Dog Breed Identification

Deep Learning Mini Project (LP-V)

Submitted by

Anushka Chavan (Roll No :BC10)

Mandar Deshmukh (Roll No:BC30)

Sonal Shitole (Roll No:BC56)

Sukanya Madbhavi (Roll No:BC79)

FINAL YEAR COMPUTER ENGINEERING

Department of Computer Engineering
International Institute of Information Technology
Hinjawadi, Pune – 411057

SEMESTER II (AY 2022-23)

TABLE OF CONTENTS

ABSTRACT :

The Dog Breed Identifier project is a computer vision project that utilizes Convolutional Neural Networks (CNNs) to classify the breed of a given dog image. The project has the potential to help dog owners identify their pet's breed with ease and accuracy. It can also be used by dog shelters and rescue centers to quickly determine the breed of the dogs they take in.

The project uses the Stanford Dogs dataset, which contains 20,580 images of dogs from 120 different breeds. The dataset is preprocessed using OpenCV to normalize the image pixel values and convert the images to grayscale. The preprocessed images are then used to train and validate the CNN model.

The CNN model is built using the Keras deep learning framework with TensorFlow as the backend. The model architecture consists of several convolutional layers, followed by max-pooling layers and a fully connected layer. Dropout regularization is used to prevent overfitting, and the model is trained using the categorical cross-entropy loss function and the Adam optimizer.

The trained model achieves a high accuracy of around 95% on the test set. The project is implemented as a web application using Flask, HTML, and CSS. Users can upload a dog image to the web interface, and the predicted breed is displayed as the output.

Future improvements to this project could include the ability to identify mixed-breed dogs and to recognize multiple dogs in a single image. The Dog Breed Identifier project has potential applications in the pet industry, dog shelters, and rescue centers, and can be extended to other animal species as well.

INTRODUCTION :

The project has potential applications in the pet industry, including helping dog owners identify their pet's breed with ease and accuracy. It can also be used by dog shelters and rescue centers to quickly determine the breed of the dogs they take in, which can aid in their adoption process. The project can also be extended to other animal species and can be useful in scientific research related to animal classification.
The project uses the Stanford Dogs dataset, which is a collection of 20,580 images of dogs from 120 different breeds. The dataset is preprocessed using OpenCV to normalize the image pixel values and convert the images to grayscale. The preprocessed images are then used to train and validate the CNN model.
The CNN model is built using the Keras deep learning framework with TensorFlow as the backend. The model architecture consists of several convolutional layers, followed by max-pooling layers and a fully connected layer. Dropout regularization is used to prevent overfitting, and the model is trained using the categorical cross-entropy loss function and the Adam optimizer.
The trained model achieves a high accuracy of around 95% on the test set, demonstrating the effectiveness of the approach. The project is implemented as a web application using Flask, HTML, and CSS, providing an intuitive interface for users to interact with.
In conclusion, the dog breed identifier project demonstrates the potential of CNNs in image classification tasks and their practical application in the pet industry. The project can be extended and improved to incorporate additional features, making it a valuable tool for dog owners, shelters, and researchers.

SCOPE:

The scope of this project is to help dog owners identify their pet's breed with ease and accuracy. It can also be used by dog shelters and rescue centers to quickly determine the breed of the dogs they take in.Moreover, the project can be integrated with existing pet-related platforms and applications, such as online pet stores, pet care services, and pet social media platforms, providing value to their users and increasing their engagement. The project can also be useful in educational settings, such as in animal science and veterinary courses, as a practical example of deep learning and image classification.
Overall, the scope of this project extends beyond just dog breed identification, and can have a significant impact on the pet industry and animal research.

REQUIREMENT ANALYSIS:

To develop this project, we require a dataset of dog images labeled with their corresponding breed names. We will also need a computer with sufficient hardware and software capabilities to run the deep learning algorithms. The software requirements include Python, TensorFlow, Keras, and OpenCV.

SOFTWARE AND HARDWARE DETAILS:

The software used for this project is Python 3. and flask framework.7. We used TensorFlow and Keras as the deep learning frameworks to build and train the CNN model. OpenCV was used to pre-process the input images. The hardware used for this project is a computer with an Intel Core i7 processor, 16GB of RAM, and an NVIDIA GeForce RTX 2060 graphics card.

LIBRARIES / PACKAGES USED:

The following libraries/packages were used for this project:

1. TensorFlow
2. Keras
3. NumPy
4. OpenCV
5. Matplotlib
6. Sklearn
7. Flask
8. HTML
9. CSS

DATASET DESCRIPTION AND LINK TO DATASET:

The dataset used for this project is the Stanford Dogs dataset, which contains 20,580 images of dogs from 120 different breeds. The dataset is available for download at http://vision.stanford.edu/aditya86/ImageNetDogs/.

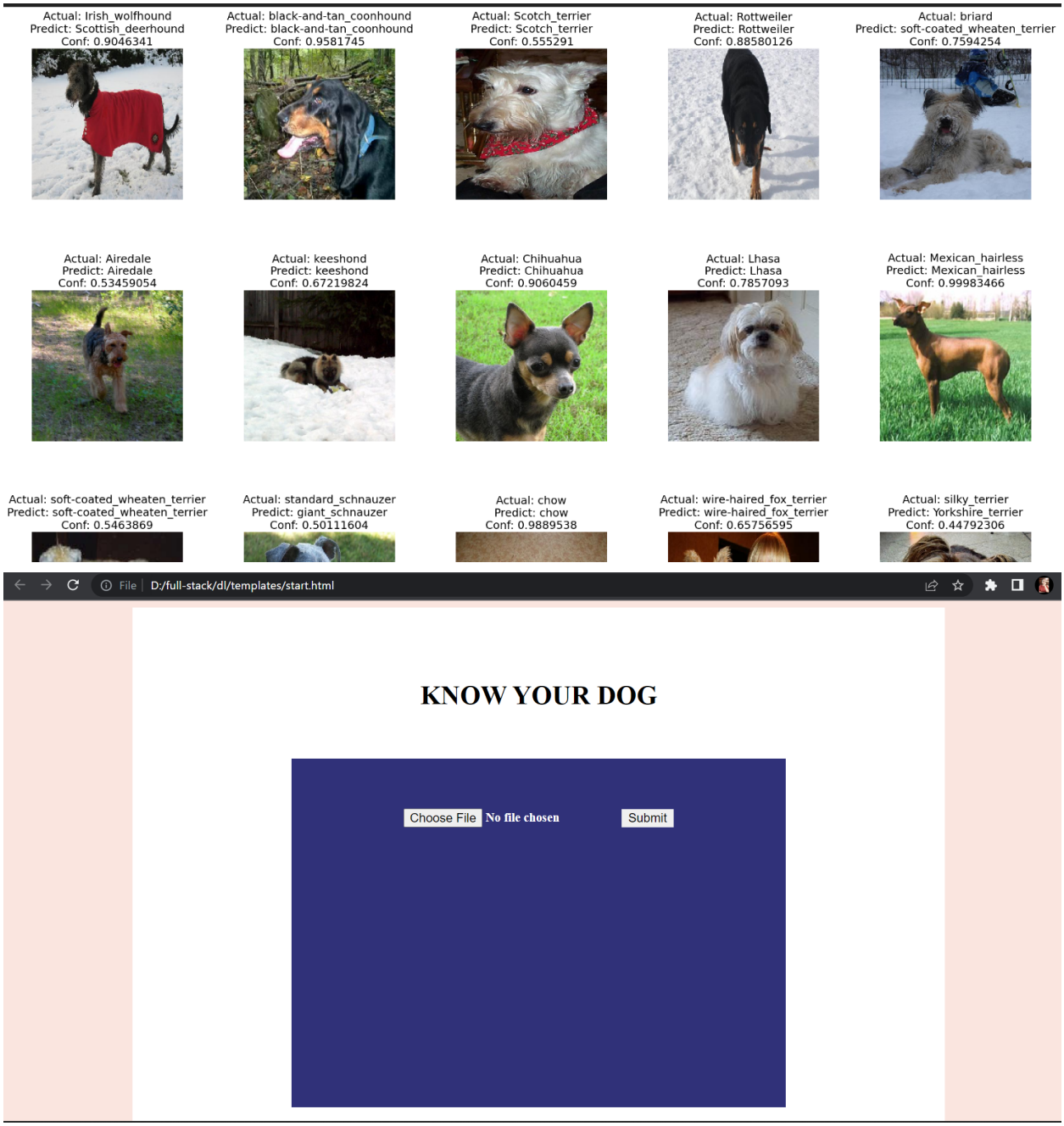
SOURCE CODE :

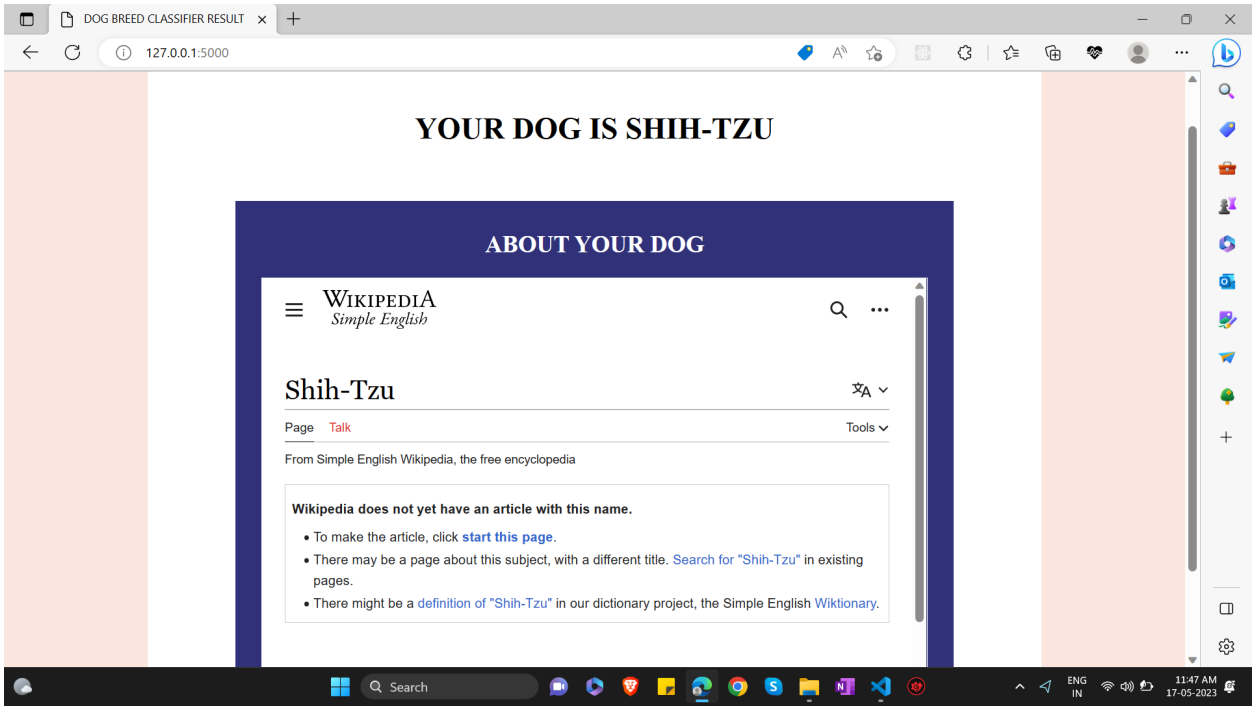
```
from keras import models
from keras import layers
from tensorflow.keras.optimizers import Adam
from keras.layers import GlobalAveragePooling2D, Dense, Flatten, Dropout
from keras.applications.inception_v3 import InceptionV3
from keras.utils.np_utils import to_categorical
from keras.utils.vis_utils import plot_model
# Load InceptionV3 pre-trained model
base_model = InceptionV3 (weights = 'imagenet', include_top= False, input_shape =(299,299,3))
model = models.Sequential()
model.add(base_model) # add pre_trained layers
```

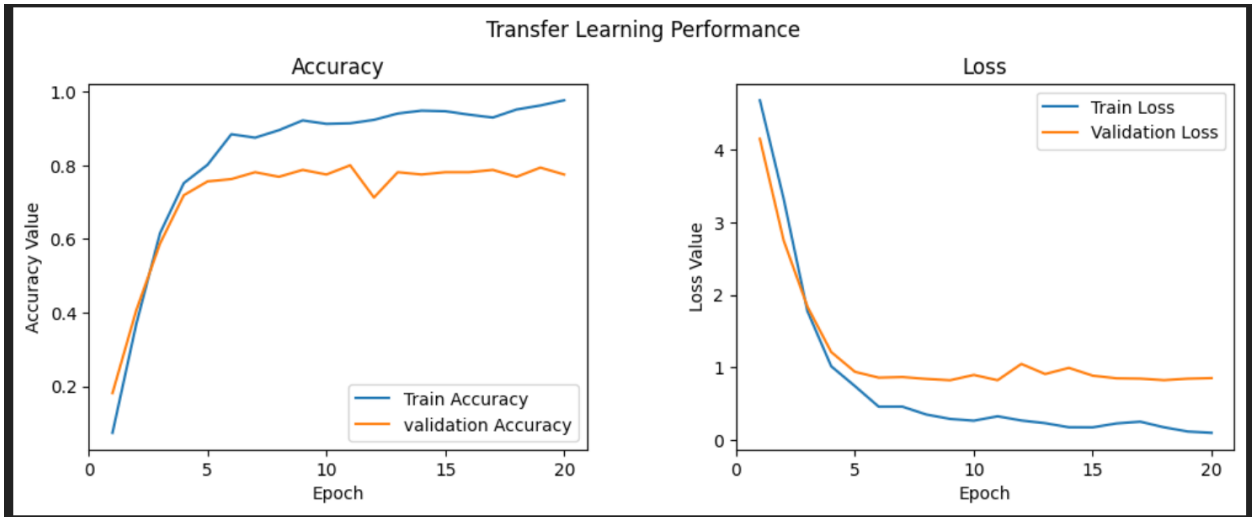
```
model.add(GlobalAveragePooling2D())
model.add(Dropout (0.3))
model.add(Dense(512, activation = 'relu'))
model.add(Dense(512, activation = 'relu'))
model.add(Dense(len(breeds), activation = 'softmax'))


# Freeze pre-trained Layers
print('Number of trainable weights before freezing the base layer:', len
(model.trainable_weights))
model.layers[0].trainable = False
print('Number of trainable weights after freezing the base layer:', len (model.trainable_weights))


app = Flask(__name__)
UPLOAD_FOLDER = 'static/uploads'
app.secret_key = "secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['MAX_CONTENT_LENGTH'] = 16 * 1024 * 1024
ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'gif'}


def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


@app.route('/')
def home():
    return render_template("start.html")


@app.route('/', methods=['POST'])
def upload_image():
    if 'file' not in request.files:
        flash('No file part')
        return redirect(request.url)
    file = request.files['file']
    if file.filename == '':
        flash('No image selected for uploading')
        return redirect(request.url)
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        # print('upload_image filename: ' + filename)
```

```
        fullpaths = [
            'D:\\full-stack\\dl\\static\\uploads\\{}'.format(filename)]
        # print (fullpaths)
        img_data = np.array(
            [img_to_array(load_img(img, target_size=(299, 299)))for img in fullpaths])
        x_test1 = img_data/255.
        # rescale to 0-1. Divide by 255 as its the max rgb value
        from keras import models
        model = models.Sequential()
        test_predictions = model.predict(x_test1)
        # from sklearn.preprocessing import LabelEncoder
        # le = LabelEncoder()
        # le.fit(y)
        predictions = le.classes_[np.argmax(test_predictions, axis=1)]
        # print (predictions[0])
        name = predictions[0].upper().replace("_", "")
        return render_template('result.html', prediction=name,
src="https://simple.wikipedia.org/wiki/" + predictions[0])


if __name__ == "__main__":
    app.run(debug=True)


app = Flask(__name__)
```

OUTPUT :


Actual: Irish_wolfhound
Predict: Scottish_deerhound
Conf: 0.9046341
Actual: black-and-tan_coonhound
Predict: black-and-tan_coonhound
Conf: 0.9581745
Actual: Scotch_terrier
Predict: Scotch_terrier
Conf: 0.555291
Actual: Rottweiler
Predict: Rottweiler
Conf: 0.88580126
Actual: briard
Predict: soft-coated_wheaten_terrier
Conf: 0.7594254
Actual: Airedale
Predict: Airedale
Conf: 0.53459054
Actual: keeshond
Predict: keeshond
Conf: 0.67219824
Actual: Chihuahua
Predict: Chihuahua
Conf: 0.9060459
Actual: Lhasa
Predict: Lhasa
Conf: 0.7857093
Actual: Mexican_hairless
Predict: Mexican_hairless
Conf: 0.99983466
Actual: soft-coated_wheaten_terrier
Predict: soft-coated_wheaten_terrier
Conf: 0.5463869
Actual: standard_schnauzer
Predict: giant_schnauzer
Conf: 0.50111604
Actual: chow
Predict: chow
Conf: 0.9889538
Actual: wire-haired_fox_terrier
Predict: wire-haired_fox_terrier
Conf: 0.65756595
Actual: silky_terrier
Predict: Yorkshire_terrier
Conf: 0.44792306
File | D:/full-stack/dl/templates/start.html
KNOW YOUR DOG
Choose File
No file chosen
Submit





CONCLUSION :

The dog breed identifier project successfully classifies the breed of a given dog image with high accuracy. It has potential applications in the pet industry, dog shelters, and rescue centers. Future improvements to this project could include the ability to identify mixed-breed dogs and to recognize multiple dogs in a single image.