A picture containing text, tableware, plate, dishware

Description automatically generated

**Introduction to Artificial Intelligence**

**Group Members:**

Muhammad Hassaan Raza 22-NTU-CS-1362

Kanza Kashaf 22-NTU-CS-1350

**Program** BSAI

**Semester** 4th

**Department**  Computer Science

**Submitted to** Mr. Waqar Ahmad

**Title:**

Animal Detection System

**Animal Detection System Documentation**

**Introduction**

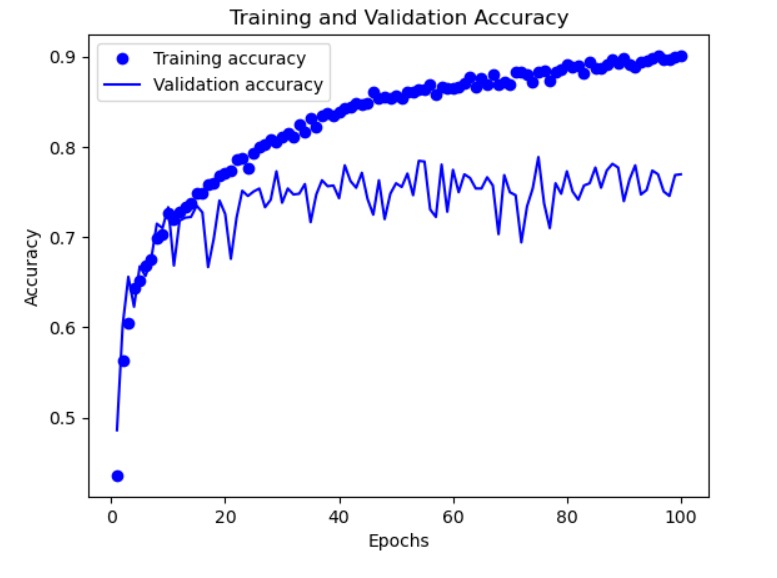
The project aims to develop an animal detection system that classifies images into three categories: cat, dog, and squirrel. The dataset sourced from Kaggle, and is split into training and testing sets based on an 80/20 split. Training set consists of more than 6000 images. And Testing set consists of 1200+ images.

Dataset sources:

* [Animals-10 Dataset](https://www.kaggle.com/datasets/alessiocorrado99/animals10)

The model's performance metrics are as follows:

* Training Data: Accuracy = 93%, Loss = 0.19
* Testing Data: Accuracy = 75%, Loss = 0.88



**Neural Network Code Explanation**

**1. Installing Required Libraries**

First, we install the TensorFlow library, which is essential for building and training our neural network.

python

Copy code

!pip install tensorflow

import numpy as np

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

tf.\_\_version\_\_

**2. Data Preprocessing**

We use ImageDataGenerator for preprocessing the images. This involves rescaling the images and applying random transformations to augment the training data, which helps improve the model's generalization.

python

Copy code

train\_datagen = ImageDataGenerator(

rescale=1./255,

shear\_range=0.2,

zoom\_range=0.2,

horizontal\_flip=True

)

training\_set = train\_datagen.flow\_from\_directory(

'training\_set',

target\_size=(64, 64),

batch\_size=32,

class\_mode='categorical'

)

test\_datagen = ImageDataGenerator(rescale=1./255)

test\_set = test\_datagen.flow\_from\_directory(

'testing\_set',

target\_size=(64, 64),

batch\_size=32,

class\_mode='categorical'

)

**3. Building the Convolutional Neural Network (CNN)**

We create a sequential model and add layers to it:

* **Convolutional Layers**: Extract features from the images using filters.
* **MaxPooling Layers**: Reduce the spatial dimensions of the feature maps.
* **Dropout Layer**: Prevent overfitting by randomly setting input units to 0.
* **Flatten Layer**: Convert the 2D matrices into a 1D vector.
* **Dense Layers**: Perform the final classification.

python

Copy code

cnn = tf.keras.models.Sequential()

cnn.add(tf.keras.layers.Conv2D(filters=64, kernel\_size=3, activation='relu', input\_shape=[64, 64, 3]))

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2, strides=2))

cnn.add(tf.keras.layers.Conv2D(filters=64, kernel\_size=3, activation='relu'))

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2, strides=2))

cnn.add(tf.keras.layers.Dropout(0.5))

cnn.add(tf.keras.layers.Flatten())

cnn.add(tf.keras.layers.Dense(units=128, activation='relu'))

cnn.add(tf.keras.layers.Dense(units=3, activation='softmax'))

**4. Compiling the CNN**

We compile the CNN using the RMSprop optimizer and categorical crossentropy loss function, and we track the accuracy metric.

python

Copy code

cnn.compile(optimizer='rmsprop', loss='categorical\_crossentropy', metrics=['accuracy'])

**5. Training the CNN**

The model is trained for 120 epochs using the training and testing datasets.

python

Copy code

cnn.fit(x=training\_set, validation\_data=test\_set, epochs=120)

**6. Making Predictions**

To predict the class of a new image, we preprocess the image to match the input shape of the model and use the trained model to make a prediction.

python

Copy code

from keras.preprocessing import image

test\_image = image.load\_img('Prediction/ccc.jpeg', target\_size=(64, 64))

test\_image = image.img\_to\_array(test\_image)

test\_image = np.expand\_dims(test\_image, axis=0)

result = cnn.predict(test\_image)

training\_set.class\_indices

if result[0][0] == 1:

print('cat')

elif result[0][1] == 1:

print('dog')

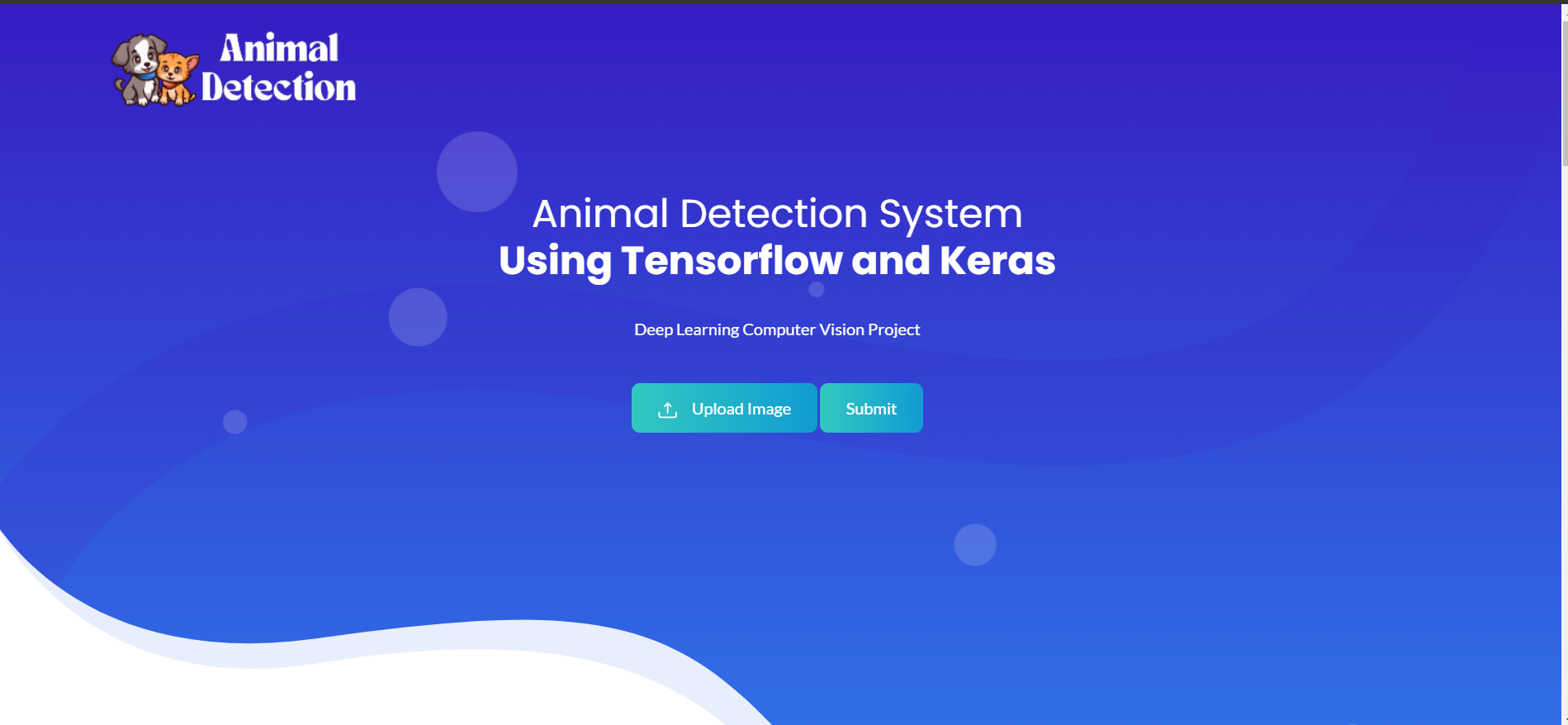
elif result[0][2] == 1:

print('squirrel')

print(result)

Web Interface using Flask:

To provide a user-friendly interface for the animal detection system, we created a web application using Flask. Flask is a micro web framework written in Python.



A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

A graph with blue lines

Description automatically generated A dog and cat on a brick surface

Description automatically generated A screenshot of a computer

Description automatically generated