RA2311047010028.

INDEX

NAME: Pooja STD.: B.Tech AI SEC.: "A" ROLL NO. RA028 SUB.: DLT (LAB)

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1.	24/07/25	Exploring the deeplearningplatform		
2.	31/07/25	Implement a classifier using open source code		
3.	07/08/25	Study of classifier with respect to statistical Parameter.		
4.	22/08/2025	Build a Sample feed forward neural network to recognize handwritten character.		
5.	09/09/2025	Study Of Activation function And its types		
6.	09/09/2025	Implementing gradient descent and back propagation in deep neural network		
7.	23/09/2025	Build a CNN model to classify cat and dog image.		

KC0923147 A

Exp: 07

Lab 7: Build a CNN model to classify cat and dog image.

Aim: To build and train a convolution Neural Network model that can classify images as either cat or Dog with high accuracy.

Objective

1. Define the classification problem
2. Collect and prepare the dataset (Resize, normalise, split)
3. Build a CNN model with Convolutional and dense layer.
4. Compile the model with binary Crossentropy loss and Adam Optimizer
5. Train the model using training data and validate on validation data
6. Evaluate model accuracy on the test dataset
7. Use the model to predict new image
8. Summarise results and discuss improvements.

Date : 23-
Exp: 07

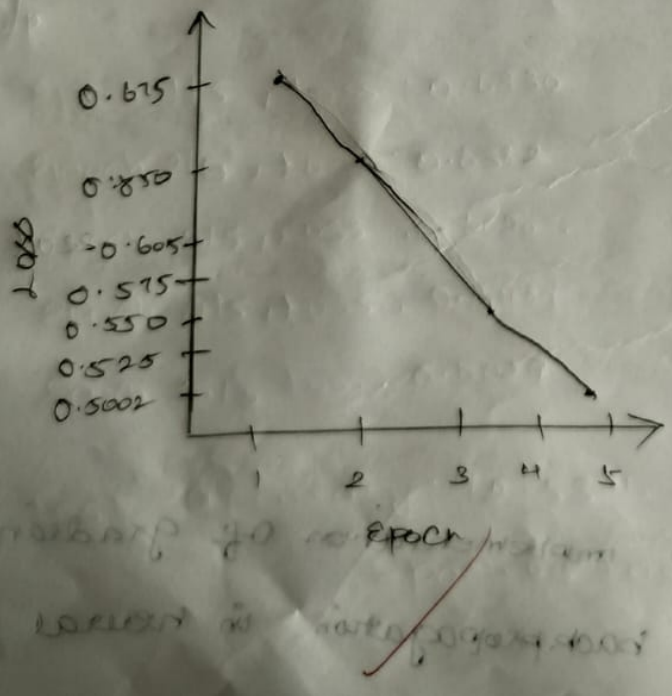
Lab 7:

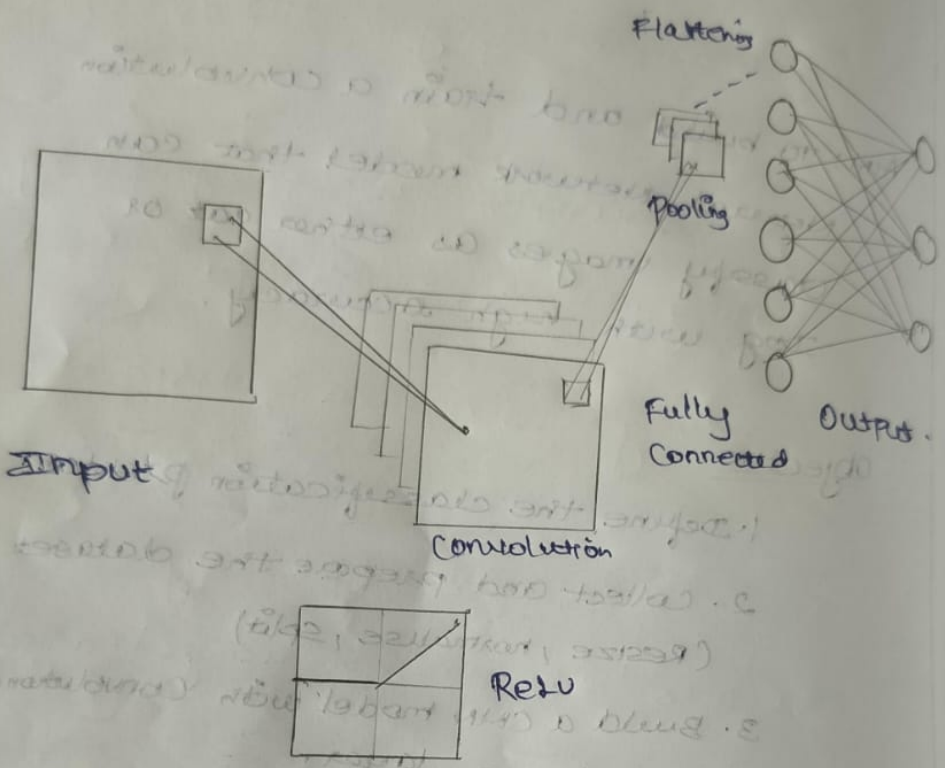
Aim:

Ob

Example	Result Table		
metric	Training set	Validation set	Test set
Accuracy	98.5 %	96.2 %	95.6 %
Loss	0.045	0.125	0.138
Precision	97.1 %	95.5 %	94.9 %
Recall	98.9 %	96.7 %	96.2 %
F1 score	98.0 %	96.0 %	95.5 %

Graph.





O/p

Epoch 1/5 , Loss : 0.6830

Epoch 2/5 , Loss : 0.6362

Epoch 3/5 , Loss : 0.6020

Epoch 4/5 , Loss : 0.5809

Epoch 5/5 Loss : 0.5029

Test accuracy : 72.53 %.

Epochvs Loss.

Pseudo cod

1. Im ce

2. Lo

3. De Pe

4. S

5.

6.

Result

Was

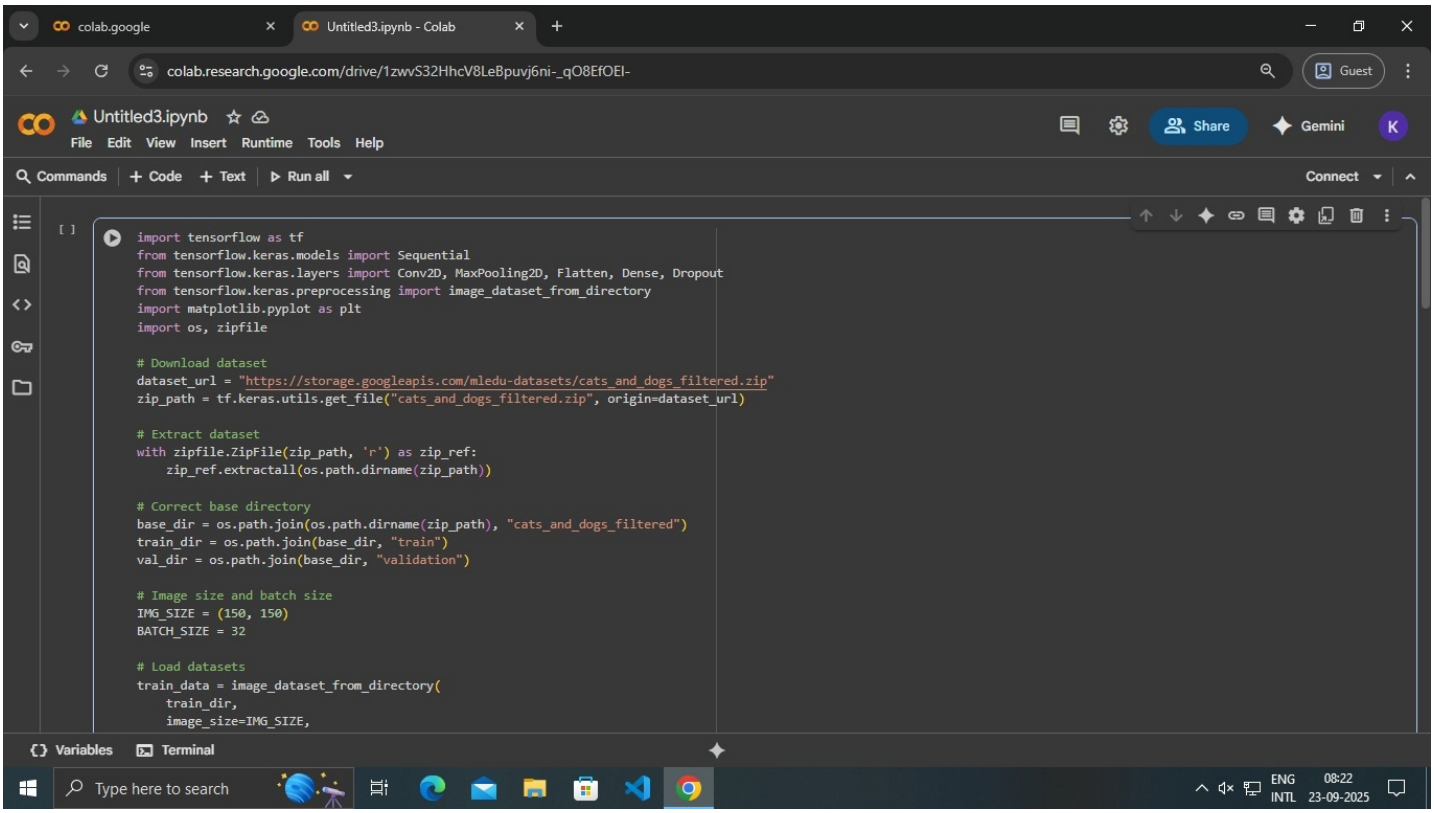
Pseudo code

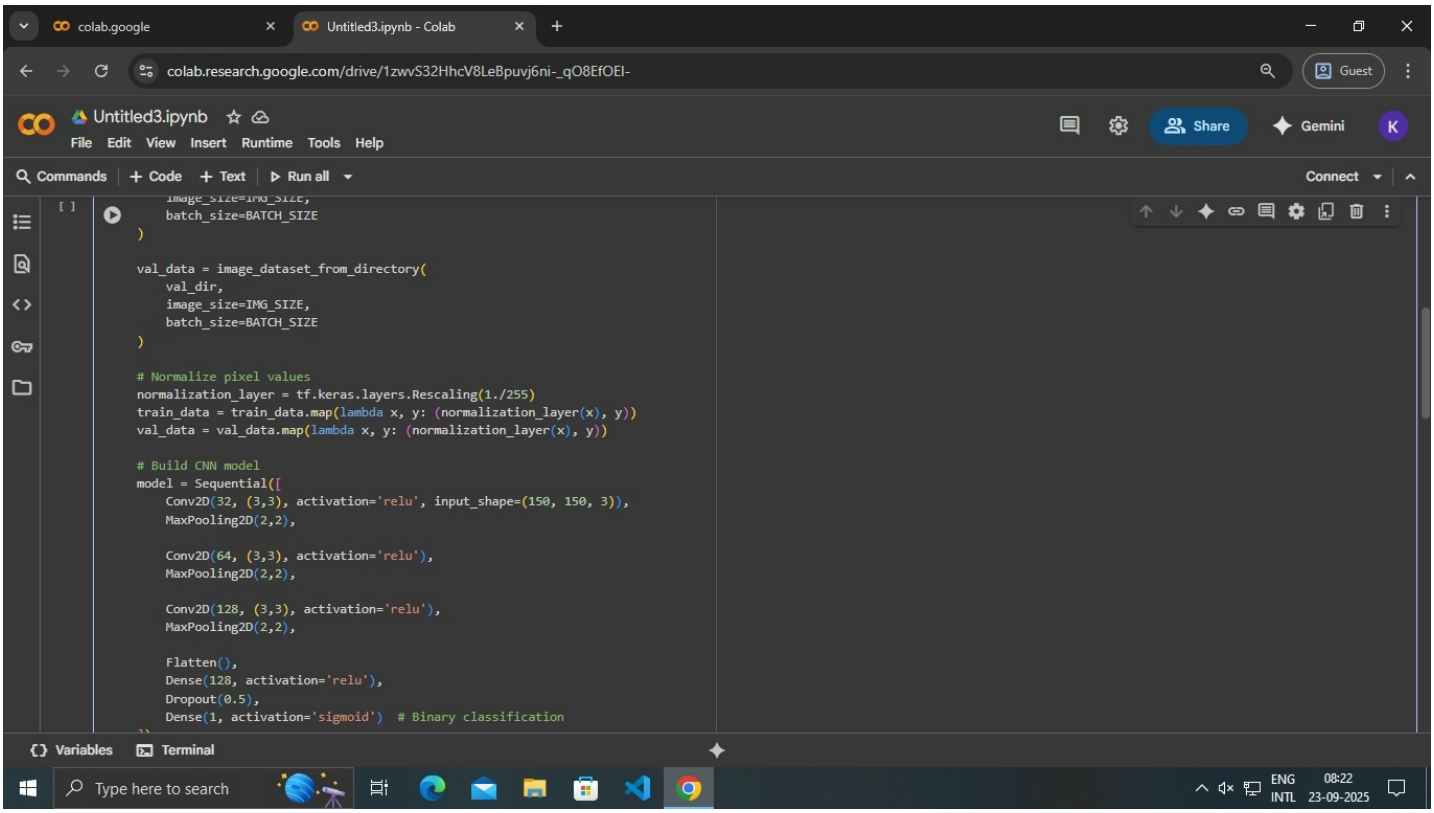
1. Import libraries
   (eg. pytorch and torchvision libraries)
2. Load and preprocess the dataset
3. Define a CNN model with convolutional, pooling, and fully connect layers.
4. Set up loss function and optimizer
5. Train the model over multiple epochs:
   * forward
   * Compute loss
   * Backpropagate and update weights
6. Validate model performance on validation data after each epoch

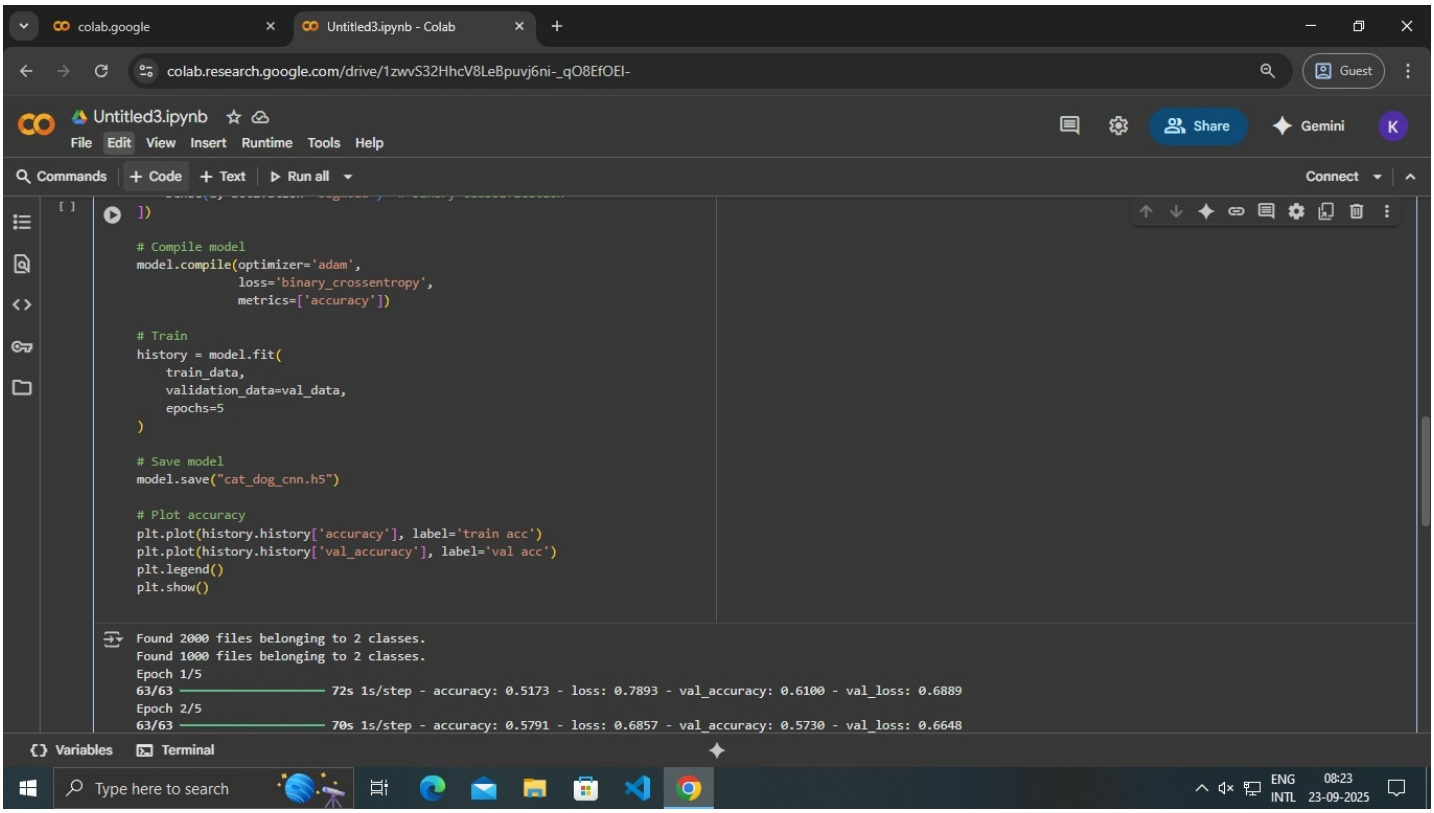
4/12/24

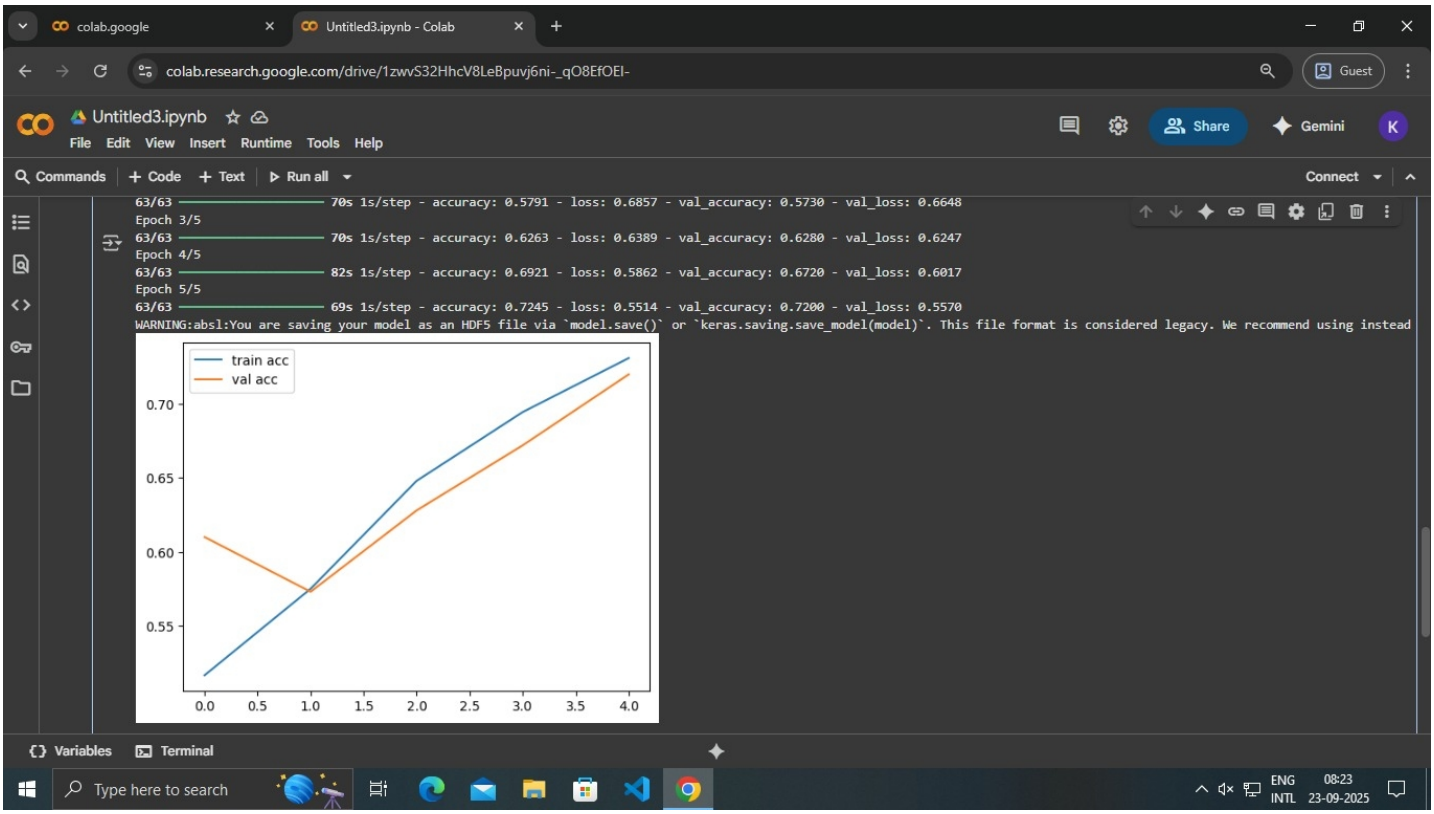
Result

the CNN model built in pytorch was able to successfully cloud cat and dog image.


colab.google
Untitled3.ipynb - Colab
colab.research.google.com/drive/1zwvS32HhcV8LeBpuvj6ni-_qO8EfOEI-
Guest
Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
Connect
[ ]
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.preprocessing import image_dataset_from_directory
import matplotlib.pyplot as plt
import os, zipfile
# Download dataset
dataset_url = "https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip"
zip_path = tf.keras.utils.get_file("cats_and_dogs_filtered.zip", origin=dataset_url)
# Extract dataset
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
zip_ref.extractall(os.path.dirname(zip_path))
# Correct base directory
base_dir = os.path.join(os.path.dirname(zip_path), "cats_and_dogs_filtered")
train_dir = os.path.join(base_dir, "train")
val_dir = os.path.join(base_dir, "validation")
# Image size and batch size
IMG_SIZE = (150, 150)
BATCH_SIZE = 32
# Load datasets
train_data = image_dataset_from_directory(
train_dir,
image_size=IMG_SIZE,
Variables
Terminal
Type here to search
ENG
INTL
08:22
23-09-2025


colab.google
Untitled3.ipynb - Colab
colab.research.google.com/drive/1zwvS32HhcV8LeBpuvj6ni-_qO8EfOEI-
Guest
Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
Connect
batch_size=BATCH_SIZE
)
val_data = image_dataset_from_directory(
val_dir,
image_size=IMG_SIZE,
batch_size=BATCH_SIZE
)
# Normalize pixel values
normalization_layer = tf.keras.layers.Rescaling(1./255)
train_data = train_data.map(lambda x, y: (normalization_layer(x), y))
val_data = val_data.map(lambda x, y: (normalization_layer(x), y))
# Build CNN model
model = Sequential([
Conv2D(32, (3,3), activation='relu', input_shape=(150, 150, 3)),
MaxPooling2D(2,2),
Conv2D(64, (3,3), activation='relu'),
MaxPooling2D(2,2),
Conv2D(128, (3,3), activation='relu'),
MaxPooling2D(2,2),
Flatten(),
Dense(128, activation='relu'),
Dropout(0.5),
Dense(1, activation='sigmoid') # Binary classification
Variables
Terminal
Type here to search
ENG
INTL
08:22
23-09-2025


colab.google
Untitled3.ipynb - Colab
colab.research.google.com/drive/1zwvS32HhcV8LeBpuvj6ni-_qO8EfOEI-
Guest
Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
Connect
[ ]
])
Compile model
model.compile(optimizer='adam',
loss='binary_crossentropy',
metrics=['accuracy'])
# Train
history = model.fit(
train_data,
validation_data=val_data,
epochs=5
)
# Save model
model.save("cat_dog_cnn.h5")
# Plot accuracy
plt.plot(history.history['accuracy'], label='train acc')
plt.plot(history.history['val_accuracy'], label='val acc')
plt.legend()
plt.show()
Found 2000 files belonging to 2 classes.
Found 1000 files belonging to 2 classes.
Epoch 1/5
63/63 72s 1s/step - accuracy: 0.5173 - loss: 0.7893 - val_accuracy: 0.6100 - val_loss: 0.6889
Epoch 2/5
63/63 70s 1s/step - accuracy: 0.5791 - loss: 0.6857 - val_accuracy: 0.5730 - val_loss: 0.6648
Variables
Terminal
Type here to search
ENG
INTL
08:23
23-09-2025


colab.google
Untitled3.ipynb - Colab
colab.research.google.com/drive/1zwvS32HhcV8LeBpuvj6ni-_qO8EfOEI-
Guest
Untitled3.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands
+ Code
+ Text
Run all
Connect
63/63 70s 1s/step - accuracy: 0.5791 - loss: 0.6857 - val_accuracy: 0.5730 - val_loss: 0.6648
Epoch 3/5
63/63 70s 1s/step - accuracy: 0.6263 - loss: 0.6389 - val_accuracy: 0.6280 - val_loss: 0.6247
Epoch 4/5
63/63 82s 1s/step - accuracy: 0.6921 - loss: 0.5862 - val_accuracy: 0.6720 - val_loss: 0.6017
Epoch 5/5
63/63 69s 1s/step - accuracy: 0.7245 - loss: 0.5514 - val_accuracy: 0.7200 - val_loss: 0.5570
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead
train acc
val acc
0.70
0.65
0.60
0.55
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Variables
Terminal
Type here to search
ENG
INTL
08:23
23-09-2025