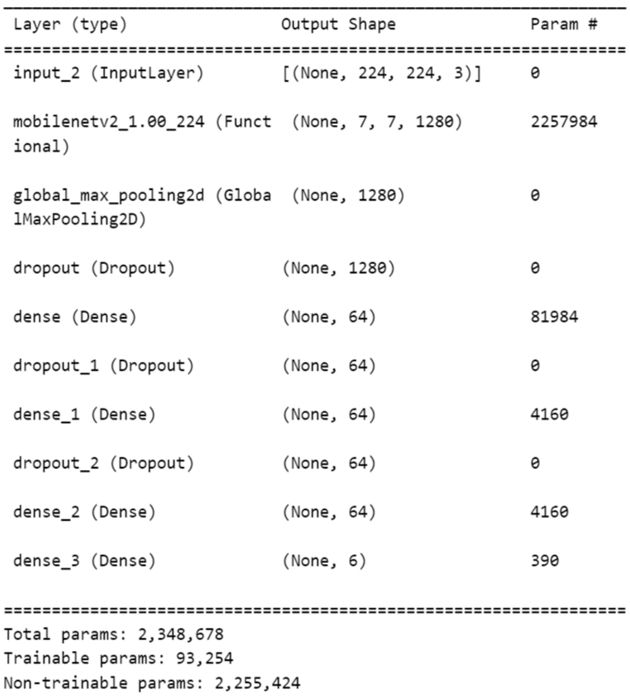
Fig. 6. Sample images after adding noise.

Fig. 7. Sample images after grayscale conversion.

grayscale conversion produced a monochrome representation of Chaar Narayan Temple, 238 instances of Chyasin Dega, 237 instances of Krishna Mandir, 230 instances of Shree Bhimsen Temple, and 230 instances of Vishwanath Temple. However, it Temple, 1 Chyasin Dega as Krishna Mandir, 1 Chyasin Dega as Not a monument, 3 Krishna Mandir as Chyasin Dega, 4 Shree Bhimsen Temple as Chaar Narayan Temple, 1 Shree Bhimsen Temple as Krishna Mandir, 3 Shree Bhimsen Temple as Not a monument, 3 Vishwanath Temple as Chaar Narayan Temple, and 4 Vishwanath Temple as Krishna Mandir. This detailed breakdown provides insights into the model's strengths and areas requiring improvement.

In addition to assessing accuracy, the model's performance is further evaluated using the average F1 score, which is notably high at 0.9783. Moreover, average precision and average recall are reported at 0.98 and 0.9767, respectively. This metric provides a measure of the model's ability to correctly classify and capture relevant patterns within the dataset.

Fig. 10 illustrates the training and validation accuracy of the model over 80 epochs. The blue line represents the training accuracy, which steadily increases and reaches 93.69% by the end of training. The orange line shows the validation accuracy, which also improves rapidly before stabilizing at a high level, reaching an accuracy of 98.23%. The small gap between training and validation accuracy suggests good generalization, with minimal



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]     0

 mobilenetv2_1.00_224 (Funct  (None, 7, 7, 1280)       2257984
 ional)

 global_max_pooling2d (Globa  (None, 1280)             0
 lMaxPooling2D)

 dropout (Dropout)           (None, 1280)              0

 dense (Dense)               (None, 64)                81984

 dropout_1 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 64)                4160

 dropout_2 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 64)                4160

 dense_3 (Dense)             (None, 6)                 390

=================================================================
Total params: 2,348,678
Trainable params: 93,254
Non-trainable params: 2,255,424
```

Fig. 8. Model showing layers, output shape, and parameters.

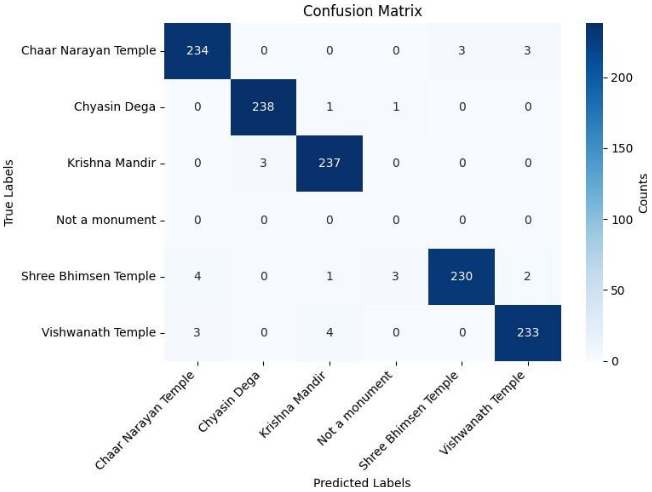


Fig. 9. Confusion matrix of model.

TABLE I
Classification Report Metrics for Monument Recognition

Chaar Narayan Temple	0.97	0.97	0.97	240
Chyasin Dega	0.99	0.99	0.99	240
Krishna Mandir	0.98	0.99	0.98	240
Not a Monument	0.00	0.00	0.00	0
Shree Bhimsen Temple	0.99	0.96	0.97	240
Vishwanath Temple	0.98	0.97	0.97	240
Accuracy			0.98	1200
Macro Avg	0.82	0.81	0.82	1200
Weighted Avg	0.98	0.98	0.98	1200

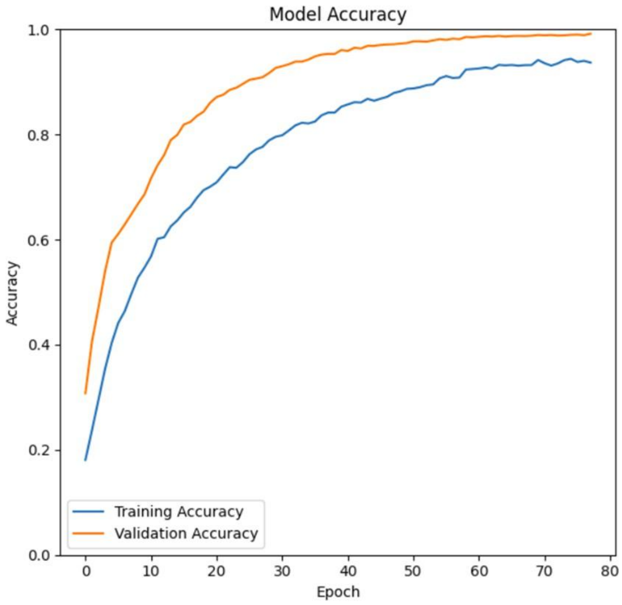


Fig. 10. Training accuracy and validation accuracy graph.

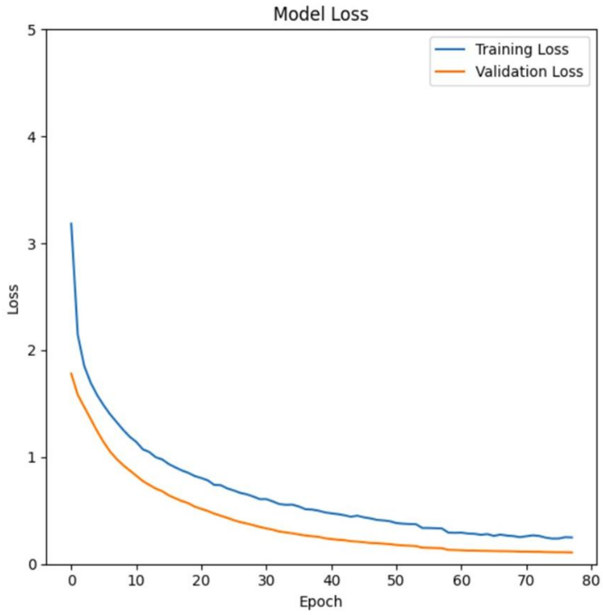


Fig. 11. Training loss and validation loss graph.

signs of overfitting. Fig.11 depicts model loss during training and validation at each epoch. The training logs illustrate a continuous decline in loss misclassified several instances: 3 Chaar Narayan Temple as Shree Bhimsen Temple, 3 Chaar Narayan Temple as Vishwanath values over successive epochs. The diminishing loss values signify a decreasing disparity between predicted and actual values, reflecting improved model convergence. Both training and validation losses exhibit consistent downward trends, indicating that the model learns to better approximate the target outputs. The parallel decline in training and validation losses underscores the model's ability to generalize well to unseen data, crucial for its performance in real-world scenarios.

D. Application Development

An app was developed using Flutter to create an intuitive and immersive application. The application serves as a gateway to explore Nepal's rich cultural heritage embodied in the monuments of Patan Durbar Square. Integrating TensorFlow Lite, the application empowers users with real-time monument recognition and classification, providing detailed insights into each monument's historical significance and architectural marvels. The commitment to user-centric design is evident in the seamless navigation and engaging features tailored to enrich users' exploration journey. As the application's capabilities are refined and expanded, the vision remains steadfast: to inspire curiosity, foster appreciation, and preserve Nepal's cultural legacy for generations to come. The application includes the following functions:

- Take a photo.
- Pick a photo from the gallery.
- Show a description about the selected/captured photo.
- Live camera.

V. Conclusion

The Monument Recognition Application holds great promise in improving tourism accessibility in Nepal by providing a simple and efficient platform to identify and locate monuments, particularly within Patan Durbar Square. With its user-friendly interface, real-time functionality, and monument recognition capabilities—achieving a remarkable training accuracy of 93.69% and validation accuracy of 98.9%-the app contributes significantly to preserving and promoting Nepal's cultural legacy. By offering historical context and insights into architectural heritage, it bridges the gap between tourists and the rich historical narrative of Nepal.

However, certain limitations persist, highlighting areas for future development. The dataset, focused exclusively on Patan Durbar Square, does not fully represent the diversity of monuments across the country. Challenges such as extreme angles, occlusions, and geometric transformations occasionally hinder accuracy. Addressing these limitations, future enhancements could include expanding the app to recognize monuments nationwide, supporting multiple languages, and integrating audio guides for an immersive user experience. These advancements would ensure broader accessibility and reinforce the app's role in preserving and disseminating Nepal's cultural heritage.

VI. Acknowledgement

The authors gratefully acknowledge the research grant provided by the National College of Engineering, Institute of Engineering, Tribhuvan University.

References

[1] M. Sandler *et al.*, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE confer- ence on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[2] A. Crudge *et al.*, "Landmark recognition using machine learning," *CS229, Project*, 2014.

[3] A. Saini *et al.*, “Image based indian monument recog- nition using convoluted neural networks,” in *2017 Inter- national conference on big data, IoT and data science (BID)*, IEEE, 2017, pp. 138–142.

[4] S. Gada *et al.*, “Monument recognition using deep neu- ral networks,” in *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, IEEE, 2017, pp. 1–6.

[5] M. Etaati *et al.*, “Cross platform web-based smart tourism using deep monument mining,” in *2019 4th International conference on pattern recognition and image analysis (IPRIA)*, IEEE, 2019, pp. 190–194.

[6] V. Palma, “Towards deep learning for architecture: A monument recognition mobile app,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 551–556, 2019.

[7] R. Fatima *et al.*, “Mobile travel guide using image recognition and gps/geo tagging: A smart way to travel,” in *2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN)*, IEEE, 2016, pp. 1–5.

[8] S. Ali, *Structure of cnn: There are some special struc- tural features in the cnn architecture*, https : / / www . researchgate . net / profile / Saqib - Ali - 33 / publication / 344807764 / figure / fig1 / AS : 949384070049794 @ 1603362209770 / Structure - of - CNN - There - are - some - special-structural-features-in-the-CNN-architecture.png, Accessed: 2025-01-21, 2020.