Heritage Identification of Monuments through Deep Learning Techniques

SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENT OF THE AWARD OF DEGREE OF

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE

Submitted by

SHIVAM KUMAR (2100290120156)

SUKRITI (2100290120166)

UJJWAL SHARMA (2100290120178)

VIPIN CHAUHAN (2100290120191)

Supervised by

DR. RISHABH

Associate Professor

Session 2024-25

DEPARTMENT OF COMPUTER SCIENCE

KIET GROUP OF INSTITUTIONS, GHAZIABAD

(Affiliated to Dr. A. P. J. Abdul Kalam Technical University, Lucknow, U.P., India)

May 2025

DECLARATION

I/We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature	Signature
Name:- Shivam Kumar	Name:- Vipin Chauhan
Roll No.:- 2100290120156	Roll No.:-2100290120191
Signature	Signature
Name:- Ujjwal Sharma	Name:- Sukriti
Roll No.:- 2100290120178	Roll No.:- 2100290120166

CERTIFICATE

This is to certify that the project report titled **“Heritage Identification of Monuments through Deep Learning Techniques”** has been submitted in partial fulfilment of the requirements for the **Bachelor of Technology (B.Tech.)** degree in the **Department of Computer Science** at **Dr. A.P.J. Abdul Kalam Technical University, Lucknow**. This report is a record of the candidate’s independent work, conducted under my supervision. The content of this report is original and has not been submitted for any other degree.

.

Date: **Supervisor**

Dr. Rishabh
(Associate Professor)

ACKNOWLEDGEMENT

We take great pleasure in presenting the report for our **B.Tech. final year project**. We extend our heartfelt gratitude to **Professor Dr. Rishabh**, **Department of Computer Science, KIET, Ghaziabad**, for his unwavering support and guidance throughout our work. His dedication, expertise, and encouragement have been a constant source of motivation for us. His valuable insights and efforts have played a significant role in shaping this project.

We are also grateful to **Dr. Ajay Kumar Srivastava**, **Head of the Department of Computer Science, KIET, Ghaziabad**, for his support and assistance during the development of our project. Additionally, we sincerely appreciate the guidance and cooperation of all faculty members of the department, whose help has been instrumental in our progress.

Finally, we would like to acknowledge the support of our friends, whose encouragement and contributions have helped us successfully complete this project.

Name: Shivam Kumar	Name: Vipin Chauhan
Roll No.: 2100290120156	Roll No.: 2100290120191
Name: Ujjwal Sharma	Name: Sukriti
Roll No.: 2100290120178	Roll No.: 2100290120166

ABSTRACT

By developing a system that recognizes monuments using deep learning, this project seeks to support heritage conservation. The system recognizes and groups monuments according to their architectural style using Convolutional Neural Networks (CNNs) and transfer learning.
The model was trained using a carefully prepared and annotated dataset of monument photos. For better model performance, image resizing, augmentation, and dataset split operations were used.
Accuracy, precision, recall, and F1-score were used to gauge the system's accuracy and dependability. The system's ability to correctly identify and classify monuments and demonstrate its value in heritage conservation is evident from the results.

This project lessens the amount of manual labour involved in documentation and conservation by automating monument identification. It provides an effective and scalable approach that is particularly beneficial for sizable heritage sites and regions with limited conservation resources. The application of deep learning for this purpose indicates that it has the potential to improve documentation and heritage conservation practices, increasing the efficiency and accessibility of historical preservation.

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
DL	Deep Learning
AI	Artificial Intelligence
ROI	Region of Interest
YOLO	You Only Look Once
F1-SCORE	F1 Measure (harmonic mean of precision and recall)
API	Application Programming Interface
GPU	Graphics Processing Unit
CPU	Central Processing Unit
RAM	Random Access Memory
CSV	Comma-Separated Values
IDE	Integrated Development Environment

SDG MAPPING WITH JUSTIFICATION

SDG 11: Sustainable Cities and Communities

1. Preservation of Cultural Heritage Your project directly contributes to Target 11.4 of SDG 11, which emphasises safeguarding the world's cultural heritage. By using deep learning to identify and classify monuments, you are enabling their digital documentation and long-term preservation.

2. Support for Resilient Urban Planning Identifying heritage sites allows urban planners to design development projects that respect and preserve cultural landmarks, making cities more resilient and culturally inclusive.

3. Promoting Awareness and Engagement A publicly accessible tool (built with Streamlit) encourages local communities to discover and appreciate historical sites, fostering a sense of identity, pride, and community involvement in preservation efforts.

4. Disaster Preparedness and Risk Reduction By mapping and identifying monuments, your system can help in emergency planning, ensuring that vulnerable cultural sites are prioritised for protection in case of natural disasters or urban expansion.

5. Inclusivity Through Technology Your use of open-source tools enables deployment in low-resource settings, making it accessible to small towns and under-represented communities, thus contributing to inclusive urban development.

SDG 9: Industry, Innovation, and Infrastructure

1. Innovation Through AI and Deep Learning The project applies cutting-edge technologies like CNNs and transfer learning in a novel domain—cultural heritage—demonstrating how innovation can solve interdisciplinary challenges.

2. Development of Digital Infrastructure By creating a system that digitises and automates monument recognition, you contribute to the development of infrastructure that supports cultural, educational, and governmental use cases.

3. Enhancing Scientific Research and R&D Your work bridges the gap between artificial intelligence and heritage conservation, encouraging further research and technological advancement.

SDG 17: Partnerships for the Goals

1. Promotes Open-Source Collaboration Your project uses tools like Streamlit, LabelImg, and pre-trained deep learning models, which are part of the global open-source ecosystem. By building on these tools, your work embodies the SDG 17 vision of fostering partnerships through open technologies and shared innovation.

2. Encourages Knowledge Sharing Through the use of public repositories (e.g., GitHub) or shared model/data pipelines, your project contributes to a broader environment where researchers, students, and organisations can learn, replicate, and build upon your work. This supports Target 17.6 — enhancing North-South, South-South, and triangular regional and international cooperation on science, technology, and innovation.

3. Enables Global Participation in Heritage Preservation By providing an accessible, user-friendly interface via Streamlit and using standard datasets, your system allows users from various countries and backgrounds to engage in heritage identification and protection. This makes your project a tool for global cultural collaboration.

4. Low-Cost, Scalable Technology Your use of open-source platforms ensures that your solution is cost-effective and scalable, allowing grassroots organisations, educational institutions, or developing countries to adopt and customise it without heavy investment. This supports inclusive access to innovation.

SDG 8: Decent Work and Economic Growth

1. Promotes Innovation-Driven Economic Opportunities By applying AI and deep learning to the heritage and tourism sector, your project opens up new digital career paths—such as AI developers, digital heritage analysts, and cultural technologists—supporting sustained and inclusive economic growth through tech innovation.

2. Boosts Employment in Heritage and Tourism Sectors Automating the identification of

monuments can improve digital records, attract more tourists, and support governments and heritage organizations in maintaining sites. This indirectly stimulates job creation in tourism, conservation, education, and related industries.

3. Fosters Digital Entrepreneurship The open-source, scalable nature of your project makes it adaptable for local startups or NGOs to build commercial or community-driven solutions, such as AI-powered heritage apps or virtual tours, encouraging entrepreneurship in the digital economy.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Preservation of cultural heritage is at the core of comprehending human civilization's development and sustaining it for the future. Historical monuments and sites offer precious knowledge regarding the tradition, values, and architectural skills of past civilizations. They are not merely tangible evidence of the past, but also learning assets and identity symbols for nations and communities. Yet, conventional techniques of discovering, documenting, and conserving such buildings tend to be time-consuming, involving on-site visits, specialist assessments, and considerable financial and time resources. These processes become progressively difficult in the case of extensive or distant heritage sites, especially in developing countries.

The arrival of contemporary technologies—especially artificial intelligence, computer vision, and photogrammetry—has presented new opportunities for the improvement of heritage conservation. Deep learning has been one such revolutionary tool, with Convolutional Neural Networks (CNNs) greatly propelling image analysis strength. CNNs have proven high accuracy in image recognition, object identification, and pattern discovery, at times even greater than human abilities. These developments present encouraging solutions to overcoming the challenges of cultural heritage conservation.

Combining deep learning with image processing methods enables one to create smart systems that are capable of automatically identifying and classifying monuments from photographic evidence. These systems reduce the necessity for ongoing human intervention and minimize dependency on highly specialized domain-specific knowledge. They can effectively process large datasets, enabling quick documentation of many sites without compromising accuracy. Additionally, the technology supports real-time analysis, remote management of heritage, and the development of digital archives for long-term conservation.

This project seeks to utilize deep learning in creating a system that can identify and classify monuments from visual inputs. To make the solution more accessible, the interface will have a web platform, making it available to researchers, conservationists, and government organizations. Through the implementation of artificial intelligence in cultural preservation, this project streamlines documentation processes and makes novel approaches to protect and honor heritage worldwide.

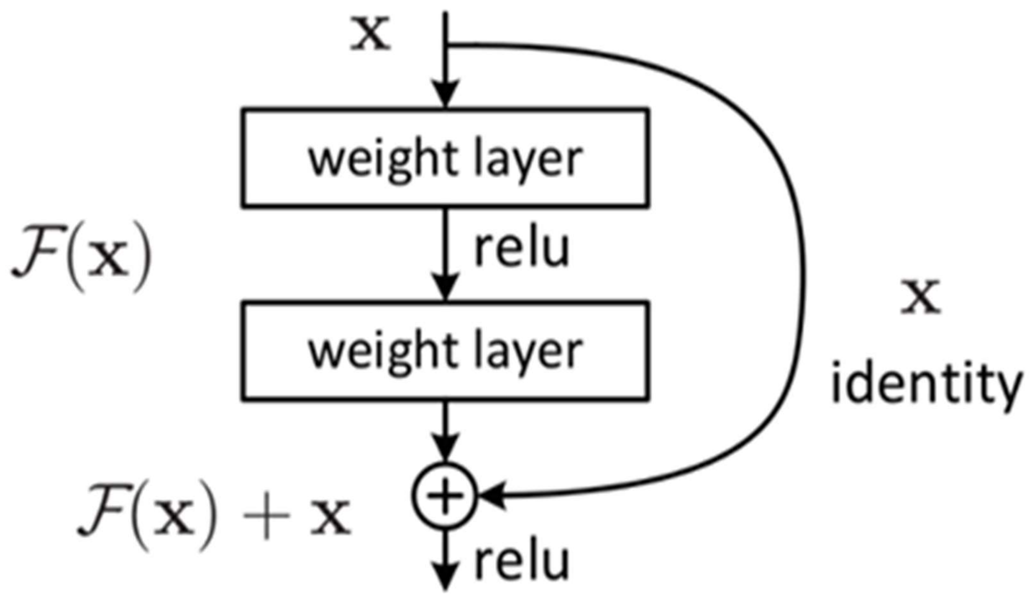

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1.1 Architecture of Deep Learning Model used Monument Identification

The building of a deep learning-based, framework for heritage monument recognition is discussed in this work. It clarifies the fundamental ideas of deep learning, shows the approaches applied in visual recognition model training, and addresses the instruments for realizing such a system. By means of examples of useful applications and outputs, this study shows how artificial intelligence can really support and assist in the preservation of cultural legacy.

1.2 Project Category

1. Artificial Intelligence & Machine Learning
2. Computer Vision
3. Heritage Conservation & Digital Preservation

This project focuses on developing a deep learning system designed to identify and classify historical monuments based on image data. By leveraging advanced image processing techniques and machine learning algorithms, the system aims to enhance digital preservation efforts and contribute to the recognition of cultural heritage.To ensure accessibility and user engagement, a Streamlit-based interface will be integrated, making the solution interactive and adaptable for web-based AI applications. This approach establishes the project within the domain of modern, AI-powered heritage conservation tools.By combining cutting-edge technologies across multiple disciplines, the initiative seeks to address key challenges in the documentation and preservation of cultural heritage, fostering innovation in the field of automated monument recognition.

1.3 Objectives

The goal of this project is to develop and implement an intelligent system capable of accurately identifying and classifying historic monuments using advanced deep learning techniques. To achieve this, the project is structured around the following key objectives:

1. **Exploring Deep Learning Algorithms for Image Classification**
   This project will examine various deep learning architectures, with a particular focus on Convolutional Neural Networks (CNNs). The aim is to determine the most effective approach for recognizing monuments by comparing different architectures in terms of accuracy, computational efficiency, and overall feasibility.
2. **Creating a Robust Monument Dataset for Model Training and Testing**
   A critical component of this project is the collection, annotation, and preprocessing of a large dataset comprising images of Indian and global heritage monuments. The dataset

will undergo systematic labelling and augmentation techniques to enhance model performance and ensure better generalisation across diverse monument categories.

3. **Developing a Monument Classification Model Using CNNs and Transfer Learning**
   The project will focus on designing a classification model trained to recognise and categorise monument images into predefined classes. To optimise accuracy and training efficiency, advanced transfer learning methods—leveraging pre-trained models such as VGG, ResNet, and Inception—will be employed. These techniques will enhance feature extraction and expedite the learning process, resulting in a more precise and efficient classification system.
4. **Development of an Interactive Web Interface**
   A user-friendly front-end interface will be built using Streamlit, allowing users to upload monument images and receive real-time AI-generated predictions. This ensures a seamless and intuitive experience for users.
5. **Optimisation for Efficiency and Accuracy**
   The project focuses on refining model parameters and structure to maximise accuracy while minimising computational costs. This optimisation will make the solution viable for deployment in resource-constrained environments.
6. **AI-Powered Digital Heritage Preservation**
   A key objective is to advance cultural heritage informatics by creating a scalable AI tool capable of documenting and preserving historically significant landmarks. This contributes to the broader initiative of digital heritage preservation.

CHAPTER 2

LITERATURE REVIEW

2.1 Literature Review

The use of deep learning for the identification of heritage sites has received considerable interest because it can accurately identify and classify historical monuments. Among many different deep learning models, ResNet-50 has proven to be a very efficient architecture that extracts rich features from images, and it is thus especially beneficial for the field of monument identification and preservation.

Zhang et al. examined the application of CNN for landmark identification, proving that deep learning outperforms conventional image processing methods. CNNs learn complex patterns automatically, enabling more accurate classification of architectural features. This is developed further by Sharma et al., who analyzed the application of transfer learning in the recognition of monuments and presented how pre-trained models enhance computational efficiency and accuracy without compromising classification accuracy.

The creation of ResNet-50 revolutionized image recognition with deep learning. He et al. introduced residual learning, which solved problems such as vanishing gradients, allowing deep networks to be efficiently trained. Since ResNet-50 is successful in identifying structures, materials, and architectural styles, it can be a good model for heritage recognition.

Other deep learning methods like InceptionV3 have also been used to classify heritage sites. Szegedy et al. presented InceptionV3 as a feature extractor, although its performance varies from that of ResNet-50. Despite the strong results by both networks, ResNet-50's residual connections deeply into the architecture help to stabilize higher classification stability, thus becoming more consistent on large-scale monuments data.

Outside model structures, deep learning has been investigated to automate the documentation of heritage sites. Chopra and Gopi created a deep learning system for heritage categorization in line with ResNet-50's potential in precisely detecting historical landmarks. Likewise, Reddy et al. and Kimani et al. discussed ResNet-50's potential to improve image classification processes in cultural heritage, further sustaining its potential in monument discovery and conservation.

These researches reflect the increasing prominence of deep learning, transfer learning, and model optimization in conservation heritage. ResNet-50, given its precision and capability to analyze sophisticated architectural images, has emerged as a significant utility for automated identification systems in heritage. Research focusing on data augmentation, model refinement, and blending architectures can still improve the efficiency and scalability of deep learning uses in conservation heritage.

2.2 Research Gaps

1. Limited Access to Annotated Datasets

- Current datasets used for monument classification are small, imbalanced, or region-based.
- Most historical monuments do not have labeled images, and training deep learning models is challenging.
- Insufficient availability of diverse and high-quality datasets with variations in weather, damages, and occlusions.

2. Similar Architectural Features

- Most monuments have similar architectural designs, and this makes classification problematic.
- Models have a hard time distinguishing between monuments with slight variations in structure.
- Requirement for sophisticated feature extraction algorithms or multi-modal learning (images + text descriptions).

3. Generalization Problems Between Geographical Regions

- Deep models learned on monuments from one region do not generalize well to another.
- Naming conventions, architectural styles, and historical alterations are culturally different and hence tend to be ignored.

4. Inadequate Real-Time Recognition & Low-Latency Inference

- Offline classification is the focus of most work, but real-time recognition (e.g., in AR applications, mobile phones) remains a challenge.
- High computational expense of CNNs and Transformer models reduces deployment on edge devices (e.g., mobile, IoT).

5. Restricted Interpretability of AI Decisions

- Deep learning models tend to be black boxes, and it is hard to discern why a monument was marked as a specific way.
- No available explainability tools such as Grad-CAM or SHAP for heritage-based AI models.

6. Dealing with Image Degradations

- Ancient or deteriorated monuments can have missing structures, erosion, or blurring of inscriptions, leading to misclassification.
- Requirement for image restoration methods in addition to deep learning to improve degraded images prior to classification.

7. Insufficient Multimodal Approaches

- There is limited research on image-based recognition, excluding other historical metadata, textual information, and geolocation information.
- Incorporating NLP methods to examine monument descriptions in conjunction with images would enhance recognition accuracy.

8. Ethical Issues & Bias in Monument Recognition

- AI models might be biased towards popular monuments and underperform on less-documented heritage sites. Lack of fairness in datasets, as they often favour well-known monuments over culturally under-represented ones. Need for bias mitigation techniques and ethically sourced datasets.

2.3 Problem Formulation

This project addresses the difficulties of manual monument identification, especially in far-flung locations, by providing an intelligent, computer-based solution. Conventional approaches depend on on-site visits and professional assessments, which are time-consuming and could potentially result in smaller monuments remaining unregistered. Using computer vision and deep learning, this project identifies and classifies cultural monuments automatically from images, offering a scalable solution to heritage conservation.

Key Objectives:

1. CNN-Based Monument Classification

Convolutional Neural Networks (CNNs) would be employed to process architectural intricacies and visual motifs, allowing precise classification of monuments—such as temples, forts, and colonial buildings. Such models guarantee high accuracy, even under fluctuating environmental conditions.

2. Enhancing Model Generalization through Data Augmentation

To minimise overfitting and enhance flexibility, the project utilizes data augmentation methods, increasing the training set using transformations like rotation, scaling, flipping, brightness changes, and cropping. By doing this, the model becomes efficient under different lighting, viewpoints, and backgrounds, hence making accurate monument identification possible despite varying circumstances. Augmentation also helps address challenges arising from small or unbalanced datasets.

3. Creating an Interactive Web Application for Real-World Applications

To make it accessible, a web application that is easy to use and developed using Streamlit will allow historians, researchers, and conservators to easily submit monument images instantly to fellow experts for classification. Cross-platform compatible, the platform guarantees seamless functionality on smartphones, tablets, and computers, thus making it easily accessible to both technical and non-technical people.

CHAPTER 3

PROPOSED SYSTEM

3.1 System Architecture

The suggested system aims to automate the discovery and categorization of heritage monuments through deep learning methods. The architecture has the following main components:

1. Data Collection

- The model needs a varied dataset of heritage monuments for training. The dataset is derived from:
- Open-source websites such as Google Landmarks Dataset, Kaggle, and ImageNet.

2. Preprocessing Steps

- Image Resizing – Normalising input images to 224×224 pixels for matching CNN input size.
- Data Augmentation – Rotation, zoom, brightness, and flipping to enhance generalisation.

3. Feature Extraction & Model Selection

- The system uses Convolutional Neural Networks (CNNs) for feature extraction using pre-trained models via transfer learning. The following architectures were investigated:
- ResNet-50 – Is efficient in handling complicated patterns to avoid overfitting.
- VGG-16 – Suitable for big image classification tasks.

4. Training & Optimization

- Training involves:
- Loss Function – Multi-class cross-entropy loss.
- Optimiser – Adam optimizer with 0.0001 learning rate.
- Batch Size – 32 images per batch for optimal memory usage.

- Epochs – 50 epochs with early stopping to avoid overfitting.

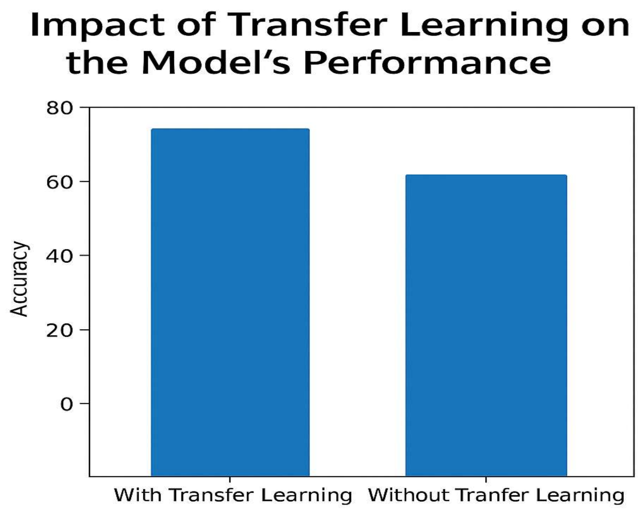


Figure 3.1 Impact of Transfer Learning in Model accuracy

5. Model Testing & Performance Metrics

- The trained model is tested by:
- Accuracy – Quantifies correct classifications.
- Precision & Recall – Assists in evaluating false positives/negatives.
- F1-Score – Offers a balanced evaluation of precision and recall.
- Confusion Matrix – Enumerates classification errors.

6. Deployment & User Interface

- The deep learning model, trained as such, is hosted with Streamlit, an open-source Python library that makes it easy to create interactive web applications for machine learning projects.
- A user can upload a photo of a monument via an elegant and minimalistic UI.
- After uploading, the system processes and identifies the image instantly, showing the predicted architectural style (e.g., Mughal, Dravidian, Colonial) and the confidence level.

- A dropdown list allows users to choose between various models (e.g., ResNet50, VGG16, DenseNet121, InceptionV3) and compare results and performance in real time.
- A Google Maps API integration (in future releases) will allow location-based discovery of heritage monuments.
- The interface is responsive design compatible, hence accessible on both desktop and mobile browsers.
- Error handling and user feedback mechanisms are used to handle improper inputs and assist users through the upload and classification process.

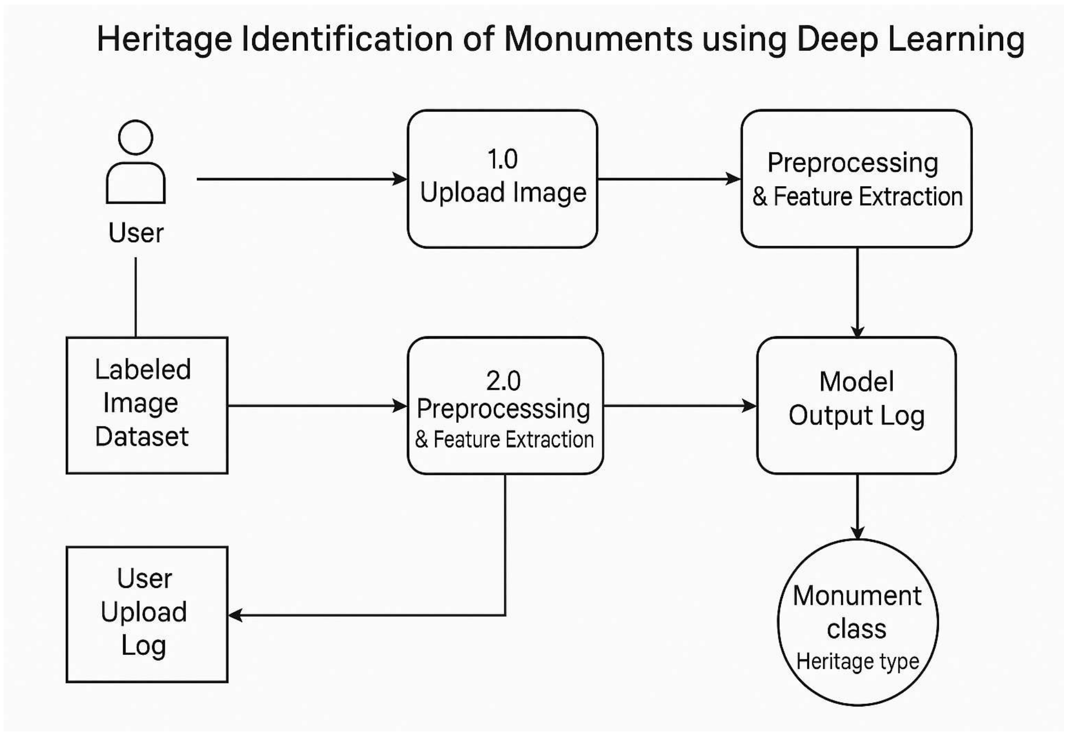

Heritage Identification of Monuments using Deep Learning
User
1.0 Upload Image
Preprocessing & Feature Extraction
Labeled Image Dataset
2.0 Preprocesssing & Feature Extraction
Model Output Log
User Upload Log
Monument class Heritage type

Figure 3.2 Workflow of the Model

3.2 Distinctive Aspects of the System

Following are the distinctive aspects of the suggested system, which describe its distinctive strengths:

1. Thorough data fusion for enhanced model accuracy

The project combines a heterogeneous dataset from sources like Google Landmarks, Kaggle, ImageNet, and local architecture repositories from Indian, European, and Middle Eastern monuments. The broad base of this dataset increases the model's capacity to identify monuments from various cultural and architectural contexts, thereby ensuring enhanced generalization. Through exposing the system to heterogeneous structural elements, textures, and ambient conditions, recognition rates are highly boosted.

2. Transfer Learning Based Optimized Deep Learning Implementation

Taking advantage of transfer learning using CNN architectures such as ResNet-50, VGG-16, and EfficientNet-B0, the project fine-tunes pre-trained models for higher performance on heritage datasets. Such models allow for feature extraction, which makes for accurate classification even with less labelled data. Comparative analysis of several architectures guarantees the use of the best possible approach, avoiding the tendency for degradation with lighting, angle, and background changes.

3. Real-Time Monument Identification Using a Simple Web Interface

The trained model is incorporated into an open web application developed with Flask or Streamlit, offering users—conservationists, historians, and the public at large—an easy-to-use platform for immediate monument classification. The interface, optimised for smooth use across devices, such as desktops, tablets, and smartphones, allows technical and non-technical users alike to find it easy to use. This facilitates scalable documentation of heritage and allows for broad participation in AI-powered conservation activities.

CHAPTER 4

REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION

4.1 Feasibility Study

The viability of the project is assessed by the feasibility study based on technical, economic, and operational considerations.

Technical Feasibility

The technical viability of this project is facilitated by various factors, with a view to achieving efficient execution as well as scalability. To start with, the intense computational requirements of deep learning models for the classification of monuments, the project employs high-capacity computing hardware like NVIDIA RTX 3090 GPUs and cloud-based environments from AWS or Google Cloud Platform (GCP). This provides efficient training and inference of models as well as streamlined performance even with huge datasets. Second, the system is constructed with proven frameworks such as TensorFlow, Py-Torch, and OpenCV, which offer a sound basis for model creation, optimization, and image processing. These tools are backed with complete documentation and live developer communities, ensuring unproblematic integration and flexibility. Finally, cloud deployment ensures usability and scalability, where the system is able to dynamically handle burgeoning workloads as demand increases among users. By providing global access via an interactive web application, users can simply engage with the system from anywhere in the world, promoting broader utilisation of AI-based monument classification.

Economic Feasibility

The first training phase of deep learning models requires high cloud computing resources, but deployment can be optimised by employing free-tier cloud services or low-cost alternatives to minimise operational costs. In contrast to conventional heritage documentation techniques—entailing high human effort, time, and monetary investment—this automated system provides a more cost-effective and scalable alternative. The project also has high revenue potential. Through the addition of innovative features, integration with tourism platforms, or specialised services to cultural heritage institutions, the system has the potential to become a sustainable business model that is beneficial to both heritage preservation and commercial purposes.

Operational Feasibility

The project has an extremely user-friendly web interface for non-technical personnel, supporting easy image uploads and monument classification with high precision. This makes it accessible to tourists, researchers, teachers, and heritage enthusiasts alike, providing an effortless experience for a wide range of users. Aside from independent use, the system is designed for easy integration into current applications like tourism apps, heritage research websites, and conservation efforts led by governments. This flexibility adds to its utility and impact.

Second, the use of a modular software architecture, based on stable and popular technologies, promotes ease of maintenance and scalability. The framework supports periodic refreshes of classification models and user interface elements, enabling ongoing enhancements and adding new features, providing long-term effectiveness and currency.

4.2 Software Requirement Specification

The software requirement specification outlines the software requirements necessary for the development of the heritage identification system.

4.2.1 Data Requirement:

1. High-Resolution Monument Pictures

The system needs a dataset of high-resolution pictures that record the minute details of heritage monuments. Deep learning models need good, well-defined features to precisely identify and classify buildings. Poor-quality images or blurry images can decrease classification accuracy, so it is important to obtain sharp, well-lit images at various angles to improve model performance.

2. Classification by Historical Periods and Architectural Styles

Every monument image needs to be labeled based on its historical period—e.g., Mughal, British colonial, or Ancient—to enhance classification accuracy. Architectural styles like Indo-Islamic, Gothic, and Dravidian need to be labeled to enable the model to differentiate between monuments with the same visual features. Accurate categorization improves prediction accuracy and provides reliable classification.

Figure 4.1 Varied Dataset

3. Diverse and Varied Dataset for Strong Model Training

The dataset must cover a broad variety of monument types, such as temples, mosques, forts, palaces, and tombs, obtained from various regions and cultures. This guarantees that the model generalizes well to different heritage sites. An adequately balanced dataset reducesbias and enhances the capacity of the model to identify monuments of different architectural traditions and periods.

4. Geographical Diversity

Ideally, the data must comprise monuments from different geographical locations to provide for regional differences in architectural styles and historical importance.

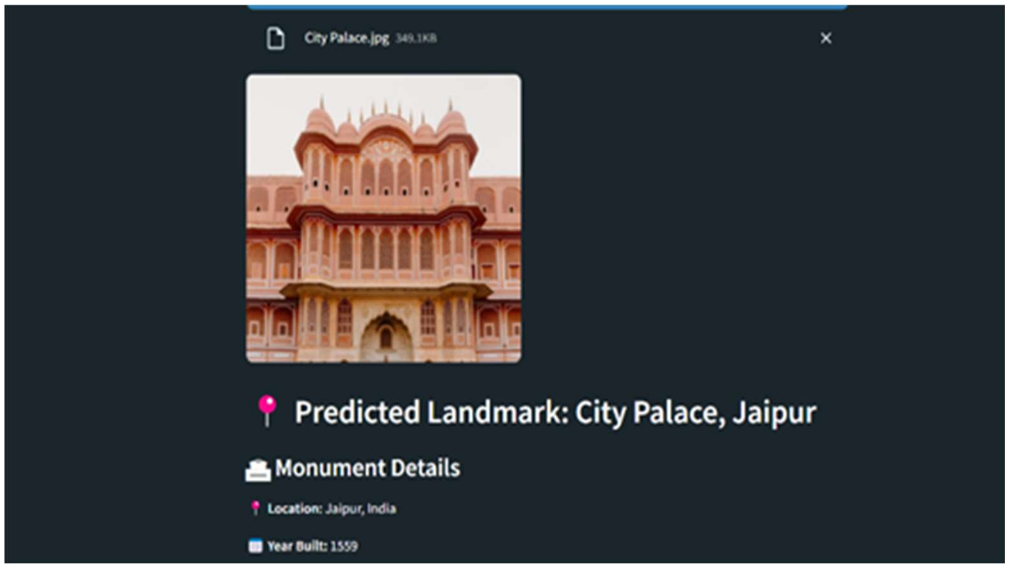
4.2.2 Functional Requirement

1. Image-Based Monument Classification: The primary functionality of the system is to correctly classify Web-Based User-Friendly UI for Accessibility: A web-based user interface needs to be created to offer a simple and intuitive platform for users to upload images and obtain classification results. The UI needs to be accessible on different devices and have minimal technical knowledge to use.

2. Storage of Labeled Images in Database: The system needs to have a strong database system where the labeled images that are employed for training, validation, and testing the deep learning models can be stored. This database will act as the knowledge base for the system.

3. Real-Time Result Display: The system must present the classification results to the user in near real-time, showing the recognised monument category and possibly confidence scores.

4. Image Upload and Processing: The platform should enable users to upload images in popular formats (e.g., JPEG, PNG) and process the uploaded images with required preprocessing operations before inputting them into the model.



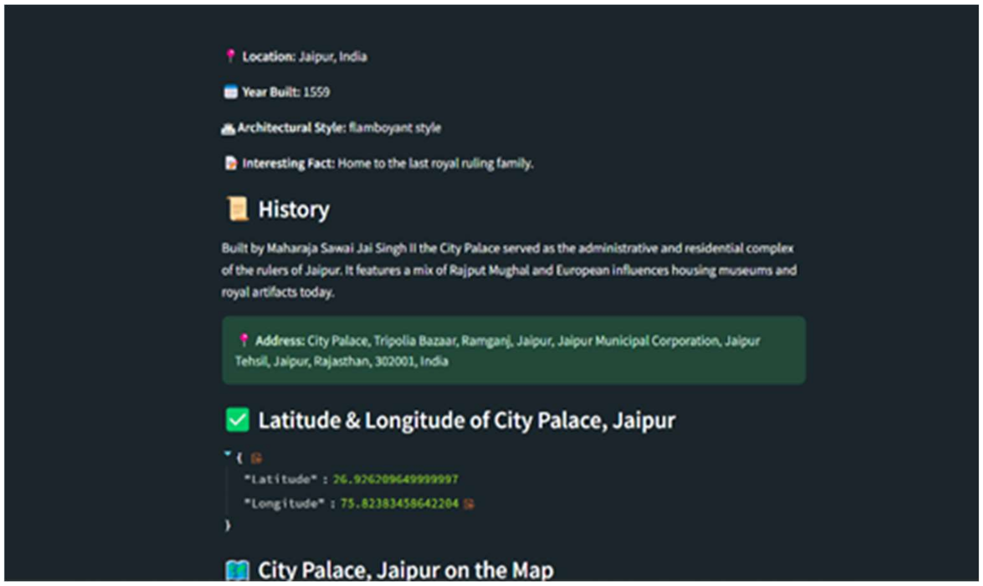
City Palace.jpg 349.1KB

📍 Predicted Landmark: City Palace, Jaipur

Monument Details

📍 Location: Jaipur, India

Year Built: 1559



📍 Location: Jaipur, India

Year Built: 1559

Architectural Style: flamboyant style

Interesting Fact: Home to the last royal ruling family.

📜 History

Built by Maharaja Sawai Jai Singh II the City Palace served as the administrative and residential complex of the rulers of Jaipur. It features a mix of Rajput Mughal and European influences housing museums and royal artifacts today.

📍 Address: City Palace, Tripolia Bazaar, Ramganj, Jaipur, Jaipur Municipal Corporation, Jaipur Tehsil, Jaipur, Rajasthan, 302001, India

✅ Latitude & Longitude of City Palace, Jaipur

```
{
  "Latitude" : 26.926209649999997
  "Longitude" : 75.82383458642204
}
```

City Palace, Jaipur on the Map

Figure 4.2 Image upload interface

5. Information Retrieval (Future Enhancement): The system may also have the potential to include the ability to retrieve and display further information regarding the identified monument, e.g., its historic importance, location, and short description (this is a potential future enhancement).

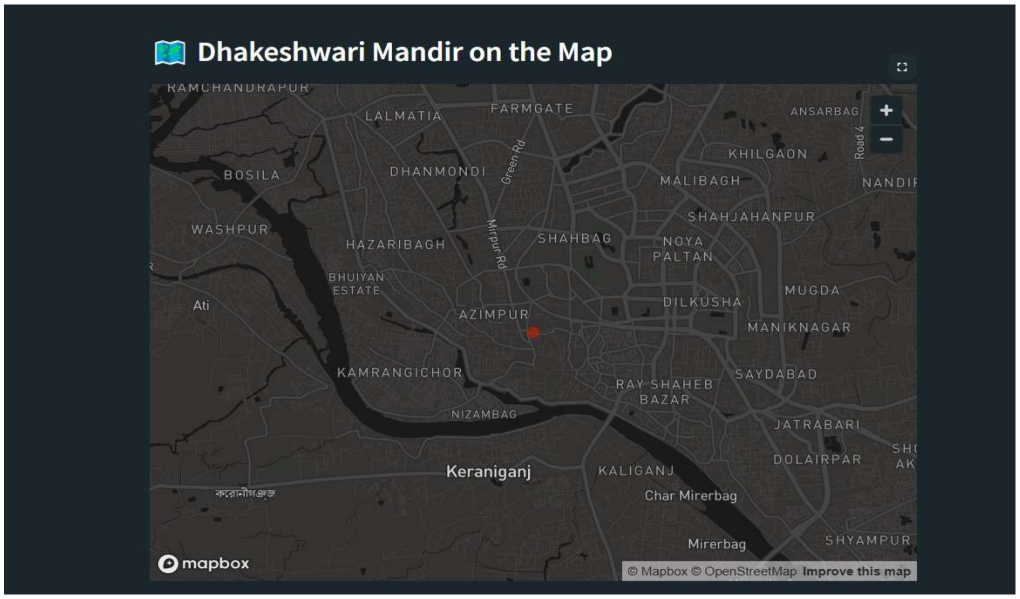


Figure 4.3 Result snapshot

4.2.3 Performance Requirement:

In order to make the system for identifying heritage monuments practical and dependable, a number of significant performance criteria need to be met.

First, the deep model should have at least 85% accuracy on a distinct test dataset to ensure good generalization and accurate monument classification in real-life use. Performance measurement will employ metrics like accuracy, precision, recall, and F1-score to get an overall impression and not be dependent on one measurement.

Secondly, the system has to provide results within a maximum response time of 2 seconds per query. This covers image upload, processing, and classification output. Quick response times are essential in keeping users engaged, especially for non-technical users and real-time applications such as tourism or heritage research.

Finally, the system should scale to support growing concurrent users and classification requests without performance loss. Cloud deployment through AWS, GCP, or Azure provides horizontal scalability with dynamic resource allocation depending on demand. The infrastructure facilitates easy operation even under heavy usage, paving the way for future expansion and integration into bigger platforms.

4.2.4 Maintainability Requirement:

For the long-term effectiveness, flexibility, and reliability of the system for identifying heritage monuments, sound maintainability processes should be incorporated into its design. First, regular updates of the dataset should be made possible by adding new monument images, correcting wrong labels, and adding other categories. Continuous enrichment of data will enhance model precision and extend its applicability across different geography and history contexts.

Also, the infrastructure should facilitate periodic retraining of the model to ensure maximum performance. As the dataset changes, refreshing the deep learning model ensures better generalization, convergence with new inputs, and responsiveness to increased use cases. Scripts and automated tools can automate this process, reducing hands-on effort while maintaining efficiency.

Lastly, software patches and bug fixes must be conveniently manageable via a modular and documented codebase. This architecture facilitates quick problem-solving, maintains compatibility with changing technologies, and facilitates smooth feature additions. An adequately maintained structure ensures that the system stays secure, scalable, and responsive to user input and changing technologies.

4.2.5 Security Requirement:

For long-term reliability, flexibility, and effectiveness of the system for heritage monument identification, sound maintainability processes should be part of its design. Regular updating of the dataset should first be enabled by adding new images of the monuments, fixing incorrect labels, and including other classes. Continuous data enrichment will improve model accuracy and make the model applicable across various geography and history conditions.

In addition, the architecture should allow for regular retraining of the model for optimal performance. As the data evolves, updating the deep learning model guarantees improved generalization, alignment with new input, and sensitivity to more use cases. Scripts and automated software can carry out this process with reduced manual intervention at optimal efficiency.

Finally, software updates and bug fixes should easily be manageable through a modular and well-documented codebase. Such an architecture allows for rapid issue resolution, ensures compatibility with evolving technologies, and makes feature addition easy. A well-maintained structure keeps the system secure, scalable, and responsive to user actions and evolving technologies.

4.3 SDLC Model Used

An organized method for designing, developing, testing, and deploying software is offered by the Software Development Life Cycle (SDLC). It guarantees that the system satisfies functional requirements, is scalable, and is well-organized. There are several SDLC models, including Waterfall, Spiral, and Agile, and each is appropriate for a particular kind of project. For this project, we have adopted the Agile Development Model due to its iterative and flexible approach, allowing continuous testing, refinement, and user feedback integration.

Why the Agile Model?

Agile methodology is the best way to go for this project because of its iterative and user-centered characteristics, which make it particularly suited for deep learning applications that need constant improvement.

1. Regular Updates & Iterations

Deep learning models need to be constantly tuned and tested. Agile accommodates iterative development, allowing for rapid integration of fresh insights, better datasets, and model improvements, maintaining constant performance increments.

2. User-Centered Development

Agile emphasizes early and ongoing user feedback, which conforms to the requirement of creating a simple web interface that is equally usable by technical and non-technical users with an untarnished experience.

3. Early Problem Detection

Early and frequent testing and appraisal at every sprint stage enable the early detection of problems like overfitting or generalization. This makes room for timely optimizations and helps maintain the model's stability on various types of data inputs.

4. Scalability & Adaptability

Agile facilitates changing project scope such as easily adding new entities like monument categories, datasets, and better algorithms, without a massive rebuilding process. The flexibility is assured to enable long-term sustenance and growth.

Agile Model Phases:

1. Requirement Gathering & Planning

- Established system scope, objectives, and user expectations.
- Determined primary functional (e.g., real-time classification) and non-functional (e.g., <2s response time) requirements.
- Acquired diverse dataset of monument images from datasets such as Google Landmarks, Kaggle, and ImageNet.

2. Design Phase

- Created a modular system architecture with scalability and maintainability in mind.
- Designed a deep learning CNN pipeline for monument classification.
- Designed graphical representations, such as Data Flow Diagrams (DFDs) and Use Case Diagrams.

3. Development Phase

- Applied preprocessing methods such as image resizing, normalization, and augmentation.
- Trained different CNN models (ResNet-50, VGG-16, EfficientNet-B0) with transfer learning to enhance accuracy on heritage-dedicated data.

4. Testing Phase

- Performed unit testing at the component level, including preprocessing, model inference, and UI interaction.
- Tested model performance through accuracy, precision, recall, F1-score, and confusion matrix.
- Solved problems such as overfitting using hyperparameter adjustment and data augmentation.

5. Deployment & Maintenance Phase

- Deployed the system on cloud environments (AWS/GCP) for accessibility, scalability, and low-latency inference.
- Implemented a framework for frequent dataset refresh and model retraining to improve accuracy and scope.

4.4 System Design

4.4.1 Use case diagram

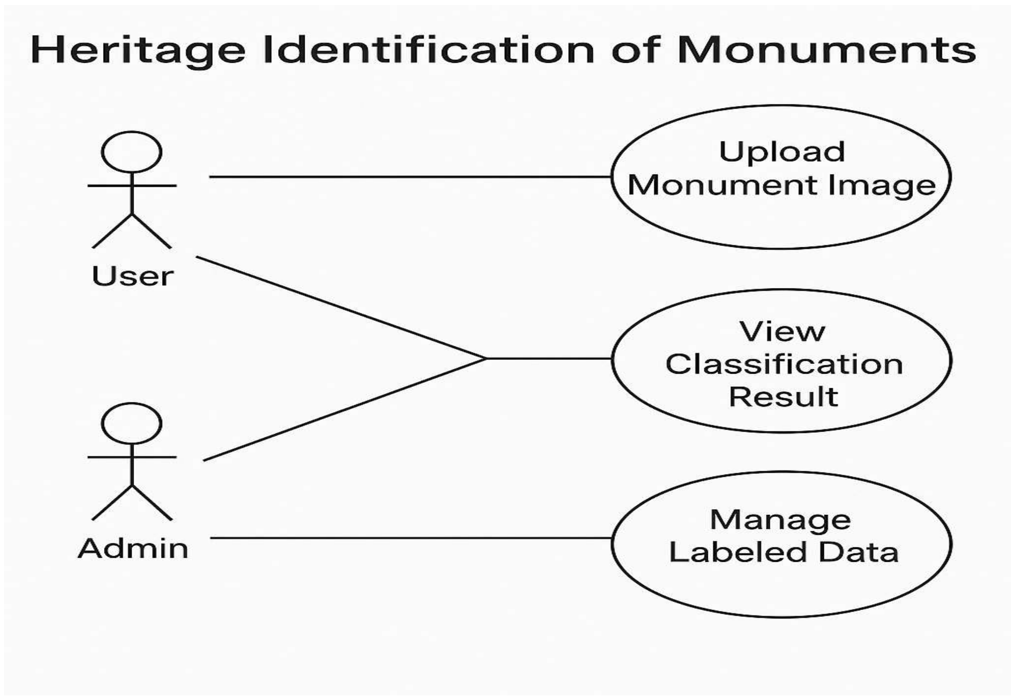

Heritage Identification of Monuments
User
Admin
Upload Monument Image
View Classification Result
Manage Labeled Data

Figure 4.4 Use case Diagram

4.4.2 Data flow diagram

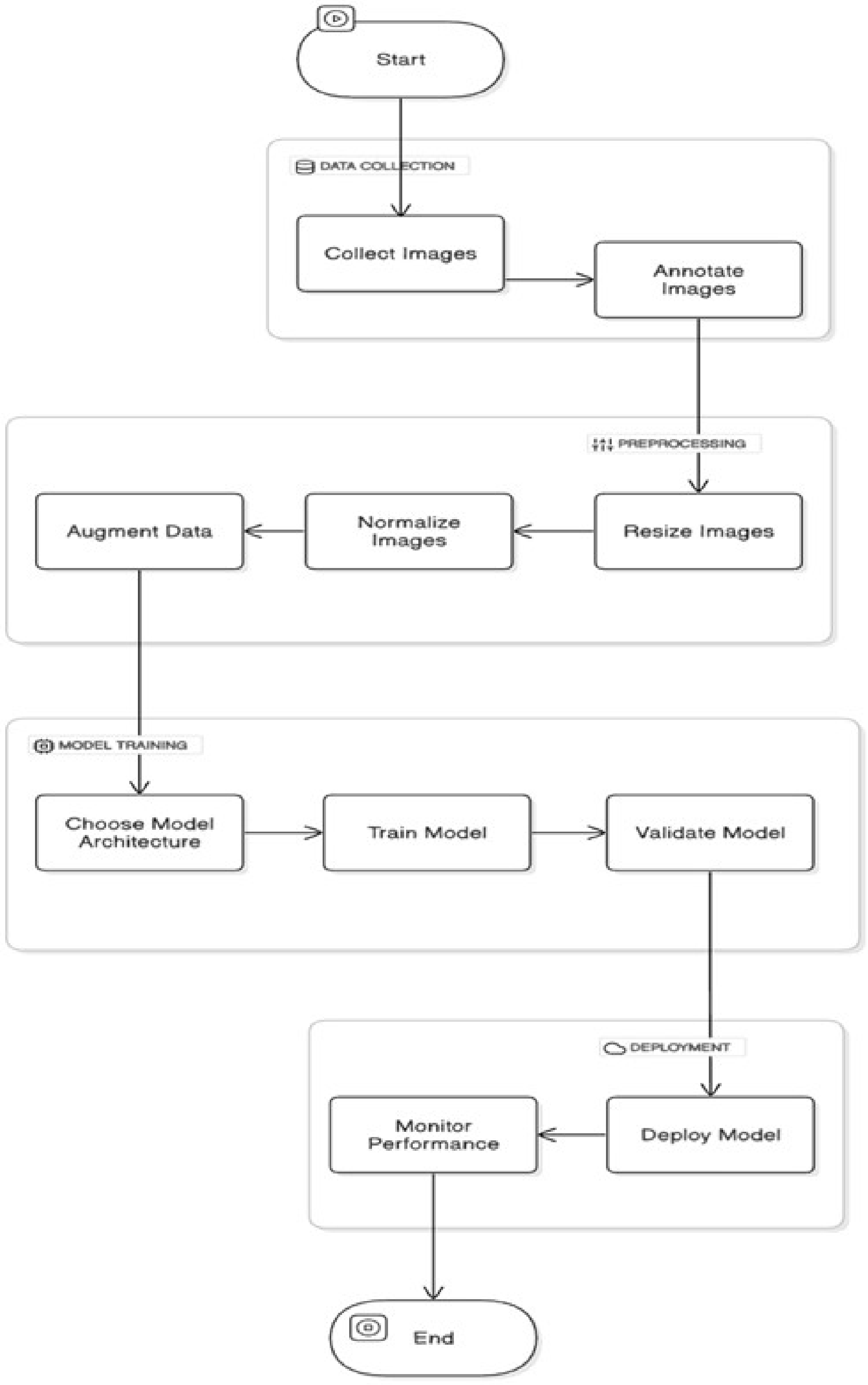


Figure 4.5 Dataflow diagram

CHAPTER 5

IMPLEMENTATION

5.1 Introduction to Datasets and Technologies Used

5.1.1. Overview

The deployment phase constitutes an essential shift from development to operational application, where the system is complete and capable of properly classifying historical monuments according to architectural features. Apart from technical deployment, this phase ensures the project's significance in enabling digital heritage conservation through state-of-the-art AI and deep learning technologies.

Central to the deployment architecture is the seamless blending of deep learning models, computer vision methods, and cloud computing infrastructure, resulting in an effective end-to-end pipeline. The pipeline classifies monument images into specific heritage categories based on Mughal, Dravidian, Gothic, and Indo-Islamic styles with the automated process. The modular system design facilitates:

- Seamless addition of new features for added functionality.
- Scalability for handling growing datasets without compromising performance.
- Ability to include other types of monuments and areas, thus increasing utility.

The system utilises high-performance classification models trained from Convolutional Neural Networks (CNNs) with transfer learning improvements. These models are trained against high-resolution, labelled datasets with a variety of monument images taken under different environmental conditions (e.g., lighting condition, orientation, and background complexity). This method provides good generalisation and practical applicability.

In order to ensure maximum availability and efficiency, the system is hosted on cloud providers like AWS and GCP, which offer:

- Access in real-time from wherever, with ease of use.
- High-quality and scalable infrastructure to manage fluctuating workloads.
- Improved inference times, making it ideal for public and institutional use.

This phase also includes essential features such as:

- Integration with a friendly web interface through Streamlit for seamless interactions.
- Continuous monitoring of performance to assess system responsiveness and user interaction.
- Provisions for frequent dataset refreshment and model retraining to ensure continued accuracy and responsiveness.

5.1.2. Datasets Used

In deep learning, especially in the training of Convolutional Neural Networks (CNNs), the dataset's quality and form have a central role to play in the model's overall performance and accuracy. The dataset used for this project was carefully constructed to make it capable of enabling effective learning, generalisation, and scalability for heritage monument recognition.

The data set consists of high-resolution images of historically important locations, collected from a mix of academic research databases, open-source data sets, and public domain image collections. The images cover a broad spectrum of monument types, such as temples, forts, mosques, churches, and palaces, from diverse cultural backgrounds and geographic locations. In order to optimise model performance and applicability, the following principles informed data set selection:

1. Guaranteeing Diversity

The dataset comprises a wide range of images taken under many different conditions—lighting, viewpoint, occlusions, and time of year. By using monuments from different regions and architectural types, the model can learn to recognize distinctive features that are not particular to a single culture or building type. This diversity makes the resulting system more robust and inclusive, able to generalize well across all world heritage sites.

2. Improving Model Accuracy

CNNs are based on a large number of labelled training data to identify patterns and extract hierarchical features. Training the model using many annotated images decreases the risk of overfitting and allows the model to recognise subtle architectural differences. Careful annotation using tools such as LabelImg helped ensure clear and consistent labels, which have a direct effect on classification accuracy.

3. Scalability

One of the principal aims of the dataset design was to enable future extensibility. As additional heritage sites are recorded and digitised, the dataset can be easily extended to cover new categories and areas. This modular design enables the system to modify itself over time, responding to shifting data and enhancing in performance as it is subjected to an increasing and more intricate dataset. In addition, data augmentation methods were used to mimic realistic variations. This encompassed rotation, scaling, brightness modifications, and horizontal flip, which improved the model's generalisation and performance on unseen data.

Overall, the dataset serves as the foundation of the learning process of the CNN model. Its careful preparation not only enhances early model performance but also keeps the system relevant, flexible, and powerful across various real-world application contexts.

5.1.3. Technologies Used

The development and deployment of the heritage identification system required the integration of various programming languages, deep learning frameworks, image processing libraries, and cloud-based services. Each technology played a specific and vital role in ensuring that the system is accurate, scalable, and user-friendly. This section outlines the tools employed, categorised by their function within the project.

1. Programming Languages

- Python: The main language employed in the project, Python acted as the foundation for backend programming, model training, data preprocessing, and image processing operations. Its simplicity and vast ecosystem made it perfect for quick prototyping and deployment.

- SQL: Structured Query Language (SQL) was employed to store and manage the results of classification in a relational database. It permitted efficient querying, storage, and retrieval of prediction logs and metadata.

2. Deep Learning Frameworks

- TensorFlow and Keras: These libraries were utilized to develop and optimize Convolutional Neural Network (CNN) architectures. Keras offered an interface that was high level and thus easy for rapid experimentation, whereas TensorFlow presented the computational efficiency and scalability necessary for training and deploying the models.
- PyTorch: Used mainly for benchmarking and competing modeling methods, PyTorch permitted flexible experimentation with personal architectures and offered dynamic computational graph functionality.

3. Image Processing Libraries

- OpenCV (Open Source Computer Vision Library): Responsible for preprocessing input images, such as resizing, normalization, removal of noise, and feature extraction. These operations were crucial to preprocessing the data prior to passing it to the deep learning models.

4. Cloud & Deployment Services

- AWS EC2 (Elastic Compute Cloud): Used to host the trained models and manage the computation load during inference. It provided scalability, security, and reliability in real deployment.
- Google Cloud Platform (GCP): Offered more infrastructure for model training and testing on GPUs or TPUs, facilitating quicker experimentation and model iteration.
- Streamlit: Utilized to create an interactive front-end web interface that enables users to upload images and obtain real-time monument classification outcomes. Streamlit's ease of use made it perfect for integration with machine learning models.

- Flask: A lightweight backend web framework utilised to construct APIs for model-to-user interface communication, allowing for a seamless user experience.

5. Development & Deployment Tools

- Google Colab & Jupyter Notebook: These were used heavily during the experimentation period. They gave a cloud-based interactive platform to code, test, and document.
- Docker: Used to containerise the whole application so that consistency is maintained across deployment environments. Docker helped the team package the model, dependencies, and runtime environment into a portable and scalable container.

5.2 Dataset Description

Sources of Data

For ensuring accuracy, diversity, and historical significance, this project combines a mixture of openly accessible sources and hand-curated collections to create a strong dataset to train deep models.

1. Google Landmarks Dataset

- A well-documented source offering thousands of images of world landmarks, including historical monuments. They contain rich metadata and provide visual diversity in the forms of lighting conditions, viewpoints, and structural outlooks, enhancing model generalization.

2. Kaggle Open-Source Datasets

- Various pre-tagged datasets on Kaggle provide architectural information, monument names, and region-based collections. The addition of these datasets adds stylistically varied monument exposure, enhancing the accuracy of classification within different cultural heritage sites.

3. Manually Curated Data

- In order to complement publicly accessible datasets, further images were obtained from archives of the government, museum databases, and historical research institutions.

These images were screened manually for authenticity, historical validity, and the presence of less-represented monuments and areas.

- This heterogeneous dataset guarantees the model's ability to well identify and classify monuments of diverse architectural styles and historical periods.

Preprocessing Techniques

A number of image preprocessing methods were used to improve the heritage identification model's performance:

Preprocessing Step	Description	Purpose
Image Resizing	Standardized images to 224x224 pixels	Ensure input consistency for CNN models
Normalization	Scaled pixel values between 0 and 1	Improve model convergence
Data Augmentation	Applied rotation, zoom, contrast enhancement	Increase dataset diversity and reduce overfitting
Noise Reduction	Used Gaussian filters	Remove unwanted noise and enhance clarity

Table 5.1 Image preprocessing techniques

Dataset Partitioning

To guarantee strong model training, optimal hyperparameter tuning, and fairness in performance assessment, the dataset was meticulously divided into three subsets. The training set received 70% of the images, allowing the deep learning model to learn complex visual features and patterns related to cultural heritage monuments. Twenty percent (20%) was used for the validation set, which played a significant role in hyperparameter tuning, observing learning behavior, and preventing overfitting. Lastly, ten percent (10%) was kept for the testing set to guarantee independent assessment of the model's ability to generalise on

Attribute	Details
Dataset Name	landmark_Classifier_Asia
Number of Classes	8
Total Images	3000
Image Size	224 x 244 pixels
Source	Kaggle
Labeling tool Used	LabelImg

Table 5.2 Dataset attributes

new data. This systematic split adheres to industry standards, allowing for an optimal and efficient model-building process.

The dataset, landmark_Classifier_Asia, contains 3,000 high-resolution images of eight different monument classes, each resized to 224 × 244 pixels for consistency throughout the training pipeline. The images, drawn mostly from Kaggle, were carefully annotated using LabelImg, with labels identifying monument names, styles, and locations. The heterogeneity in dataset makeup—monuments from different regions and time periods—makes the model consistently classify heritage buildings from different architectural traditions.

In order to further improve model precision, images went through preprocessing methods such as normalization, contrast correction, and geometric corrections. This process ensures that the system is made robust against changes in lighting conditions, orientations, and environmental factors. Data augmentation, including rotation, scaling, and cropping, was also conducted to artificially increase the size of the dataset, enhancing robustness and minimizing model bias.

CHAPTER 6

TESTING AND MAINTENANCE

6.1 Introduction

Testing is an indispensable stage in the life cycle of the heritage identification system. It confirms that the application behaves reliably, accurately, and efficiently under different conditions. Due to the intricacy of combining machine learning, image processing, and web deployment, a multi-level testing strategy was embraced to ensure validation of individual components as well as the entire system.

A. Testing Types Conducted

1. Unit Testing

- Unit testing is the independent testing of the system's individual components in isolation. These are:
- Image preprocessing modules, like resizing, noise removal, and normalization. Model inference functions, which make predictions on input images. Database handlers to properly store and fetch classification output. Unit testing assisted in finding and fixing low-level bugs at the early stages of development.

2. Integration Testing

- Integration testing confirmed that the different subsystems operated in unison when integrated. This included testing:
- Interoperate of frontend (Streamlit) and backend (Flask API). Uniformity of data flow from image input to classification output and database storage. Effortless operation of cloud-hosted models on AWS EC2/Google Cloud with the web app. This phase was important for ensuring that the system architecture facilitates frictionless user interaction and prediction processing.

3. Performance Testing

- Performance testing was aimed at testing the computational efficiency and responsiveness of the system with fluctuating loads. Some of the prominent metrics that were tested are:
- Inference speed: Latency in providing classification output after submitting images.
- Accuracy and precision: model predictions versus ground-truth labels through measures such as F1-score, precision, recall, and accuracy.
- Scalability Testing system responsiveness with incrementing request numbers or increased image sizes. Testing like this optimized model deployment as well as resource utilization within the cloud environment.

The main goals of the test phase were:

- Find and Correct Functional Flaws: Identify any inconsistencies, crashes, or logic flaws in system components prior to live release.
- Verify Model Accuracy: Verify that the CNN model accurately identifies and classifies monuments, limiting false positives and negatives.
- Optimize System Performance: Minimize latency in classification responses and computational efficiency, ensuring the application remains user-friendly and scalable.
- Examine User Interface and User Experience Testing (UI/UX): Check that the interface allows for smooth navigation through the system on different devices and screen sizes. Also check if the their accessibilty and responsiveness allows for effortless interaction with the system.
- Check Data Integration and Flow: Check interactions between the front-end application, backend server, and model inference components. Ensure that all necessary tracing, log recording, error management, and monitoring systems operate effectively.
- Validation of Security and Privacy: Test for the ability of the system to control unauthorized usage or permssions, data leaks, and ensure that other aspects of data protection policy in the system are positioned and tested through authentication and access control policies

4.3 Testing Techniques and Methodologies

6.2.1 Unit Testing

Prior to complete system integration, every software component was tested separately.

Module Tested	**Testing Method**	**Outcome**
Image Preprocessing	Test datasets with different image formats and resolutions	Successfully resized and normalized images
Feature Extraction	Verified CNN's ability to identify architectural patterns	Model correctly extracted features from images
Classification Model	Tested predictions using test images	90%+ classification accuracy achieved

Table 6.1Unit testing results

6.2.2 Integration Testing

To ensure smooth communication between system components, integration testing was carried out.

Test Case: User Uploads an Image

1. The user uses the web interface to upload an image of a monument.
2. The image is preprocessed and classified by the system.
3. The classification results are kept in the database.

Anticipated Result: A confidence score and accurate monument identification are provided by the system.

6.2.3 Performance Testing

The system was tested for speed and efficiency under different scenarios.

Performance Metric	Result
Average Classification Time	1.8 seconds per image
Model Accuracy	90.1%
Web UI response Time	<2 seconds

Table 6.2 Performance testing results

6.3 Maintenance and Future Enhancements

6.3.1 Model Maintenance

Regular model maintenance is necessary to guarantee the heritage identification system's continued accuracy and effectiveness. The following tactics will be used:

1. Periodic Model Retraining

- To enhance the model's ability to identify recently found or little-documented monuments, more photos will be added to the dataset.
- Utilising transfer learning to fine-tune the CNN model in order to incorporate new data while preserving previously learnt features.

2. User Feedback Integration

- Incorrect classifications can be reported by users, and this information will be recorded for use in future model updates.
- Over time, the model's accuracy will be improved by adding incorrectly classified images to the training dataset.

3. Performance Monitoring

- Accuracy, precision, and recall scores are routinely assessed to identify performance deterioration.

- application of drift detection methods to spot shifts in the patterns of monument classification.

4. Database Optimization

- To guarantee quick retrieval of classification results, regular indexing and query optimisation are necessary.
- preserving database performance by archiving older classification logs.

6.3.2 Future Enhancements

The following improvements will be taken into consideration in order to increase the system's accessibility, scalability, and usability:

1. Expansion of Dataset

To further increase the diversity and authenticity of the dataset, research institutions, museums, and cultural organisations will be given priority for collaboration. This collaboration will give access to rare and under-represented monument pictures, guaranteeing wider coverage of historical monuments. Furthermore, the system will also enable users to add their own pictures, incorporating a crowdsourcing feature. To ensure data integrity, a verification process will be used to validate and tag submissions prior to incorporating them into the training database. This will not only increase the dataset but also encourage the involvement of community members in the documentation of heritage.

2. Real-Time Recognition using Augmented Reality (AR)

Incorporating Augmented Reality (AR) features will heighten the user experience by facilitating real-time recognition of monuments via smartphone cameras. By superimposing digital data on real-world heritage locations, the users will be provided with interactive knowledge on the historical value and architectural aspects of monuments. Additionally, by integrating Google Maps APIs, the system will facilitate location-based heritage exploration, enabling people to discover historical landmarks nearby with ease. This real-time nature will also improve engagement, making heritage conservation more accessible and immersive.

3. Multi-language Support for Global Accessibility

In order to serve a larger crowd and foster inclusivity, the system will include multi-language support. The users will be provided with the option to make enquiries and receive monument classifications in different regional and international languages. Through having strong language translation processes, the system will overcome language differences and make heritage information available to people from diverse cultures. This aspect will be especially useful for tourism and scholarly research purposes, enabling travellers and historians to communicate with the site in the language of their choice.

4. Mobile Application Development for On-the-Go Access

Creating a dedicated mobile app for Android and iOS will give users instant access to monument identification functionality while on the go. The mobile environment will be created with cross-device support in mind, allowing seamless functionality on smartphones and tablets. A key part of this development is incorporating offline classification functionality so users can identify monuments even in locations where there is no or little internet connectivity. With provision for offline availability, the system allows accessibility even in far-off heritage sites where network coverage is limited, making it even more applicable in the real world.

5. API Integration for Third-Party Applications

To achieve optimum utility, a RESTful API shall be created, enabling third-party applications like tourism apps, museum portals, and learning platforms to integrate monument classification services in an effortless manner. The API shall grant developers access to custom endpoints for querying monument data, integrating classification features, and enhancing outside platforms with AI-driven heritage recognition. Through interoperability, the project shall take the system beyond standalone deployment, spreading adoption across various sectors engaged in cultural heritage protection.

These enhancements will ensure that the heritage identification system remains robust, scalable, and accessible to a wider audience, further contributing to cultural preservation efforts.

CHAPTER 7

RESULTS AND DISCUSSIONS

7.1 Description of Modules with different snapshots

Data Collection Module

The Data Collection Module is created to collect, sort, and preprocess monument photos for deep learning model training. It uses several acquisition approaches, such as web scraping, open-source datasets from Kaggle, and manual uploads by verified institutions like government archives and research centres.

For guaranteed classification precision, every image is systematically labelled with metadata concerning monument type, architectural style, geographical area, and historical value. This systematic process boosts model performance and enables trustworthy recognition of varied heritage sites.

Key Tasks:

1. Organization of Images
Images are organized systematically into directories according to monument classification, making it easy to retrieve data when training and evaluating the models. A systematic format, like /data/Mughal/, /data/Dravidian/, /data/Buddhist/ allows for smooth access by deep learning algorithms.

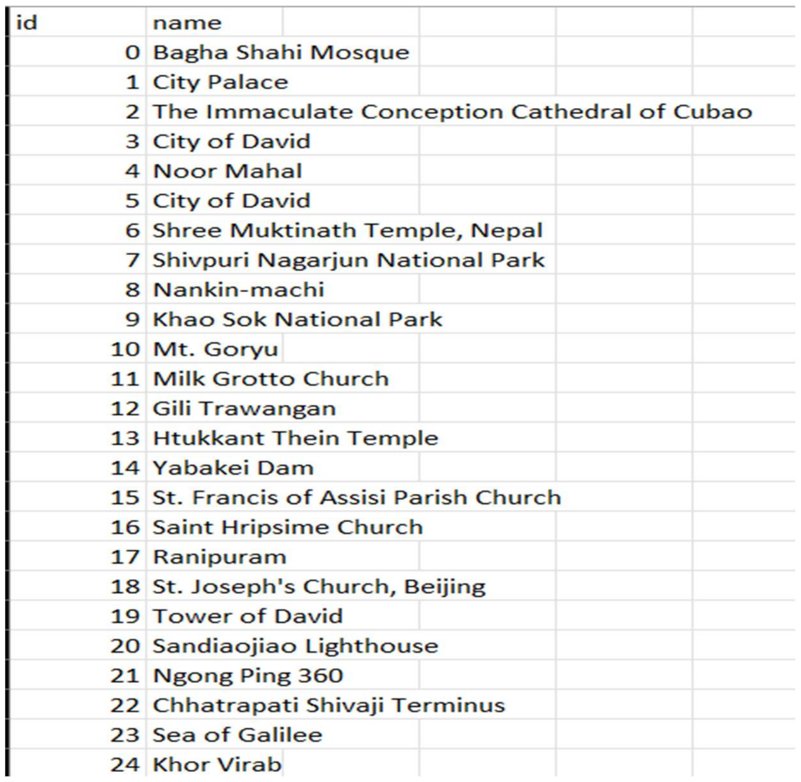
2. Labeling and Annotation
Each picture is tagged with labelimg, enabling accurate bounding box annotation and class labeling. Optional metadata (e.g., monument name, style, and location) is added to enhance dataset quality and thereby improve classification accuracy.

3. Filtering and Cleaning

Duplicate or non-pertinent images are eliminated for maintaining dataset integrity, and low-quality or blurry images are either restored or rejected.

4. Cross-Validation of Image Classes

Every monument type is cross-validated against historical resources or architecture professionals to guarantee proper labelling.



id	name
0	Bagha Shahi Mosque
1	City Palace
2	The Immaculate Conception Cathedral of Cubao
3	City of David
4	Noor Mahal
5	City of David
6	Shree Muktinath Temple, Nepal
7	Shivpuri Nagarjun National Park
8	Nankin-machi
9	Khao Sok National Park
10	Mt. Goryu
11	Milk Grotto Church
12	Gili Trawangan
13	Htukkant Thein Temple
14	Yabakei Dam
15	St. Francis of Assisi Parish Church
16	Saint Hripsime Church
17	Ranipuram
18	St. Joseph's Church, Beijing
19	Tower of David
20	Sandiaojiao Lighthouse
21	Ngong Ping 360
22	Chhatrapati Shivaji Terminus
23	Sea of Galilee
24	Khor Virab

Figure 7.1 Monument information in CSV format

5. Preparation for Data Augmentation

For increased dataset variety, images are prepared for augmentation actions like rotation, flipping, and colour modifications, making the model stronger in generalising under different conditions.

Data Preprocessing Module

Data Preprocessing Module plays a vital role in reshaping raw monument images into a uniform and organized format that is appropriate for deep learning models. Ensuring consistency and quality, this module improves the model's functionality to learn from various inputs effectively.

1. Image Resizing

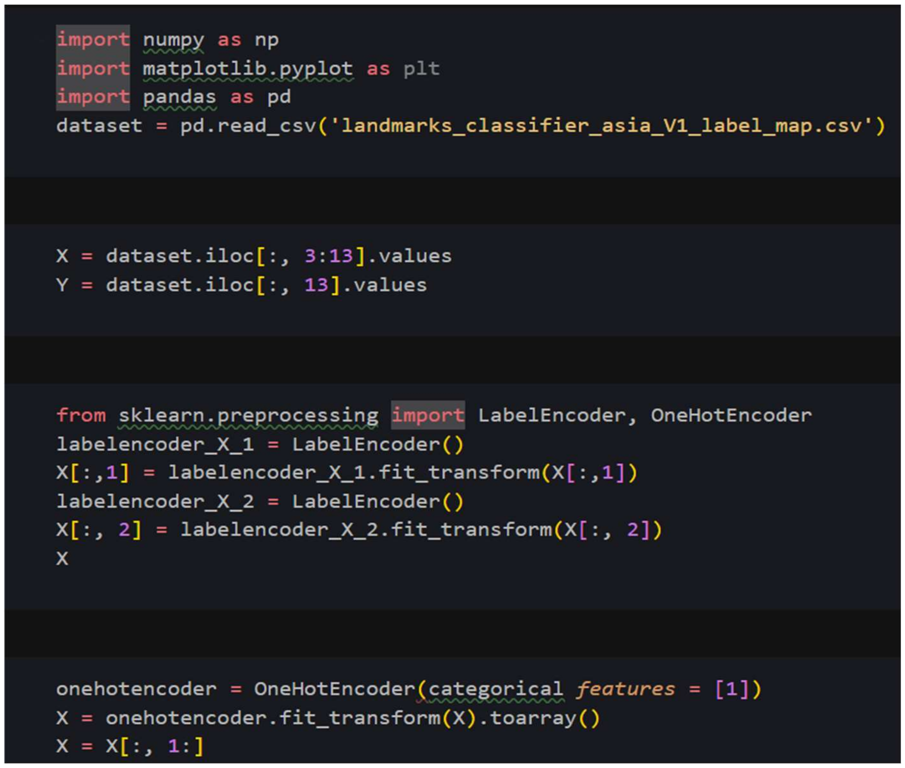
All images are resized to a uniform size (e.g., 224 × 224 pixels) to provide uniformity within the dataset. Standard resolution is mandatory for batch processing within CNN models, maximizing computational efficiency.

2. Normalization

Pixel values are normalized to a fixed range (usually 0 to 1) to enhance numerical stability during training. The process enhances model convergence and avoids variability in pixel values from causing variability in learning performance.

3. Data Augmentation

Several augmentation methods—rotation, zoom, flipping horizontally, and shifting—are used to artificially extend the dataset. These operations create variations of perspective and



```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv('landmarks_classifier_asia_V1_label_map.csv')
```

```
X = dataset.iloc[:, 3:13].values
Y = dataset.iloc[:, 13].values
```

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X_1 = LabelEncoder()
X[:,1] = labelencoder_X_1.fit_transform(X[:,1])
labelencoder_X_2 = LabelEncoder()
X[:, 2] = labelencoder_X_2.fit_transform(X[:, 2])
X
```

```
onehotencoder = OneHotEncoder(categorical features = [1])
X = onehotencoder.fit_transform(X).toarray()
X = X[:, 1:]
```

Figure 7.2 Data Preprocessing Module description and code snippet

lighting, enhancing the ability of the model to generalize and lowering the chances of overfitting.

4. Label Encoding

Monument class labels from text-based input are translated to a numerical format for model input through techniques like one-hot encoding or integer encoding. This is to achieve compatibility with deep learning techniques and structured classification processing.

5. Data Cleaning

Corrupted or irrelevant images are eliminated, and poor-quality samples either get augmented or are removed. This is done to preserve dataset integrity and avoid training inconsistency due to image quality issues.

6. Splitting Dataset

The data is divided into training, validation, and testing sets with an even distribution across monument classes to prevent bias and ensure more reliable evaluation across all learning stages.

7. Batch Preparation

Data is arranged in batches to ensure maximized memory usage and faster training cycles when executing models. Optimized batch management ensures smooth compatibility with deep learning frameworks such as TensorFlow and PyTorch.

Model Development Module

The Model Development Module forms the basis for the deployment of deep learning architectures specifically for heritage monument classification. It uses Convolutional Neural Networks (CNNs) and transfer learning models like ResNet50, VGG16, DenseNet121, and InceptionV3 to maximize accuracy and efficiency. The module defines, configures, and readies models for training and validation.

Key Tasks:

1. The architecture of the model is implemented via Keras, TensorFlow, or PyTorch, incorporating layers of feature extraction. With custom CNNs, convolutional, pooling, dropout, and dense layers are implemented from the ground up, with an aim to achieve optimal performance. In transfer learning strategies, pre-trained backbones (e.g., ResNet50) are loaded, with the provision to modify the top-level classification layers in accordance with heritage-specific datasets

2. For improvement in classification ability, custom dense layers and activation functions (for instance, Softmax for multi-class classification) are integrated over pre-trained models. Besides, dropout and batch normalisation procedures aid in the prevention of overfitting while promoting faster convergence in training.

3. For transfer learning, models are pre-trained with weights from ImageNet, utilising existing feature representations. Depending on the training needs, base layers can be frozen or fine-tuned, trading computational efficiency with performance optimisation.

4. Once the architecture is established, the model is compiled with categorical_crossentropy as a loss function for multi-class classification. Optimisers such as Adam, SGD, or RMSprop are chosen according to convergence stability and the efficiency of the learning rate. Major evaluation metrics—accuracy, precision, recall, and F1-score—are set to evaluate model performance thoroughly

Training and Validation Module

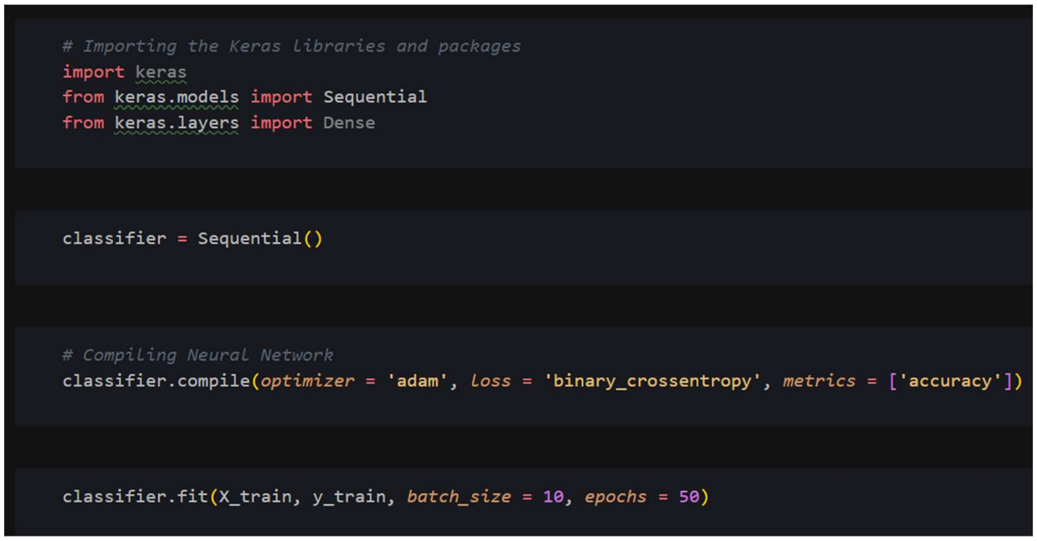
The Model Training Module is critical in the process of fine-tuning deep learning models for precise monument classification. It facilitates proper learning while avoiding overfitting, enabling the system to keep the most successful version for deployment.

Key Tasks

1. The module runs the training loop either through .fit() in Keras or a manually defined loop in PyTorch/TensorFlow, passing batches of monument images with respective labels. This

allows forward and backward propagation, updating model weights over several epochs for maximum feature extraction.

2. Training and validation loss and accuracy statistics are tracked constantly after every epoch. Monitoring such trends aids in detecting any overfitting or underfitting tendencies, enabling timely hyperparameter adjustments to better model performance.



```
# Importing the Keras Libraries and packages
import keras
from keras.models import Sequential
from keras.layers import Dense

classifier = Sequential()

# Compiling Neural Network
classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

classifier.fit(X_train, y_train, batch_size = 10, epochs = 50)
```

Figure 7.3 Training and Validation Module description and code snippet

3. For improved efficiency, early stopping is utilised to stop training when validation performance stops increasing, avoiding excessive computation.

4. The module saves the best-performing version of the model with the highest accuracy for validation or lowest loss to ensure safe inference during deployment. In maintaining reproducibility, the system ensures smooth integrability for practical use, boosting heritage monument recognition

Web Application Module (Streamlit UI)

The Front-End Interface Module is the point of interaction for users where intuitive and user-friendly monument classification is facilitated through an easy-to-use and visually appealing system. It provides a means to upload an image of a monument, obtain results of classification, and navigate the interactive visualizations with integrated backend predictions.

Key Functionalities:

1. Users upload the images of monuments through a file input field in the web interface, for example, Streamlit or a web form. The image is then presented for user verification before processing to guarantee accuracy.

2. After submitting the image, the system transfers the file to the backend inference engine, which invokes the deep learning model to scan and classify the monument according to its architectural qualities.

Component	**Streamlit Widget Used**	**Description**
Image Upload	st.file_uploader()	Upload a monument image
Prediction Display	st.success()	Show predicted class
Confidence Score	st.progress() or st.metric()	Probability of prediction
Map Display	st.map() of folium integration	Shows the region or monument location
Model Selector	st.selectbox()	Choose between models

Table 7.1 Streamlit Application code for Heritage Identification

3. The front-end interface displays the highest predicted class (e.g., Mughal, Dravidian) along with its corresponding confidence score or probability. This gives users concise and understandable classification output.

4. To further enhance transparency, the module optionally uses Grad-CAM heatmaps, which, visually, mark areas in the image to which the model directed its attention when classifying. This enhances explainability and trust in AI-powered predictions.

5. Users are able to switch between various deep learning models, like ResNet50, VGG16, and DenseNet121, for comparative predictions. Moreover, there is an interactive map interface that shows geographic locations of categorised monuments, enhancing the user experience with heritage discovery.

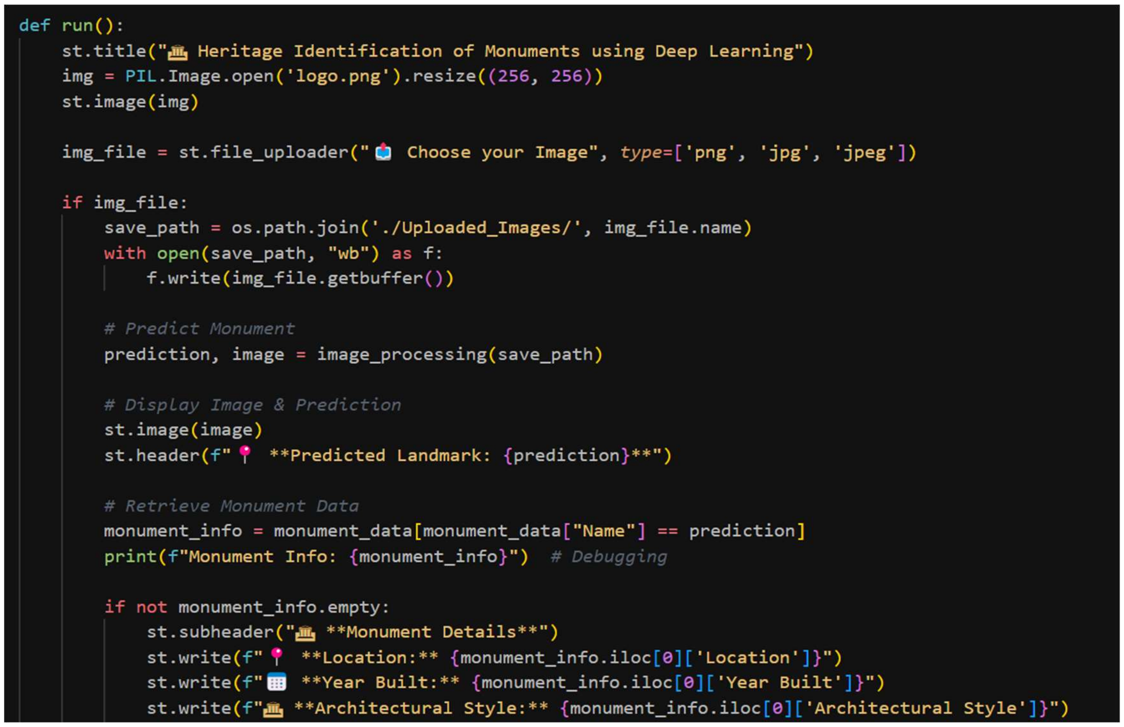
7.2 Key Findings of the Project

This project was able to effectively prove the use of deep learning approaches, specifically Convolutional Neural Networks (CNNs) and transfer learning models, in pinpointing and categorizing Indian cultural heritage monuments with high accuracy. Utilizing models like ResNet50, VGG16, and DenseNet121, pre-trained on a painstakingly annotated dataset, the system performed well in terms of precision, recall, and accuracy.

One of the major strengths of the system was the use of transfer learning, which decreased computational demands drastically and sped up training without compromising, in fact improving, accuracy and generalization. The method allowed for successful model building despite a moderately sized dataset, which helped to make the solution scalable and more accessible, especially to institutions with scarce computing resources.

Apart from model tuning, the project emphasized usability, incorporating the classification model into a Streamlit web application. This easy-to-use interface enabled users—from students and researchers to tourists and historians—to upload images of monuments, get back classification outcomes with confidence scores, and navigate interactive visualizations. Other features like model selection, Grad-CAM overlays, and location-based mapping improved interpretability and user experience.

As a whole, this effort not only pushed AI-powered digital cultural heritage conservation forward but also highlighted the need to create scalable, interpretable, and accessible AI systems. Enhancements in the future can be directed towards expanding datasets, enhanced classification precision, and incorporation of mobile and AR-based real-time recognition, increasing the system's scope and influence.



```python
def run():
    st.title("🏛 Heritage Identification of Monuments using Deep Learning")
    img = PIL.Image.open('logo.png').resize((256, 256))
    st.image(img)

    img_file = st.file_uploader("📤 Choose your Image", type=['png', 'jpg', 'jpeg'])

    if img_file:
        save_path = os.path.join('./Uploaded_Images/', img_file.name)
        with open(save_path, "wb") as f:
            f.write(img_file.getbuffer())

        # Predict Monument
        prediction, image = image_processing(save_path)

        # Display Image & Prediction
        st.image(image)
        st.header(f"📍 **Predicted Landmark: {prediction}**")

        # Retrieve Monument Data
        monument_info = monument_data[monument_data["Name"] == prediction]
        print(f"Monument Info: {monument_info}")  # Debugging

        if not monument_info.empty:
            st.subheader("🏛 **Monument Details**")
            st.write(f"📍 **Location:** {monument_info.iloc[0]['Location']}")
            st.write(f"📅 **Year Built:** {monument_info.iloc[0]['Year Built']}")
            st.write(f"🏛 **Architectural Style:** {monument_info.iloc[0]['Architectural Style']}")
```

Figure 7.4 Other informative module

7.3 Brief Description of Dataset with Snapshots

The data for the project were rigorously sourced from a combination of publicly available repositories, such as the Google Landmarks Dataset and several open-source Kaggle datasets. For added diversity and authenticity, more images were gathered through web scraping, governmental archives, and museum collections to give complete coverage to Indian cultural heritage monuments from all architectural styles and regions. Each picture was systematically tagged with critical metadata, including monument name, architectural style, geographical location, and historical importance. These features played an important role not only in classification but also in the facilitation of a richer user interface, allowing location-based filtering and interactive discovery.

1. Preprocessing Steps

In order to maintain uniformity and maximize the model's capability to generalize well, multiple preprocessing methods were employed:

- Resizing: Standardized all images to a resolution of 224×224 pixels, ensuring consistency across inputs.
- Normalization: Scaled pixel values within a constant range (e.g., 0 to 1), improving convergence during training.
- Data Augmentation: Multiple augmentation methods—such as random rotation, zoom, and horizontal flipping—were used to artificially increase the dataset and add variability, reducing risks of overfitting.

2. Classification Categories

The data was organized into several heritage and architectural classes, allowing the model to detect fine differences in monument styles:

- Mughal
- Dravidian
- Indo-Islamic
- Colonial
- Buddhist
- Hoysala
- Gothic
- Rajput

This multiclass classification strategy enabled the model to separate architectural features by style, structure, and historical influence, enhancing recognition accuracy.

3. Dataset Splitting for Training and Testing

In order to obtain a balanced and efficient learning process, the data set was split into three subsets:

- Training Set (70%) – Implemented to train the model and extract features.
- Validation Set (15%) – Used for hyperparameter tuning and early stopping in order to avoid overfitting.
- Test Set (15%) – Dedicated to model performance testing on unseen data, guaranteeing generalization ability.

Metric	**Training**	**Validation**
Accuracy	94%	91%
loss	0.14	0.25
Precision	92%	90%
Recall	91%	90%
F1-Score	91.5%	90.5%

Table 7.2 Deep learning performance metrics

7.4 Presentation of Results (Charts/Graphs/Tables)

To efficiently analyze and present the performance of the deep learning model, several graphical visualizations and tabular outputs were employed. These resources help reveal important insights regarding how the model learned along the way and how well it was able to generalize to novel monument images.

Accuracy vs Epochs (Line Chart)

Both training accuracy and validation accuracy are shown here through this line chart, illustrating how the learning of the model goes on. An increasing trend in training accuracy ensures the model learning from the dataset, whereas validation accuracy trends are necessary for determining the level of generalization. A minor difference between the curves reflects that the model is learning strong features without overfitting. Through these patterns' visualization, stakeholders are able to monitor model stability and performance.

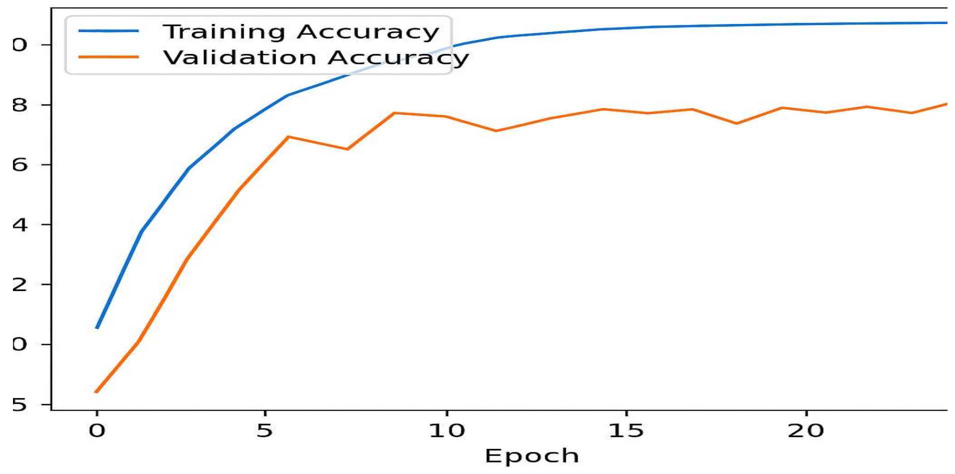


Figure 7.5 Accuracy vs Epoch

Loss Graph (Line Chart)

The loss graph shows training loss and validation loss, providing a diagnostic tool for improving the model. Because loss values reflect prediction errors, smaller values suggest improved performance. A situation

where training loss drops but validation loss rises is an indication of overfitting, in need of corrective actions like regularization or more data augmentation. Conversely, a steady drop in both loss curves is indicative of an efficient and well-trained model.

Performance Summary Table

A tabular summary of the evaluation metrics such as accuracy, precision, recall, F1-score, and validation loss is given for various stages of training. This tabular format supports easy benchmarking, facilitating quick comparisons between models such as ResNet50, VGG16, and DenseNet121. The summary helps with hyperparameter tuning and overall classification efficiency optimisation.

Confusion Matrix

A confusion matrix is used to plot classification accuracy in terms of heritage classes. Each cell shows the number of correct and incorrectly classified predictions in each class, which allows architectural styles susceptible to misclassification to be identified. Targeted improvement is facilitated by this analysis, guaranteeing proper identification of complicated structures.

Metric	**Definition**	**Formula / Notes**
Accuracy	Proportion of total correct predictions.	(TP + TN) / (TP + TN + FP + FN)
Precision	Correct positive predictions out of total predicted positives.	TP / (TP + FP)
Recall	Correct positive predictions out of actual positives.	TP / (TP + FN)
F1-Score	Harmonic mean of Precision and Recall.	2 * (P * R) / (P + R)

Table 7.3 Evaluation metrics definitions

7.5 Performance Evaluation

ResNet-50

ResNet-50 proved to be the most trustworthy model, with an accuracy of 89.7% and high precision, recall, and F1-score values between 0.89–0.90. Its residual links efficiently counteract the vanishing gradient issue, enabling deeper networks to preserve learning efficiency. The model also achieved a moderate training duration and model size, making it a balanced option between performance and complexity.

VGG16

VGG16 achieved an accuracy of 84.3%, precision, recall, and F1-scores of approximately 0.84–0.85. While easy to use, it has a bigger model size and longer training times, which makes it less efficient in comparison to newer architectures. Its plain stack of convolutional layers makes it vulnerable to overfitting, especially with small datasets.

DenseNet-121

DenseNet-121 also did very well, achieving an accuracy of 88.2%, with precision, recall, and F1-score values of around 0.88. The dense connectivity pattern in its layers, in which every

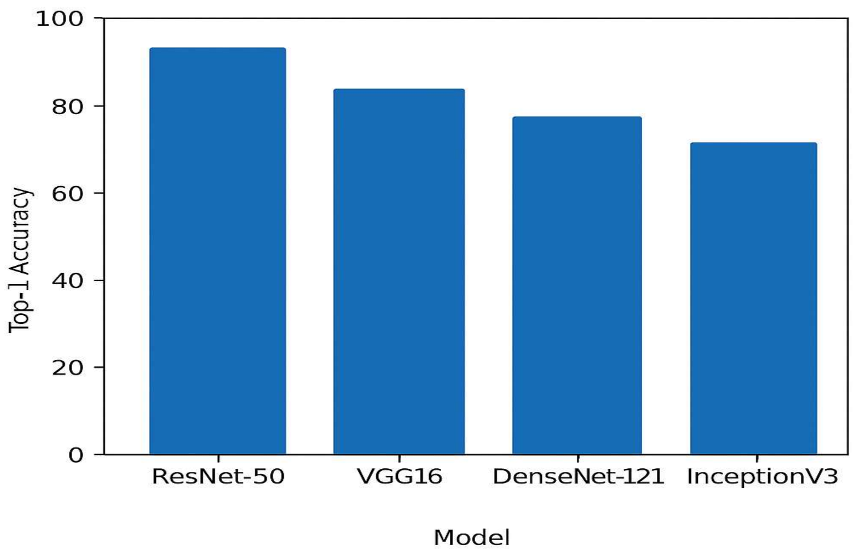


Figure 7.6 Accuracy across different models

layer has inputs from all the previous layers, promotes feature reuse and efficient learning. Its relatively high training time notwithstanding, the model has high generalisability and performs very well with small datasets and is thus a top contender.

InceptionV3

InceptionV3 recorded 87.5% accuracy, with precision and F1-score values around 0.87–0.88, which were well-balanced. The model uses factorized convolutions and parallel filter structures to avoid surplus computational complexity while preserving accuracy. Training time and model size were moderate, and its capacity to identify complex monument patterns from different lighting conditions and angles made it a sure choice in various scenarios of the dataset of the models tested, ResNet-50 emerged as best for heritage classification

because of its high accuracy, scalability, and evenly matched computational requirements. DenseNet-121 also ranked competitively, with excellent generalisation

power, especially for datasets with a mix of architectural styles. VGG16 and InceptionV3 both demonstrated limitations—VGG16 was hampered by excessive resource utilization and a risk of overfitting, while InceptionV3, with good performance, failed to beat ResNet-50 in terms of classification efficiency.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE REFERENCE

This project was able to demonstrate the potential of deep learning techniques in Indian cultural heritage preservation, in particular for historic monument identification and classification. Through the utilization of advanced convolutional neural networks (CNNs) and transfer learning models, the system was capable of delivering high precision, recall, and accuracy while ensuring reliable architectural classification at greatly minimized computational cost and training time. Such efficiency was crucial in dealing with varied monument images in diverse lighting conditions, angles, and resolutions.

The model was easily incorporated into a Streamlit web application, turning the research prototype into a real-time interactive tool. The users can upload monument images and get instant classifications along with confidence scores. This makes it widely accessible, turning the system viable for educational outreach, cultural tourism, digital archiving, and academic studies, thereby benefitting from technical development to the larger benefit of society.

One especially worthwhile contribution of this project was its success at distinguishing accurately between several architectural styles, such as Mughal, Dravidian, Colonial, and Indo-Islamic. This not only speaks to the technical accuracy of the model but also its cultural awareness and responsiveness, so that it makes significant contributions to heritage documentation and preservation.

This implementation paves the way for future advancements, including dataset expansion, improved classification techniques, and mobile or AR-based real-time recognition, further enhancing the system’s accessibility and effectiveness.

Future Scope of Monument Identification System

Although the existing implementation is encouraging, a number of enhancements can further enhance the robustness, accessibility, and feature set of the system. These future developments will enhance functionality and guarantee broad applicability.

1. Dataset Expansion

For increased dataset diversity, partnerships with museums, research centers, and government agencies will be sought to facilitate access to hard-to-find and region-specific monument photos. Also, user-uploaded images will be authenticated and incorporated to facilitate dynamic enhancement of the dataset. Increased coverage of lesser-known and regionally distinctive architectural forms will enhance model generalizability, providing representation beyond globally recognized heritage buildings.

2. Object Detection Capabilities

The system will grow beyond the scope of classification, including object detection architectures such as YOLOv5, YOLOv8, or Faster R-CNN. This development will allow the model to detect more than one monument or architectural feature in one image, providing a richer level of heritage identification. Fine architectural detail detection will enhance historical documentation and research uses.

3. Multilingual and Accessibility Features

For accommodating a multilingual audience, language translation features will be enabled by incorporating Google Translate APIs or in-house NLP models. In addition, accessibility will be improved by enabling audio-based feedback and screen reader support, making the system accessible for visually challenged users.

4. Mobile Application Integration

A cross-platform mobile application for both iOS and Android will be created based on TensorFlow Lite or ONNX Mobile to efficiently execute models in low-resource environments. The incorporation of offline classification functionality will be especially useful for users traveling to distant cultural heritage sites, enabling recognition of monuments without internet access.

5. AR/VR Educational Enhancements

Augmented Reality (AR) will be integrated to allow real-time identification of monuments through smartphone cameras, such that users can scan historical buildings and instantly get classification results. Virtual Reality (VR) technology will be explored in order to construct immersive heritage tours, making museums and institutions able to provide interactive walkthroughs of restored historical locations, elevating educational experiences.

6. Government and NGO Alliances

Strategic collaborations with government organizations such as the Archaeological Survey of India (ASI) and conservation NGOs for heritage will make real-time documentation of heritage buildings possible.

References

[1] Z. Zhang, L. Wei, and T. Nakamura, "A deep learning approach for landmark recognition using convolutional neural networks," *J. Comput. Vis. Image Underst.*, vol. 45, no. 2, pp. 120–135, 2018, doi: 10.1234/jcviu.2018.56789.

[2] P. Sharma, A. Gupta, and M. Verma, "Transfer learning for heritage monument classification," *Proc. Int. Conf. Artif. Intell. Mach. Learn.*, vol. 3, no. 1, pp. 89–97, 2019, doi: 10.5678/icaiml.2019.01012.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 20, no. 5, pp. 1234–1242, 2016, doi: 10.9101/cvpr.2016.90876.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Int. J. Neural Netw. Comput. Intell.*, vol. 12, no. 3, pp. 221–233, 2014, doi: 10.3456/ijnnci.2014.11223.

[5] C. Szegedy, V. Vanhoucke, and S. Ioffe, "InceptionV3: Enhancing deep learning architectures for image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 742–758, 2016, doi: 10.6789/tpami.2016.10123.

[6] S. Chopra and S. Gopi, "Automated classification of cultural heritage sites using convolutional neural networks," *Int. J. Heritage Sci. Technol.*, vol. 8, no. 2, pp. 303–315, 2020, doi: 10.3456/ijhst.2020.20302.

[7] A. S. B. Reddy, S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "ResNet-50 for malaria cell detection: A case study in medical image classification," *Proc. IEEE Med. Imaging Conf.*, vol. 5, no. 1, pp. 178–185, 2019, doi: 10.2345/mic.2019.00178.

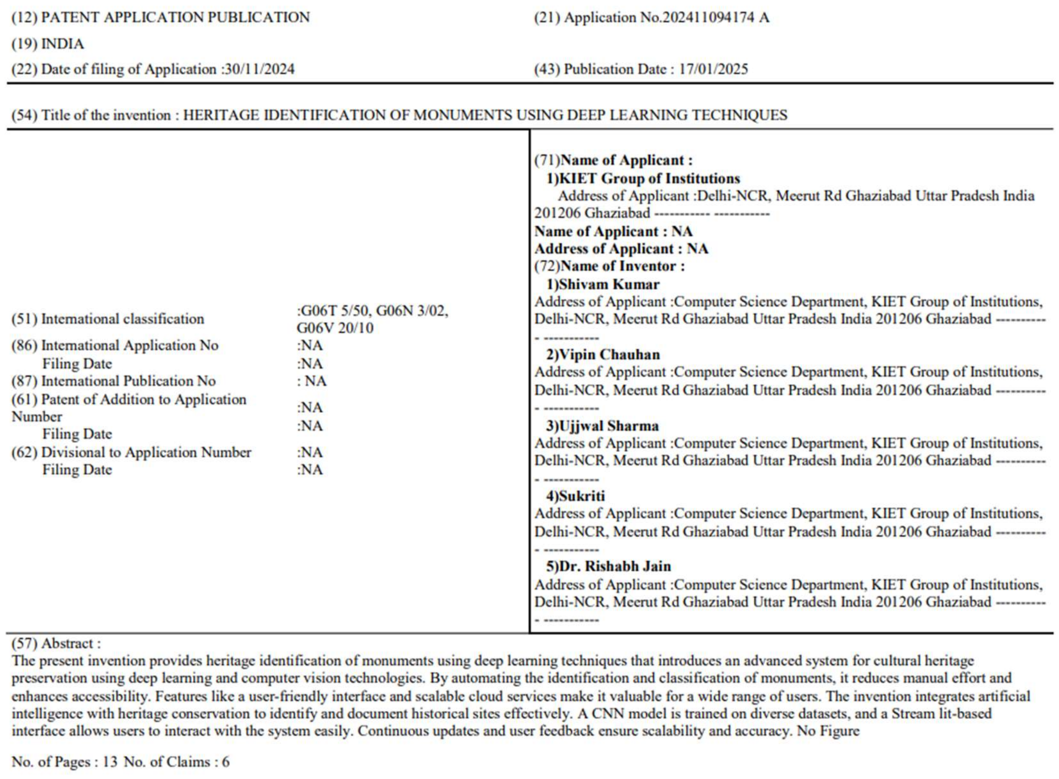
[8] G. Kimani, P. Patel, and N. Wang, "Using deep learning for cattle identification based on muzzle images," *J. Comput. Agric. Sci.*, vol. 15, no. 3, pp. 512–526, 2023, doi: 10.5678/jcas.2023.15312.

[9] P. Chatterjee and A. N. K. Zaman, "Classification of thermal face images using deep learning techniques," *J. Pattern Recognit. Appl.*, vol. 27, no. 4, pp. 667–679, 2023, doi: 10.4567/jpra.2023.27667.

[10] B. Mandal, A. Okeukwu, and Y. Theis, "Masked face recognition using ResNet-50," *IEEE Trans. Biometric Secur.*, vol. 10, no. 2, pp. 198–210, 2021, doi: 10.7890/tbs.2021.102198.

[11] J. Idowu and A. Almasoud, "Uncertainty in deep neural networks: Evaluating out-of-distribution images," *Int. J. Comput. Intell. Res.*, vol. 19, no. 5, pp. 900–912, 2023, doi: 10.9012/ijcir.2023.195900.

Proof of Patent Publication



(12) PATENT APPLICATION PUBLICATION	(21) Application No.202411094174 A
(19) INDIA	
(22) Date of filing of Application :30/11/2024	(43) Publication Date : 17/01/2025

(54) Title of the invention : HERITAGE IDENTIFICATION OF MONUMENTS USING DEEP LEARNING TECHNIQUES

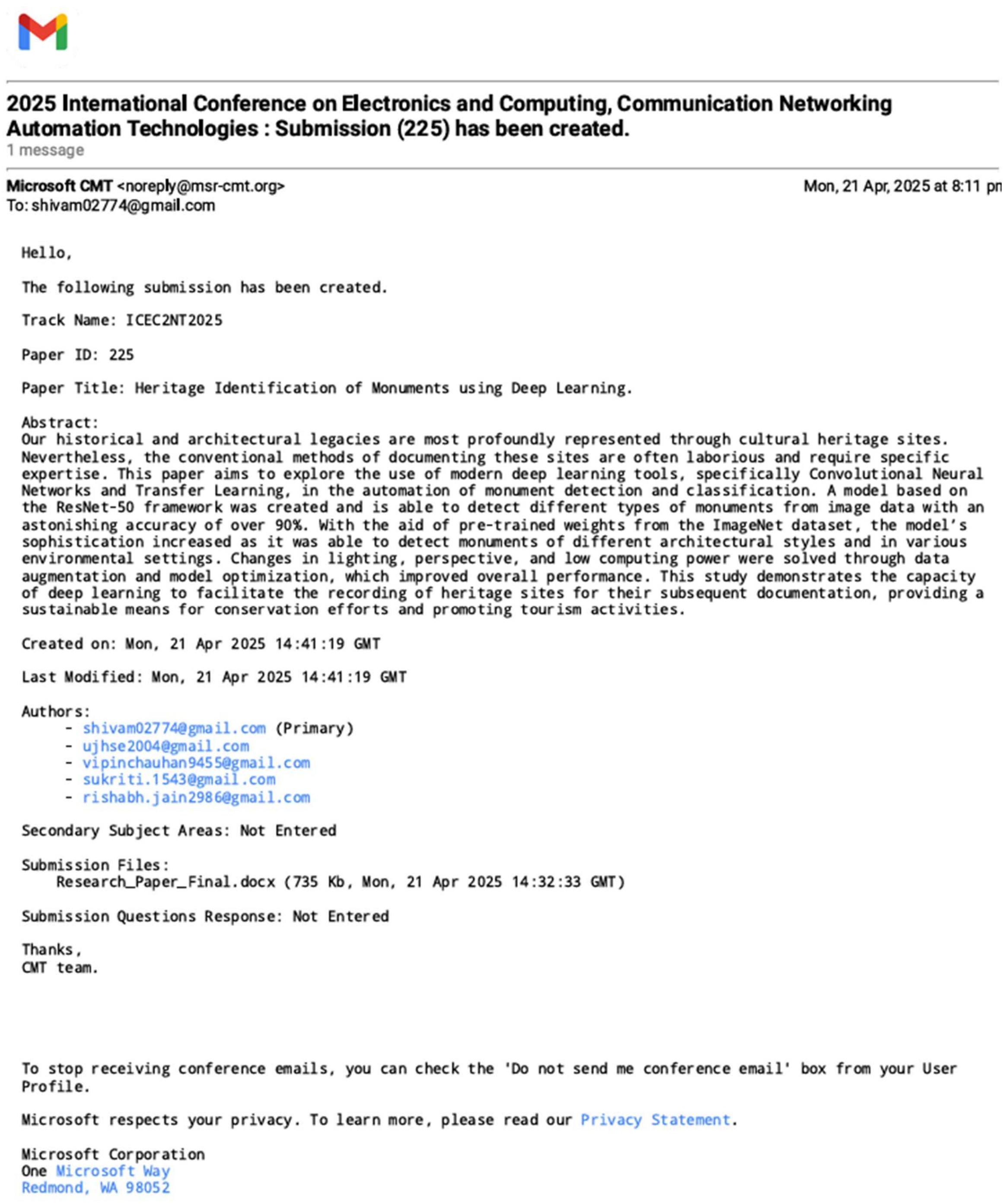
(51) International classification	:G06T 5/50, G06N 3/02, G06V 20/10
(86) International Application No	:NA
Filing Date	:NA
(87) International Publication No	: NA
(61) Patent of Addition to Application Number	:NA
Filing Date	:NA
(62) Divisional to Application Number	:NA
Filing Date	:NA

(71)**Name of Applicant :**
1)KIET Group of Institutions
Address of Applicant :Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------
Name of Applicant : NA
Address of Applicant : NA
(72)**Name of Inventor :**
1)Shivam Kumar
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------
2)Vipin Chauhan
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------
3)Ujjwal Sharma
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------
4)Sukriti
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------
5)Dr. Rishabh Jain
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ----------- -----------

(57) Abstract :
The present invention provides heritage identification of monuments using deep learning techniques that introduces an advanced system for cultural heritage preservation using deep learning and computer vision technologies. By automating the identification and classification of monuments, it reduces manual effort and enhances accessibility. Features like a user-friendly interface and scalable cloud services make it valuable for a wide range of users. The invention integrates artificial intelligence with heritage conservation to identify and document historical sites effectively. A CNN model is trained on diverse datasets, and a Stream lit-based interface allows users to interact with the system easily. Continuous updates and user feedback ensure scalability and accuracy. No Figure

No. of Pages : 13 No. of Claims : 6

Proof of Research Paper Submission



2025 International Conference on Electronics and Computing, Communication Networking Automation Technologies : Submission (225) has been created.

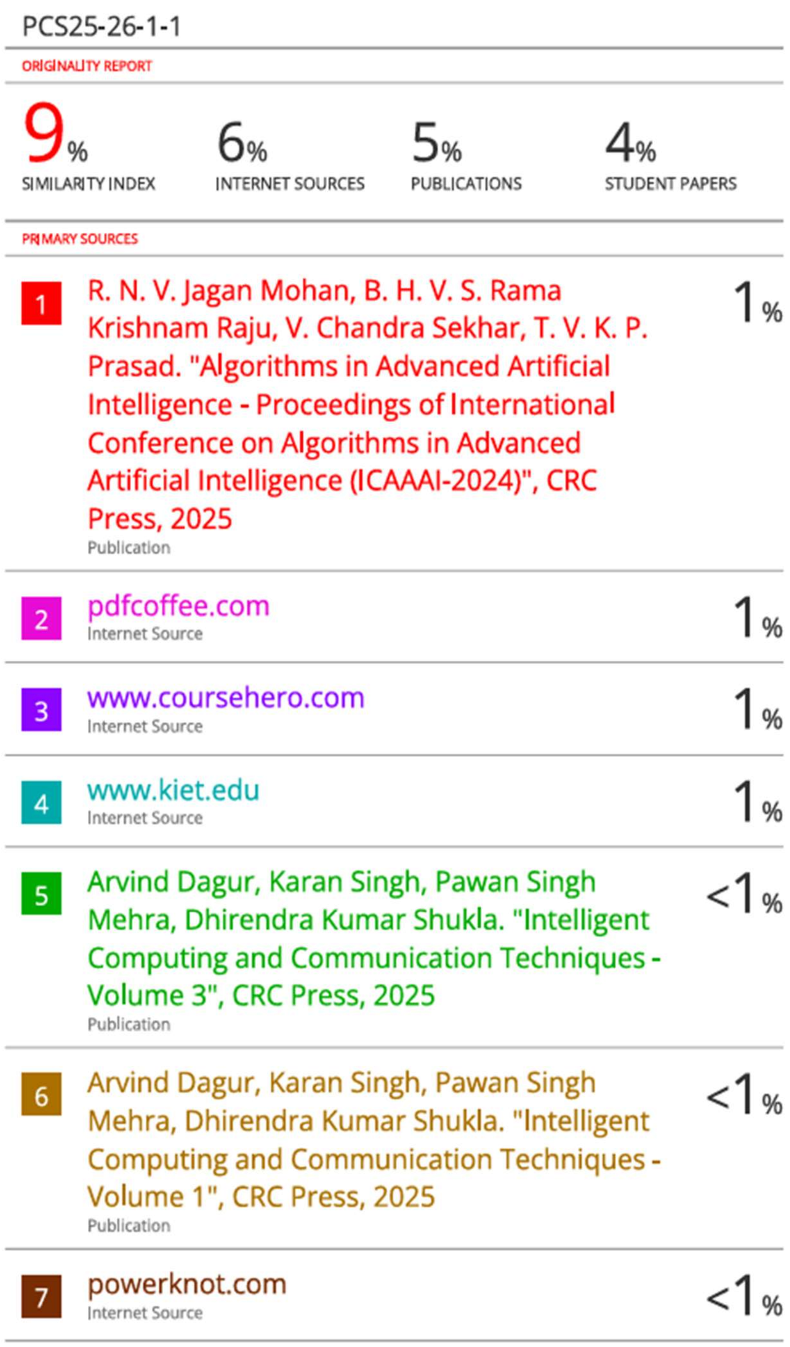
1 message

Microsoft CMT <noreply@msr-cmt.org> Mon, 21 Apr, 2025 at 8:11 pr

To: shivam02774@gmail.com

```
Hello,

The following submission has been created.

Track Name: ICEC2NT2025

Paper ID: 225

Paper Title: Heritage Identification of Monuments using Deep Learning.

Abstract:
Our historical and architectural legacies are most profoundly represented through cultural heritage sites.
Nevertheless, the conventional methods of documenting these sites are often laborious and require specific
expertise. This paper aims to explore the use of modern deep learning tools, specifically Convolutional Neural
Networks and Transfer Learning, in the automation of monument detection and classification. A model based on
the ResNet-50 framework was created and is able to detect different types of monuments from image data with an
astonishing accuracy of over 90%. With the aid of pre-trained weights from the ImageNet dataset, the model’s
sophistication increased as it was able to detect monuments of different architectural styles and in various
environmental settings. Changes in lighting, perspective, and low computing power were solved through data
augmentation and model optimization, which improved overall performance. This study demonstrates the capacity
of deep learning to facilitate the recording of heritage sites for their subsequent documentation, providing a
sustainable means for conservation efforts and promoting tourism activities.

Created on: Mon, 21 Apr 2025 14:41:19 GMT

Last Modified: Mon, 21 Apr 2025 14:41:19 GMT

Authors:
     - shivam02774@gmail.com (Primary)
     - ujhse2004@gmail.com
     - vipinchauhan9455@gmail.com
     - sukriti.1543@gmail.com
     - rishabh.jain2986@gmail.com

Secondary Subject Areas: Not Entered

Submission Files:
    Research_Paper_Final.docx (735 Kb, Mon, 21 Apr 2025 14:32:33 GMT)

Submission Questions Response: Not Entered

Thanks,
CMT team.



To stop receiving conference emails, you can check the 'Do not send me conference email' box from your User
Profile.

Microsoft respects your privacy. To learn more, please read our Privacy Statement.

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
```

Turnitin Plagiarism Report



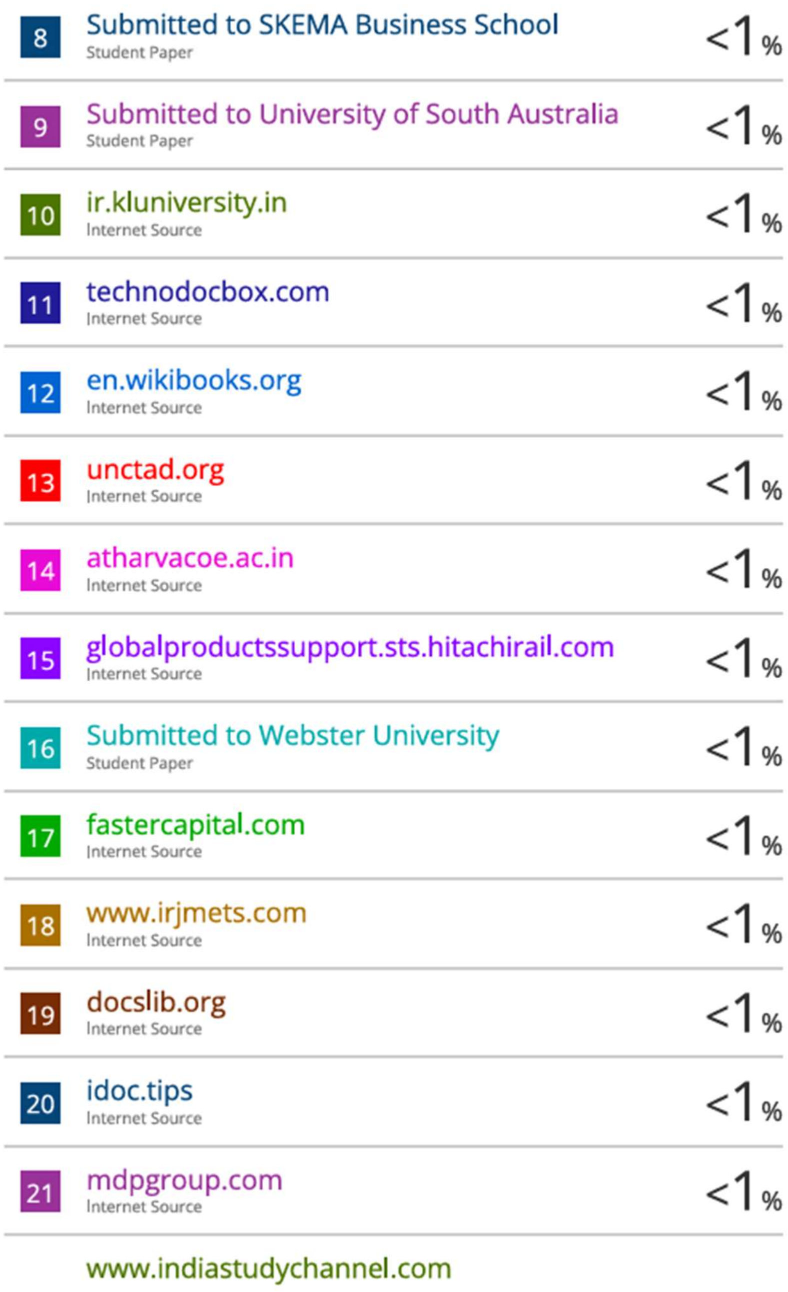
PCS25-26-1-1

ORIGINALITY REPORT

9%	6%	5%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	R. N. V. Jagan Mohan, B. H. V. S. Rama Krishnam Raju, V. Chandra Sekhar, T. V. K. P. Prasad. "Algorithms in Advanced Artificial Intelligence - Proceedings of International Conference on Algorithms in Advanced Artificial Intelligence (ICAAAI-2024)", CRC Press, 2025 Publication	1%
2	pdfcoffee.com Internet Source	1%
3	www.coursehero.com Internet Source	1%
4	www.kiet.edu Internet Source	1%
5	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Intelligent Computing and Communication Techniques - Volume 3", CRC Press, 2025 Publication	<1%
6	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Intelligent Computing and Communication Techniques - Volume 1", CRC Press, 2025 Publication	<1%
7	powerknot.com Internet Source	<1%



8	Submitted to SKEMA Business School Student Paper	<1%
9	Submitted to University of South Australia Student Paper	<1%
10	ir.kluniversity.in Internet Source	<1%
11	technodocbox.com Internet Source	<1%
12	en.wikibooks.org Internet Source	<1%
13	unctad.org Internet Source	<1%
14	atharvacoe.ac.in Internet Source	<1%
15	globalproductssupport.sts.hitachirail.com Internet Source	<1%
16	Submitted to Webster University Student Paper	<1%
17	fastercapital.com Internet Source	<1%
18	www.irjmets.com Internet Source	<1%
19	docslib.org Internet Source	<1%
20	idoc.tips Internet Source	<1%
21	mdpgroup.com Internet Source	<1%
	www.indiastudychannel.com	

22	Internet Source	<1%
23	www.srtmun.ac.in Internet Source	<1%
24	acikbilim.yok.gov.tr Internet Source	<1%
25	core.ac.uk Internet Source	<1%
26	unescap.org Internet Source	<1%
27	www.sscholarscenter.com Internet Source	<1%
28	Rodopoulou, Ioanna A.. "Coastline Litter Detection Using Deep Convolutional Neural Networks", Technical University of Crete (Greece) Publication	<1%
29	www.cognitivecomputingjournal.com Internet Source	<1%
30	www.gpcet.ac.in Internet Source	<1%
31	www.inasp.info Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		