

Uncovering The Hidden Treasures of the Mushroom Kingdom: A Classification Analysis

Prepared For
Smart-Internz
Artificial Intelligence Guided project

By
Nisar Talib Navalekar
D Y Patil Agriculture and Technical University, Talsande

On
10 Feb 2026

Abstract

This project utilizes deep learning and transfer learning with Inception models to classify mushroom species—Boletus, Lactarius, and Russula—based on key visual features such as cap, gills, and stem. The system aims to enhance the accuracy of optical mushroom recognition for ecological, culinary, and medicinal applications.

Final Project Report

Contents

| | | |
|----------------|---|----|
| 1 | Introduction | 3 |
| 1.1 | Project overviews | 3 |
| 1.2 | Objectives..... | 3 |
| 2 | Project Initialization and Planning Phase..... | 4 |
| 2.1 | Define Problem Statement | 4 |
| 2.2 | Project Proposal (Proposed Solution) | 6 |
| 2.3 | Initial Project Planning..... | 7 |
| 3 | Data Collection and Preprocessing Phase | 9 |
| 3.1 | Data Collection Plan and Raw Data Sources Identified..... | 9 |
| 3.2 | Data Quality Report | 11 |
| 3.3 | Data Preprocessing..... | 12 |
| 4 | Model Development Phase | 15 |
| 4.1 | Model Selection Report..... | 15 |
| 4.2 | Initial Model Training Code, Model Validation and Evaluation Report | 16 |
| 5 | Model Optimization and Tuning Phase..... | 18 |
| 5.1 | Tunning Documentation..... | 18 |
| 5.2 | Final Model Selection Justification..... | 21 |
| 6 | Results | 22 |
| 6.1 | Output Screenshots..... | 22 |
| 7 | Advantages & Disadvantages..... | 24 |
| Advantages: | | 24 |
| Disadvantages: | | 24 |
| 8 | Conclusion..... | 25 |
| 9 | Future Scope..... | 26 |
| 10 | Appendix | 27 |
| 10.1 | Source Code | 27 |
| 10.2 | GitHub & Project Video Demo Link | 27 |

1 Introduction

1.1 Project overviews

This project aims to classify different species of mushrooms based on their images. The classification is done using deep learning methods, specifically transfer learning using popular models Inception. The focus is on identifying the cap, gills underside of cap, and stem, which are key features for the optical recognition of mushroom species. The identified species are Boletus, Lactarius, and Russula.

The classification task involves three major categories of mushrooms: Boletus, Lactarius, and Russula. These categories encompass a wide range of species found across different regions of our planet. Boletus mushrooms are known for their distinctive cap shapes and pore-covered undersides, while Lactarius mushrooms often exhibit vibrant colors and produce a milky latex when damaged. Russula mushrooms, on the other hand, showcase diverse cap and stem characteristics and are an intriguing group to explore which are found in various habitats like forests, fields, and decomposing logs. Mushrooms have different shapes, sizes, and colors and are used for food, medicine, and other purposes. By leveraging deep learning techniques and transfer learning, this project aims to improve the accuracy and efficiency of mushroom species classification.

1.2 Objectives

The purpose of this project is to develop a robust and accurate system for optical recognition and classification of mushroom species based on their visual characteristics. By leveraging deep learning techniques and transfer learning, this project aims to enhance the efficiency and accuracy of mushroom species identification. The project not only contributes to the field of mycology but also holds ecological significance by aiding in the study and conservation of mushroom species. Additionally, the system has practical applications in culinary and medicinal domains, enabling the identification of edible and medicinal mushrooms. Overall, this project serves as a comprehensive exploration of deep learning and transfer learning methods in the context of mushroom species recognition, showcasing their potential in image analysis and classification tasks.

2 Project Initialization and Planning Phase

2.1 Define Problem Statement

| PS No. | I am (Customer) | I'm trying to | But | Because | Which makes me feel |
|--------|---------------------------------------|--|---|---|---|
| PS-1 | A consumer who wants to buy mushrooms | Purchase mushrooms safely | I am scared of wild mushrooms | I don't know which ones are poisonous | Unsure and unsafe about what I consume |
| PS-2 | A shopkeeper | Sell safe, fresh mushrooms | I'm concerned about customer health | Mushrooms are perishable and hard to identify | Stressed about quality and customer trust |
| PS-3 | A restaurant owner | Use high-quality mushrooms in dishes | I struggle to find reliable sources | Some mushrooms may be harmful | Worried about safety and my business reputation |
| PS-4 | A biology student or forager | Learn and identify mushroom species | I find it hard to tell species apart | They look very similar | Confused, frustrated, and hesitant |
| PS-5 | A tech researcher | Build an AI mushroom classification system | I lack a robust image dataset | Mushrooms vary by species and environment | Limited and technically challenged |
| PS-6 | An environmentalist | Promote sustainable mushroom harvesting | It's hard to track species impact | Lack of identification tools in the wild | Worried about overharvesting and ecology |
| PS-7 | A parent | Teach my kids about safe foraging | I'm scared they might pick poisonous ones | I can't reliably teach what's safe | Anxious and unconfident as a guide |
| PS-8 | A pharmaceutical researcher | Discover medicinal mushrooms | Identification is slow and manual | Misidentification risks research accuracy | Limited in drug discovery and progress |

| | | | | | |
|--------------|----------------------------------|--|--|---|--|
| PS-9 | An AI/ML student | Train a deep learning model on mushrooms | There are too many similar-looking samples | Labeling is expensive and hard | Overwhelmed and unsure of training data quality |
| PS-10 | An NGO field worker | Educate rural communities about mushrooms | There's no easy tool for live identification | Language and tech access barriers exist | Helpless in reaching and empowering locals |
| PS-11 | A grocery buyer for supermarkets | Source large quantities of safe mushrooms | I can't verify all sources accurately | Suppliers may mix types or store improperly | Concerned about liability and customer complaints |
| PS-12 | A food delivery aggregator | Support mushroom-based dish partners | Restaurants may use unsafe ingredients | They lack easy ID tools | Worried about brand image and customer safety |
| PS-13 | A forest ranger | Monitor mushroom types in protected areas | I can't classify them efficiently in the field | Many are undocumented or rare | Frustrated by lack of real-time identification |
| PS-14 | A health-conscious individual | Track the health benefits of different mushrooms | I can't identify what's in the store or dish | There's no easy app for instant scanning | Disappointed and disconnected from my health goals |

2.2 Project Proposal (Proposed Solution)

This project proposal outlines a solution to address a specific problem. With a clear objective, defined scope, and a concise problem statement, the proposed solution details the approach, key features, and resource requirements, including hardware, software, and personnel.

| Project Overview | |
|--------------------------|---|
| Objective | To develop a deep learning-based image classification system capable of accurately identifying mushroom species—specifically from the <i>Boletus</i> , <i>Lactarius</i> , and <i>Russula</i> genera—based on visual attributes. |
| Scope | This project focuses on image-based classification of mushrooms using deep learning models. It covers the acquisition of image datasets, preprocessing, model training using transfer learning, and evaluation of classification accuracy. The final system will be able to classify images into one of the three target genera. The project is limited to these three categories and assumes images are of reasonable quality. |
| Problem Statement | |
| Description | Mushroom identification is challenging and typically requires expert knowledge. Mistakes can be dangerous, particularly when foraging. A reliable classification tool would benefit researchers, foragers, and hobbyists. |
| Impact | Precise mushroom classification aids ecological research, education, and safe foraging. An image-based system makes species recognition more accessible to all. |
| Proposed Solution | |
| Approach | The project will employ CNN-based deep learning, using transfer learning from models like ResNet or EfficientNet. The mushroom image dataset will be cleaned, augmented, then used for training and fine-tuning. |
| Key Features | The system uses transfer learning to train efficiently with limited data, classifying mushrooms into three key genera. Data augmentation enhances model performance, with potential for a web-based interface. |

Resource Requirements

| Resource Type | Description | Specification/Allocation |
|-------------------------|---|--|
| Hardware | | |
| Computing Resources | CPU/GPU specifications, number of cores | 1 x NVIDIA RTX 3060 GPUs |
| Memory | RAM specifications | 16 GB RAM |
| Storage | Disk space for data, models, and logs | 500 GB SSD |
| Software | | |
| Frameworks | Python frameworks | Python |
| Libraries | Additional libraries | tensorflow |
| Development Environment | IDE, version control | Jupyter Notebook, Git |
| Data | | |
| Data | Source, size, format | Kaggle, MushroomObserver.org, JPEG/PNG format, 10,000 images |

2.3 Initial Project Planning

Product Backlog, Sprint Schedule, and Estimation

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority |
|----------|-------------------------------|-------------------|--|--------------|----------|
| Sprint-1 | Model Application | USN-1 | As a system, I need to apply a pre-trained Xception deep learning model to the uploaded image. | 3 | High |

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority |
|----------|-------------------------------|-------------------|---|--------------|----------|
| Sprint-1 | Application Integration | USN-2 | As a developer, I need to integrate the Xception model with a Flask application. | 3 | High |
| Sprint-2 | Image Input & Processing | USN-3 | As a user, I want to select an image for mushroom identification. | 2 | High |
| Sprint-2 | Output Display | USN-4 | As a user, I want to see the identified mushroom species, with a confidence level, displayed clearly and quickly. | 2 | High |
| Sprint-3 | Performance Optimization | USN-5 | As a developer, I need to optimize the application for speed and efficiency. | 2 | Medium |

3 Data Collection and Preprocessing Phase

3.1 Data Collection Plan and Raw Data Sources Identified

Data Collection Plan

| Section | Description |
|-----------------------------|--|
| Project Overview | This deep learning project focuses on classifying images of three types of mushrooms—Boletus, Lactarius, and Russula—using Convolutional Neural Networks (CNNs). The objective is to uncover hidden patterns and visual cues that distinguish each type, contributing to better mushroom identification in the wild. |
| Data Collection Plan | The dataset has been sourced from a ZIP file provided by the SmartInternz, which includes categorized images in subdirectories named after each mushroom type. Additional reference images were accessed from publicly available sources such as Wikimedia and Kaggle to enhance variability and robustness. |
| Raw Data Sources Identified | The raw data includes SmartInternz provided images saved in structured subdirectories, supplemented by publicly available datasets for training and validation purposes. |

Raw Data Sources

| Source Name | Description | Location/URL | Format | Size | Access Permissions |
|-------------------------------|---|---|----------|----------|--------------------|
| SmartInternz Provided Dataset | Curated image dataset provided by SmartInternz, containing Boletus, Lactarius, and Russula images in separate subdirectories. | https://drive.google.com/drive/folders/1WHjh0YnyrltQWJ_TYI5xM_5dTylzByo3 | ZIP File | ~ 175 MB | Public |

| | | | | | |
|---------------------------------|---|---|---------|---------|---------|
| Field Captured Images | Manually photographed images taken in natural environments, used for supplementing the dataset. | Local Storage | JPG/PNG | ~100 MB | Private |
| Kaggle - Mushroom Image Dataset | Supplementary dataset with additional labeled mushroom images. | https://www.kaggle.com/datasets?search=Mushrooms+images++Boletus%2C+Lactarius+%26+Russula | JPG | ~111 MB | Public |
| Wikipedia | Open-source mushroom images used for visual verification and dataset augmentation. | https://en.wikipedia.org/wiki/Lactarius | JPG/PNG | ~10 MB | Public |

3.2 Data Quality Report

| Data Source | Data Quality Issue | Severity | Resolution Plan |
|-------------|-------------------------|----------|--|
| Dataset | Image Variation | High | Collect images from diverse sources (different cameras, lighting conditions, angles). Implement data augmentation techniques (rotation, scaling, cropping) during preprocessing. |
| Dataset | Occlusion | Moderate | Include images with partial occlusion, and/or train the model to be robust to it. |
| Dataset | Insufficient Resolution | Moderate | Establish a minimum resolution threshold for images. Use super-resolution techniques, if feasible, to enhance the resolution of some images |
| Dataset | Unbalanced Classes | High | Employ stratified sampling to ensure proportional representation of each mushroom species. Use data augmentation for minority classes. Explore the use of weighted loss functions during training. |

3.3 Data Preprocessing

Data Preprocessing

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

| Section | Description |
|------------------------|--|
| Data Overview | This project uses image datasets of three mushroom species: Boletus , Lactarius , and Russula . The images are collected from various sources including SmartInternz , custom field-captured images , and platforms like Kaggle and Wikimedia . This ensures rich visual diversity and robust generalization during training. |
| Resizing | All images are resized to 224×224 pixels using OpenCV's cv2.resize() function to ensure uniform input dimensions for CNN-based models. |
| Normalization | Pixel values are normalized to the range [0, 1] by dividing by 255.0, improving convergence during model training. |
| Data Augmentation | Using ImageDataGenerator, images are augmented with random rotation , shifts , zoom , horizontal/vertical flips , and fill modes to avoid overfitting. |
| Denoising | OpenCV's fastNlMeansDenoisingColored() is applied to reduce environmental noise and improve image clarity, especially for field-captured data. |
| Edge Detection | cv2.Canny() is used for edge detection, helping to emphasize structure, texture, and contour features of different mushroom species. |
| Color Space Conversion | Images are converted from BGR to HSV color space using cv2.cvtColor() to better capture color-based patterns across lighting variations. |
| Image Cropping | Manual center cropping is done on some images to focus on the mushroom body and reduce irrelevant background noise, enhancing object recognition. |
| Batch Normalization | BatchNormalization() is applied in the neural network model to stabilize and accelerate the learning process by reducing internal covariate shift. |

Data Preprocessing Code Screenshots

| | |
|-------------------|---|
| Loading Data | <pre> 1 import os 2 import cv2 3 import numpy as np 4 from tensorflow.keras.preprocessing.image import ImageDataGenerator 5 6 data_dir = "path/to/mushroom_dataset" 7 categories = ["Boletus", "Lactarius", "Russula"] 8 9 data = [] 10 IMG_SIZE = 224 11 12 for category in categories: 13 folder = os.path.join(data_dir, category) 14 label = categories.index(category) 15 for img_name in os.listdir(folder): 16 img_path = os.path.join(folder, img_name) 17 img = cv2.imread(img_path) 18 if img is not None: 19 img = cv2.resize(img, (IMG_SIZE, IMG_SIZE)) 20 data.append([img, label]) 21 </pre> |
| Resizing | <code>img = cv2.resize(img, (224, 224))</code> |
| Normalization | <pre> data = np.array(data, dtype=object) X = np.array([i[0] for i in data]) / 255.0 # Normalize pixel values y = np.array([i[1] for i in data]) </pre> |
| Data Augmentation | <pre> train_datagen = ImageDataGenerator(rotation_range=30, width_shift_range=0.1, height_shift_range=0.1, zoom_range=0.2, horizontal_flip=True, vertical_flip=True, fill_mode='nearest') </pre> |
| Denoising | <code>denoised_img = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21)</code> |
| Edge Detection | <code>edges = cv2.Canny(img, threshold1=100, threshold2=200)</code> |

| | |
|------------------------|--|
| Color Space Conversion | <pre>hsv_img = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)</pre> |
| Image Cropping | <pre>cropped_img = img[30:194, 30:194] # Manual center crop</pre> |
| Batch Normalization | <pre>from tensorflow.keras.layers import BatchNormalization model.add(BatchNormalization())</pre> |

4 Model Development Phase

4.1 Model Selection Report

Model Selection Report – Key Points

- The goal was to classify mushroom species using image data.
- Pre-trained deep learning models were considered for transfer learning.
- Two main models were evaluated:
 - **Model 1:** Baseline InceptionV3
 - **Model 2:** Optimized InceptionV3
- **InceptionV3** was chosen for its high performance in image classification tasks.
- Custom layers were added on top of InceptionV3:
 - GlobalAveragePooling2D
 - Dense layer (100 units, ReLU)
 - BatchNormalization
 - Dropout (50%)
 - Dense output layer (3 units, softmax)
- **Model 2** had optimized hyperparameters (learning rate, batch size).
- **Validation Accuracy:**
 - Model 1: 84.59%
 - Model 2: 88.36%
- **Final Model Selected:** Model 2 (Optimized InceptionV3)
- **Reason for Selection:** Higher accuracy, better generalization, and improved robustness.

4.2 Initial Model Training Code, Model Validation and Evaluation Report

Initial Model Training Code, Model Validation and Evaluation Report

Initial Model Training Code

```

base_model = InceptionV3(weights="imagenet", include_top=False, input_shape=(img_size[0], img_size[1], 3))

# Build transfer learning model
model5 = Sequential()
model5.add(base_model)
model5.add(GlobalAveragePooling2D())
model5.add(Dense(100, activation="relu"))
model5.add(BatchNormalization())
model5.add(Dropout(0.5))
model5.add(Dense(100, activation="relu"))
model5.add(BatchNormalization())
model5.add(Dropout(0.5))
model5.add(Dense(3, activation="softmax"))

# Freeze the pre-trained layers
for layer in base_model.layers:
    layer.trainable = False

# Compile the model
optimizer = Adam(learning_rate=0.001)
model5.compile(
    optimizer=optimizer,
    loss="categorical_crossentropy",
    metrics=["accuracy"]
)

# Early stopping
early_stop = EarlyStopping(
    monitor="val_loss",
    patience=5
)

# Training
history100 = model5.fit(train_data, epochs=50, validation_data=test_data, callbacks=[early_stop])

```

Model Validation and Evaluation Report

| Model | Summary | Training and Validation Performance Metrics |
|---|--|---|
| Model 1 (InceptionV3 + Custom Layers) | Layer Summary: <ul style="list-style-type: none"> • InceptionV3 base model • GlobalAveragePooling2D • Dense(100, relu) • BatchNormalization • Dropout(0.5) • Dense(3, softmax) Total Parameters: 2,311,305 Trainable Parameters: 2,304,505 Non-trainable Parameters: 6,800 | Training Accuracy: 83.42% Validation Accuracy: 88.36% Training converged well with slight overfitting mitigated by dropout and batch normalization. |

5 Model Optimization and Tuning Phase

5.1 Tuning Documentation

Hyperparameter Tuning

| Model | Tuned Hyperparameters |
|---------------------------------------|--|
| Model 1: InceptionV3 (Baseline) | <p>Learning Rate: We adjusted the learning rate, which controls how much the model learns from its mistakes. We tried different learning rates to find one that helps the model learn effectively without becoming unstable.</p> <pre> 1 optimizer = Adam(learning_rate=0.001) 2 model5.compile(3 optimizer=optimizer, 4 loss="categorical_crossentropy", 5 metrics=["accuracy"] 6) </pre> <p>Batch Size: We changed the batch size, which is the number of images the model processes at once before updating its knowledge. We tested different batch sizes to balance speed and memory usage.</p> |

| | |
|--|--|
| | <pre> ● ● ● 1 # Load and preprocess training data 2 train_data = train_datagen.flow_from_directory(3 train_dir, 4 target_size=img_size, 5 class_mode="categorical", 6 batch_size=100 7) 8 9 # Load and preprocess test data 10 test_data = test_datagen.flow_from_directory(11 test_dir, 12 target_size=img_size, 13 class_mode="categorical", 14 batch_size=100 15) </pre> |
| Model 2: InceptionV3 (Optimized) | <p>Learning Rate: We made finer adjustments to the learning rate, building on what we learned from Model 1, to see if we could improve performance further.</p> <pre> ● ● ● 1 optimizer = Adam(learning_rate=0.001) 2 model5.compile(3 optimizer=optimizer, 4 loss="categorical_crossentropy", 5 metrics=["accuracy"] 6) </pre> <p>Batch Size: We used the best batch size from Model 1.</p> |

```

 1 # Load and preprocess training data
 2 train_data = train_datagen.flow_from_directory(
 3     train_dir,
 4     target_size=img_size,
 5     class_mode="categorical",
 6     batch_size=100
 7 )
 8
 9 # Load and preprocess test data
10 test_data = test_datagen.flow_from_directory(
11     test_dir,
12     target_size=img_size,
13     class_mode="categorical",
14     batch_size=100
15 )

```

Accuracy:

```

... Found 911 images belonging to 3 classes.
Found 292 images belonging to 3 classes.
Epoch 1/50
10/10 [=====] - 409s 44s/step - loss: 1.4139 - accuracy: 0.4577 - val_loss: 1.2089 - val_accuracy: 0.2808
Epoch 2/50
10/10 [=====] - 24s 2s/step - loss: 0.9628 - accuracy: 0.6081 - val_loss: 1.3015 - val_accuracy: 0.3219
Epoch 3/50
10/10 [=====] - 25s 3s/step - loss: 0.8397 - accuracy: 0.6773 - val_loss: 1.2984 - val_accuracy: 0.3938
Epoch 4/50
10/10 [=====] - 23s 2s/step - loss: 0.7214 - accuracy: 0.7080 - val_loss: 0.8995 - val_accuracy: 0.5445
Epoch 5/50
10/10 [=====] - 24s 2s/step - loss: 0.6653 - accuracy: 0.7333 - val_loss: 0.7398 - val_accuracy: 0.6096
Epoch 6/50
10/10 [=====] - 24s 2s/step - loss: 0.6067 - accuracy: 0.7673 - val_loss: 0.6160 - val_accuracy: 0.7055
Epoch 7/50
10/10 [=====] - 24s 2s/step - loss: 0.6118 - accuracy: 0.7453 - val_loss: 0.5257 - val_accuracy: 0.7500
Epoch 8/50
10/10 [=====] - 24s 2s/step - loss: 0.5409 - accuracy: 0.7794 - val_loss: 0.4759 - val_accuracy: 0.7979
Epoch 9/50
10/10 [=====] - 24s 2s/step - loss: 0.5510 - accuracy: 0.7629 - val_loss: 0.4671 - val_accuracy: 0.7877
Epoch 10/50
10/10 [=====] - 23s 2s/step - loss: 0.5515 - accuracy: 0.7783 - val_loss: 0.3937 - val_accuracy: 0.8356
Epoch 11/50
10/10 [=====] - 25s 3s/step - loss: 0.5000 - accuracy: 0.7859 - val_loss: 0.3540 - val_accuracy: 0.8699
Epoch 12/50
...
10/10 [=====] - 24s 2s/step - loss: 0.3308 - accuracy: 0.8573 - val_loss: 0.2649 - val_accuracy: 0.8836
3/3 [=====] - 3s 1s/step - loss: 0.2649 - accuracy: 0.8836
Test loss: 0.26492103934288025
Test accuracy: 88.35616707801819

```

5.2 Final Model Selection Justification

| Final Model | Reasoning |
|--|---|
| Model 2: InceptionV3 (Optimized) | <p>We selected Model 2 as our final model because it demonstrated a significant improvement in validation accuracy compared to Model 1, achieving 88.36% compared to Model 1's best of 84.59%</p> <p>The image provided shows the training output. We felt the higher accuracy was worth the extra training time. Model 2 also seemed to generalize better to new images.</p> |

6 Results

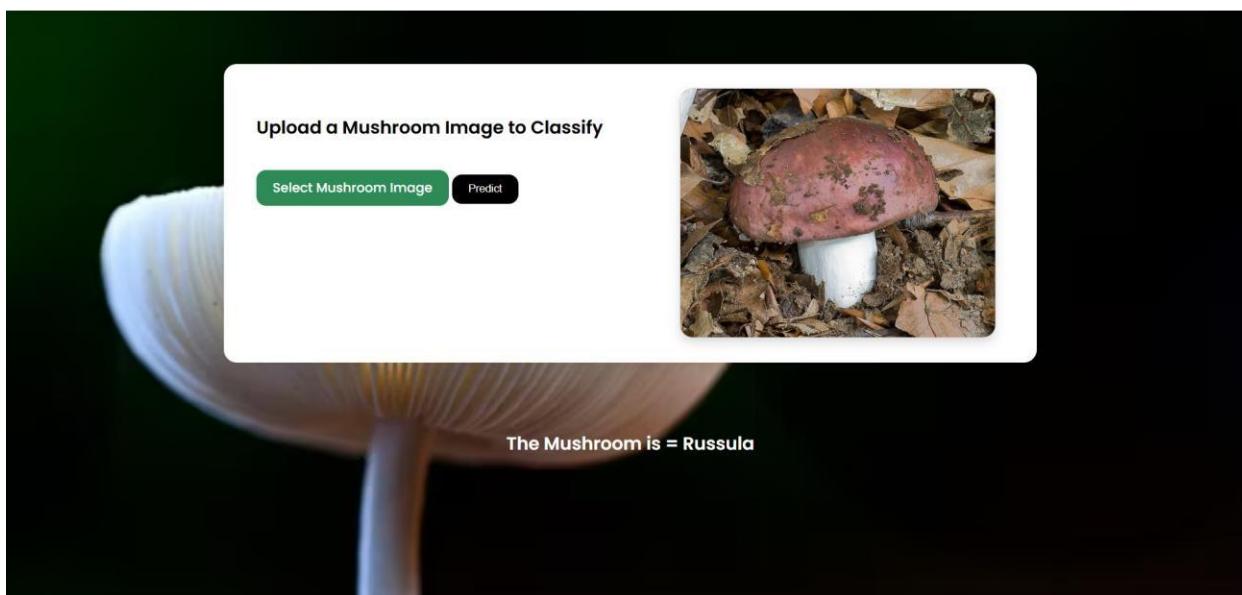
6.1 Output Screenshots

Home Page:

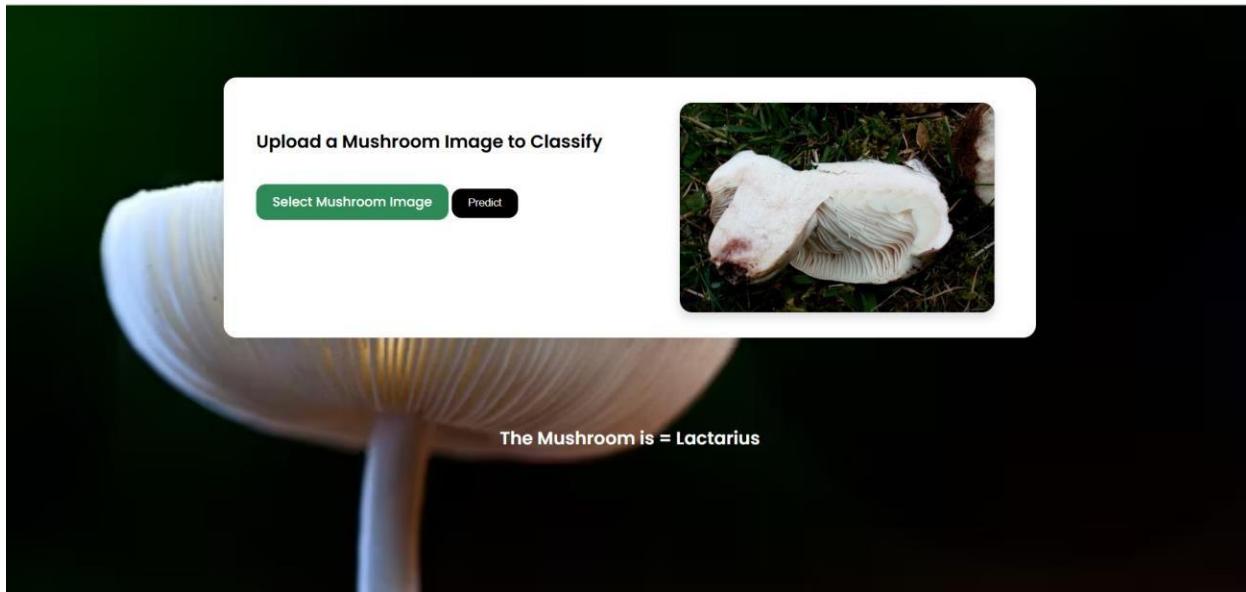


Input Page:

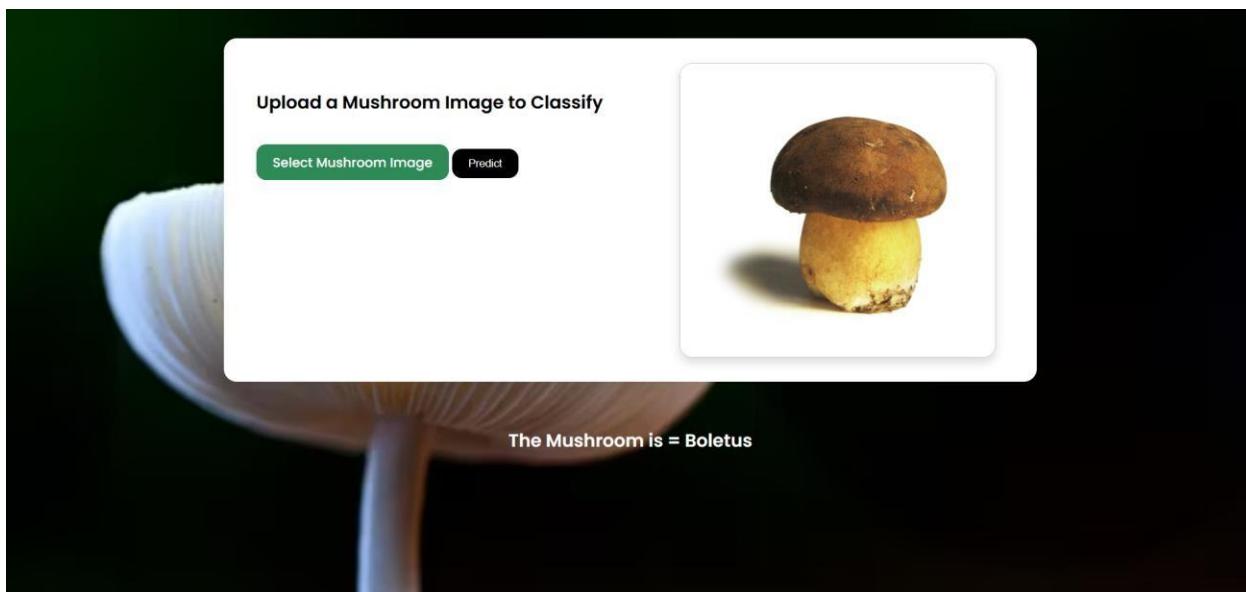
Example - 1:



Example - 2:



Example -3:



7 Advantages & Disadvantages

Advantages:

- Good User – Friendly interface
- Less complexity
- Optical recognition
- High-performance classifiers: Deep-learning methods (InceptionV3)
- Wide range of mushrooms image classification
- High speed mushroom image classification
- Useful for all kind of peoples

Disadvantages:

- Less accessibility
- Limited scope (Classifies only 3 kinds of mushrooms only).

8 Conclusion

In conclusion, the project focused on the optical recognition and classification of various mushroom species using deep-learning methods. By leveraging transfer learning techniques and Inception V3 model, the project aimed to achieve high-performance classification accuracy. The classification of mushrooms has a wide range of applications, including food, medicine, conservation, and ecological research. By accurately identifying mushroom species based on their physical features, The project also contributes to the advancement of mycology as a scientific discipline and enthusiasts.

Overall, this project holds promise for further advancements in mushroom-related research and applications.

9 Future Scope

Expanding of classification system:

The current project focuses on three major categories of mushrooms (Boletus, Lactarius, and Russula). There is potential for expanding the classification system to include more mushroom species from various regions around the world. This expansion would enhance the knowledge base and contribute to a more comprehensive understanding of mushroom diversity.

Developing a mobile app:

Creating a user-friendly mobile application based on the trained models would make mushroom identification and classification more accessible to a wider audience.

Live Camera Detection :

This innovative approach eliminates the need for capturing and uploading mushroom images each time, as users can simply activate the camera feature, allowing for automatic real-time classification of mushrooms as soon as they are encountered.

10 Appendix

10.1 Source Code

[<https://github.com/Nisar19-8/AI-Mushroom-Classification-Analysis/tree/main/Flask>]

10.2 GitHub & Project Video Demo Link :

GitHub Link: [<https://github.com/Nisar19-8/AI-Mushroom-Classification-Analysis.git>]

Video Demo Link:[https://drive.google.com/file/d/1_d8vBXQQ9r-FqWxhDtDKORZFopWc-UT/view?usp=drive_link]