GARBAGE CLASSIFICATION USING DEEP LEARNING

DOCUMENTATION

TEAM MEMBERS:

GAMMINGI ADARSH

NANDAVARAPU DEVI VARAPRASAD

TIMMAPATRUNI PAVANKUMAR

MUNUGOTI NAGENDRAPRASAD

Alluri SajithVarma

Garbage Classification using DeepLearning

Introduction

The Garbage Classification Using Deep Learning project is an innovative web application that utilizes state-of-the-art deep learning techniques to classify images of garbage into specific categories. The main objective of this project is to contribute to waste management and environmental conservation efforts by providing an efficient and user-friendly tool for waste classification.

Waste management is a critical global issue, and improper disposal of waste can have severe environmental consequences. To address this problem, accurate waste classification is essential, as it allows for effective recycling, reuse, and proper disposal of waste materials. However, manual waste classification can be time-consuming and error-prone.

This web application leverages the power of deep learning to automate the waste classification process. The underlying deep learning model has been trained on a diverse dataset of garbage images, allowing it to identify different types of waste materials accurately. By providing users with an intuitive interface to upload images of garbage, the web app quickly classifies the waste into categories such as cardboard, glass, metal, paper, plastic, and trash.

The Garbage Classification project not only contributes to waste management but also raises awareness about the importance of responsible waste disposal and recycling. With this tool, individuals, businesses, and waste management organizations can make informed decisions to reduce their environmental footprint.

The user-friendly nature of the web application makes it accessible to a wide audience, from individuals interested in waste management to educational institutions and environmental organizations seeking practical waste classification solutions. By empowering users with an efficient waste classification tool, we aim to promote a cleaner and more sustainable environment for future generations.

In the following sections of this documentation, we will explore the technical details of the project, including the model architecture, data preprocessing, and the implementation of the web application. Additionally, we will provide step-by-step instructions on how to use the web app and navigate its various features. Let's dive into the details and start making a positive impact on waste management and environmental conservation!

Prerequisites

Before using the Garbage Classification web application, ensure that you have the following prerequisites installed:

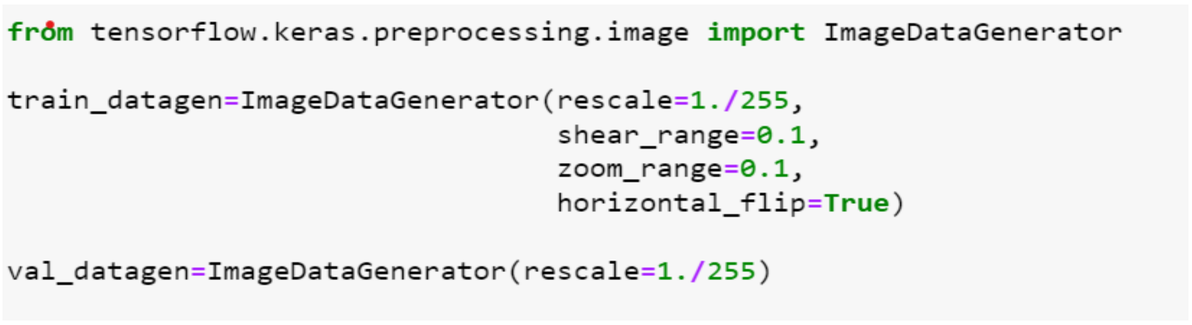
- Anaconda navigator(contains jupyter notebook,spyder etc)
- Jupyter notebook
- Python Libraries which should contain tensorflow,keras,numpy,pandas
- flask

Technical Details

Dataset Preparation and Data Augmentation

The first step in building the Garbage Classification project involved preparing the dataset. We collected a diverse set of garbage images and organized them into six categories: cardboard, glass, metal, paper, plastic, and trash. This dataset was then split into training, validation, and test sets.

To increase the dataset's size and enhance the model's generalization ability, we used data augmentation techniques. The `ImageDataGenerator` class from the Keras library was employed to apply various transformations to the images, such as rotation, scaling, and horizontal flipping.



```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen=ImageDataGenerator(rescale=1./255,
                                 shear_range=0.1,
                                 zoom_range=0.1,
                                 horizontal_flip=True)

val_datagen=ImageDataGenerator(rescale=1./255)
```

Model Architecture

The core of the Garbage Classification project is the deep learning model used for waste classification. We implemented a convolutional neural network (CNN) architecture from scratch using the Keras library.

The model architecture comprises several layers, including Convolutional, MaxPooling, BatchNormalization, and Dense layers. The Convolutional layers learn to extract relevant features from the input images, while MaxPooling layers reduce the spatial dimensions of the feature maps. BatchNormalization layers were added to accelerate the training process and improve model convergence.

After several Convolutional and MaxPooling layers, the feature maps were flattened and passed through Dense layers with ReLU activation functions. Dropout layers were introduced to prevent overfitting, and the final layer used a softmax activation function for multi-class classification.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from tensorflow.keras.optimizers import Adam
```

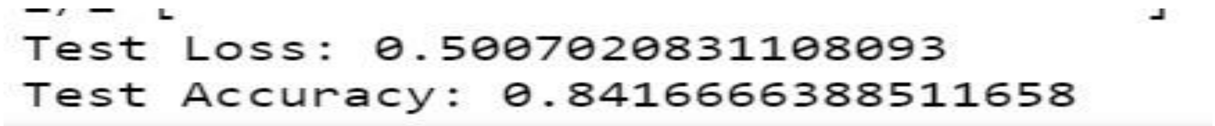
Model Training and Evaluation

The model was trained on the augmented dataset using the Adam optimizer and categorical cross-entropy loss function. During training, a learning rate schedule was applied to adjust the learning rate after a certain number of epochs, leading to improved convergence.

To monitor the model's performance, we evaluated it on the validation set during each epoch. This allowed us to track the loss and accuracy metrics and identify any signs of overfitting or underfitting.

After training, we evaluated the model's performance on the test set to get an accurate measure of its classification capabilities. The test loss and accuracy served as key metrics to assess the model's overall performance.

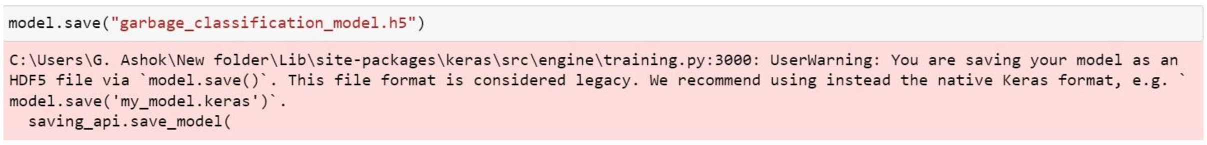
```
# Compile the model
model.compile(loss='categorical_crossentropy', optimizer=Adam(), metrics=['acc'])

# Train the model
res = model.fit(
    train_transform,
    steps_per_epoch=len(train_transform),
    epochs=25,
    validation_data=val_transform,
    validation_steps=len(val_transform),
    callbacks=[lr_scheduler]
)
test_steps = len(test_transform)
test_loss, test_accuracy = model.evaluate(test_transform, steps=test_steps)
print("Test Loss:", test_loss)
print("Test Accuracy:", test_accuracy)
```



```
Test Loss: 0.5007020831108093
Test Accuracy: 0.8416666388511658
```

Model Deployment as a Web Application

Once we had a well-trained model, we need to save the model



```
model.save("garbage_classification_model.h5")
```

```
C:\Users\G. Ashok\New folder\Lib\site-packages\keras\src\engine\training.py:3000: UserWarning: You are saving your model as an
HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `
model.save('my_model.keras')`.
  saving_api.save_model(
```

the next step was to deploy it as a web application using Flask. Flask is a lightweight web framework in Python that allowed us to create a simple and efficient web app.

We created a Flask app and defined two main routes: the home page and the prediction route. The home page displayed the user interface, allowing users to upload an image of garbage. Upon image submission, the image was sent to the prediction route, where the model classified the garbage into one of the predefined categories.

Integration of HTML and CSS

To enhance the web application's visual appeal and user experience, we designed a simple yet elegant HTML template. The HTML template consisted of the home page layout, including the image upload form and the result display area. The html file should be saved with **“.html”extension**

We also integrated CSS styles to customize the appearance of the web app. The CSS styles were applied to various elements, such as the header, buttons, and image container, to ensure a cohesive and user-friendly design. The css file should be saved with **“.css”extension**

Running the Application

The Garbage Classification web application was tested and run locally using Jupyter Notebook and Anaconda Prompt. The application could be accessed by running the Flask app, which started the local development server. Users could access the web app through their web browser by navigating to the provided local address.

The final web application provides an intuitive and interactive interface for users to classify garbage images accurately. By automating the waste classification process, the Garbage Classification project aims to contribute to waste management and promote environmental conservation efforts.

Usage

Step 1: Running the Flask App

To run the web application, execute the following command in the terminal

```
Python app.py
```

This will start the Flask development server, and the web app will be accessible at **`http://127.0.0.1:5000/`**.

Step 2: Classifying Images

Using the Garbage Classification web app, you can classify images of garbage into specific categories. Follow these steps:

1. Click on the "Choose File" button to select an image from your local system.
2. After selecting the image, click the "upload" button to process the image using the pre-trained model.
3. The web app will display the predicted class label below the image, indicating the type of waste.

Step 3: Navigation Buttons

On the top-right corner of the web app, you will find three buttons:

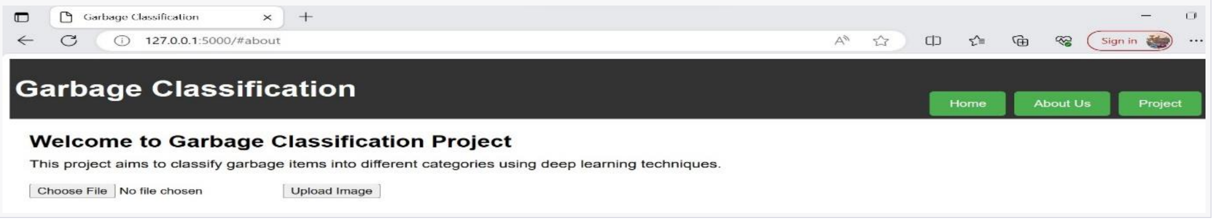
- **Home**: It shows the information of our project
- **About Us**: Clicking this button will display information about the project, its goals, and its contributors.
- **Project**: Clicking this button will provide detailed information about the Garbage Classification Project, including its objectives and implementation.

Folder Structure

The Garbage Classification project has the following folder structure:

- **app.py**: Contains the Flask app code for handling HTTP requests and responses.
- **garbage_classification_model.h5**: The pre-trained deep learning model for garbage classification.
- **static**: This folder contains CSS stylesheets used for styling the web app.
- **templates**: Contains the HTML templates for rendering the web pages.

THE RESULTED WEB PAGE OF OUR PROJECT



Garbage Classification

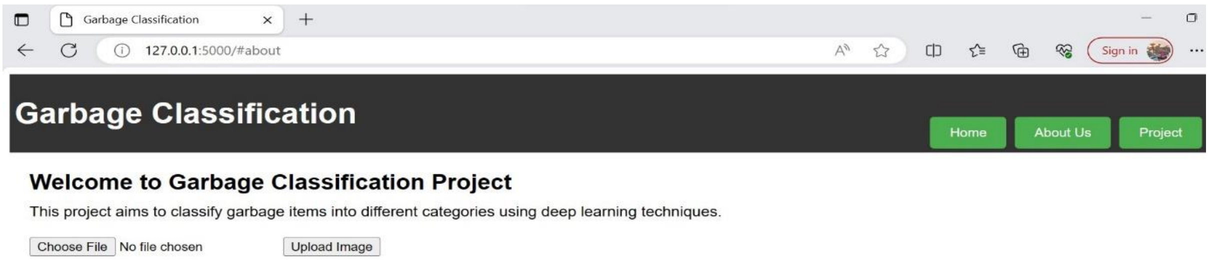
127.0.0.1:5000/#about

Garbage Classification

Welcome to Garbage Classification Project

This project aims to classify garbage items into different categories using deep learning techniques.

THE OUTPUT



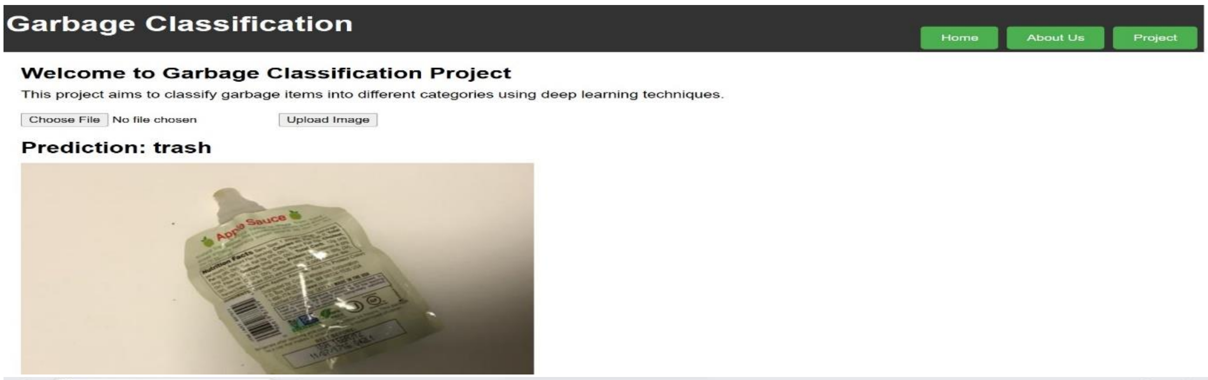
Garbage Classification

127.0.0.1:5000/#about

Garbage Classification

Welcome to Garbage Classification Project

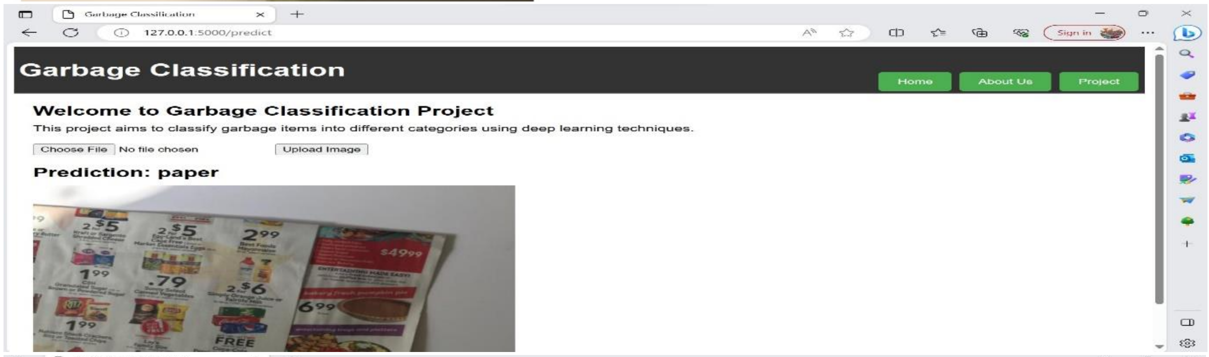
This project aims to classify garbage items into different categories using deep learning techniques.


Garbage Classification

Welcome to Garbage Classification Project

This project aims to classify garbage items into different categories using deep learning techniques.

Prediction: trash

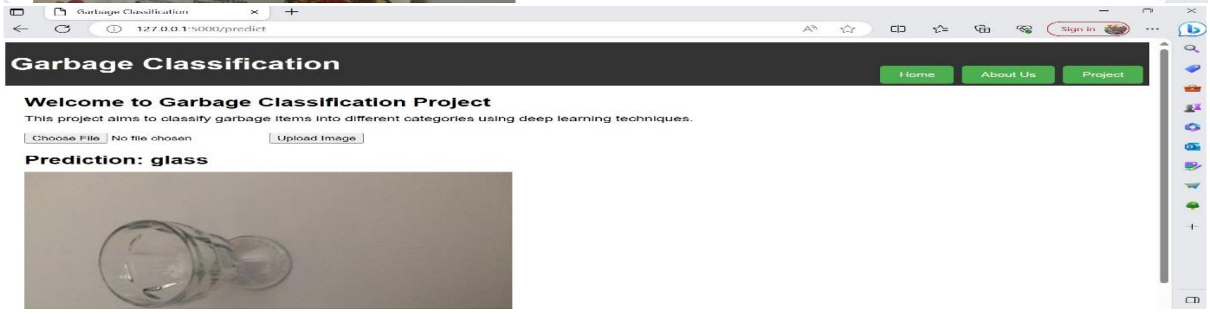

Garbage Classification

127.0.0.1:5000/predict

Welcome to Garbage Classification Project

This project aims to classify garbage items into different categories using deep learning techniques.

Prediction: paper


Garbage Classification

127.0.0.1:5000/predict

Welcome to Garbage Classification Project

This project aims to classify garbage items into different categories using deep learning techniques.

Prediction: glass

CONCLUSION

The Garbage Classification Using Deep Learning project aims to promote waste management and environmental awareness by providing a simple and efficient way to classify garbage images. By using this web application, users can contribute to a cleaner environment by efficiently categorizing different types of waste materials. We hope that this project helps in raising awareness about proper waste disposal and encourages sustainable practices in waste management. Thank you for using the Garbage Classification web app!

REFERENCES

HTTPS://APSCHE.SMARTINTERNZ.COM/STUDENT/EXTERNSHIPS_WORKSPACE_INFO/6999