TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN HỌC PHẦN

Môn: HỌC SÂU

XÂY DỰNG PHẦN MỀM NHẬN DẠNG 5 ĐỐI TƯỢNG TRONG PHÒNG

GIẢNG VIÊN HƯỚNG DẪN: ThS. Tôn Quang Toại

SINH VIÊN THỰC HIỆN:

Đinh Phú Cường – 20DH111722

Đồng Đăng Quang – 20DH110560

Nguyễn Hoàng Gia Bảo – 20DH111925

Dương Thái Bảo – 20DH110422

TP. HỒ CHÍ MINH – 12/2023

PHIẾU CHẤM ĐIỂM

Họ tên sinh viên 1 (SV1): Đồng Đăng Quang _____ Mã SV: 20DH110560

Họ tên sinh viên 2 (SV2): Nguyễn Hoàng Gia Bảo___ Mã SV: 20DH111925

Họ tên sinh viên 3 (SV3): Dương Thái Bảo_____ Mã SV: 20DH110422

Họ tên sinh viên 4 (SV4): Đinh Phú Cường _____ Mã SV: 20DH111722

CLO	Nội dung/Chuẩn đầu ra	ĐIỂM CỦA SV 1	ĐIỂM CỦA SV 2
1	Xây dựng và huấn luyện mạng neuron (Bài toán phân lớp ảnh, huấn luyện mô hình phân lớp…)		
2	Tinh chỉnh mô hình và thuật toán huấn luyện (Bài toán phân đoạn, tìm gốc quay, phát hiện vị trí, …)		
3	Vận dụng mạng neuron tích chập và mạng hồi quy (Triển khai mô hình trên web, desktop, mobile, …)		
4	Có khả năng giải quyết một số vấn đề thực tế (Thu thập dữ liệu, xử lý dữ liệu, …)		
5	Có năng lực trình bày giải pháp kỹ thuật (Thuyết trình, trình bày báo cáo, …)		
	Tổng		

Họ tên GV 1: _____ Ký tên: _____

Họ tên GV 2: _____ Ký tên: _____

TÓM TẮT ĐỒ ÁN

- Bài toán (problem): Tạo ra mô hình nhận biết các vật dụng trong nhà. Phân biệt được giữa các vật thể được chọn ngẫu nhiên trong tập dữ liệu đã được chọn lọc. Các vật thể bao gồm bàn, giày boot, laptop, ghế, sofa. Ngoài ra mô hình còn có thể tìm được góc quay của một vật thể nhất định.
- Phân lớp các đối tượng
- Sử dụng mô hình web để hiển thị góc quay và nhận diện của hình
- Kết quả (result): Có thể nhận biết được các vật thể từ tập dữ liệu và có thể tính được góc quay của vật

Mục lục

Chương 1. Giới thiệu bài toán

1.1 Câu hỏi nghiên cứu

- Nội dung của bài toán: Chọn lọc 1000 hình ảnh của các đồ vật trong nhà. Tiến hành tạo ra một mô hình máy học để có thể nhận biết các đồ vật có trong từng hình ảnh riêng biệt. Ngoài ra sẽ có thêm một mô hình để có thể tính góc quay của vật thể nhất định.
- Mô tả Input của bài toán: Đưa vào tập dữ liệu chứa 1000 ảnh của các đồ vật trong nhà.
- Ngoài ra còn tính góc quay của vật.
- Mô tả Output của bài toán: Trả về các hình ảnh tương ứng với tên đồ vật có trong hình. Tìm ra được góc độ của đồ vật.
  - Ví dụ: Trong hình có một cái laptop sau khi chạy qua mô hình sẽ trả về hỉnh ảnh laptop kèm theo dòng chữ "laptop" trên đầu của bức ảnh.
- Dataset được dùng ở đây gồm 1000 tấm ảnh của các vật dụng trong nhà được góp nhặt từ nhiều nguồn khác nhau

1.2 Giới hạn nghiên cứu

- Giới hạn: Có thể sẽ không tìm ra đúng góc quay của đồ vật. Đây vẫn chưa phải là mô hình tối ưu nhất.

1.3 Bố cục đồ án

Đồ án này được chia thành một số chương và mục để tạo ra một cấu trúc rõ ràng và logic. Tổng cộng, đồ án này bao gồm bốn chương và năm mục, mỗi phần đều đóng vai trò quan trọng trong việc trình bày và phân tích nội dung. Mỗi chương và mục đều được thiết kế để phản ánh sự sắp xếp logic của thông tin, từ giới thiệu đến kết luận, giúp người đọc dễ dàng theo dõi và hiểu rõ về nội dung của đồ án. Các chương và mục được tổ chức hợp lý để tạo nên một bài báo cáo toàn diện về chủ đề được nghiên cứu, đồng thời thể hiện sự chi tiết và sâu sắc trong phân tích và thảo luận. Chương 1 của đồ án tập trung vào việc giới thiệu bài toán, mô tả vấn đề cụ thể mà đồ án đang nghiên cứu. Nó cung cấp một cái nhìn tổng quan về ngữ cảnh và ý nghĩa của việc giải quyết bài toán, đặt ra những câu hỏi nghiên cứu và định rõ mục tiêu của nghiên cứu. Chương 2 trình bày giải pháp đề xuất để phân tích dữ liệu bài toán đã được đặt ra. Nó chi tiết hóa các phương pháp, mô hình, hoặc công nghệ được sử dụng để đạt được mục tiêu. Chương này là nơi để đề cập đến các kiến thức lý thuyết, các nghiên cứu liên quan, và mô tả chi tiết về cách giải pháp được xây dựng. Chương 3 tập trung vào quá trình thực nghiệm, mô phỏng, hoặc thử nghiệm giải pháp đề xuất. Nó cung cấp thông tin về việc triển khai và kiểm thử giải pháp trong môi trường thực tế, đồng thời đánh giá hiệu suất và độ chính xác của nó. Cuối cùng, Chương 4 tập trung vào việc tìm hiểu và phân tích góc quay của đồ án. Nó có thể bao gồm những xem xét kết quả, nhận xét về khả năng áp dụng của giải pháp trong các tình huống khác nhau, và đề xuất hướng phát triển hoặc nghiên cứu tiếp theo. Phần kết luận sẽ giúp công trình nghiên cứu và làm rõ ý nghĩa của kết quả đạt được.

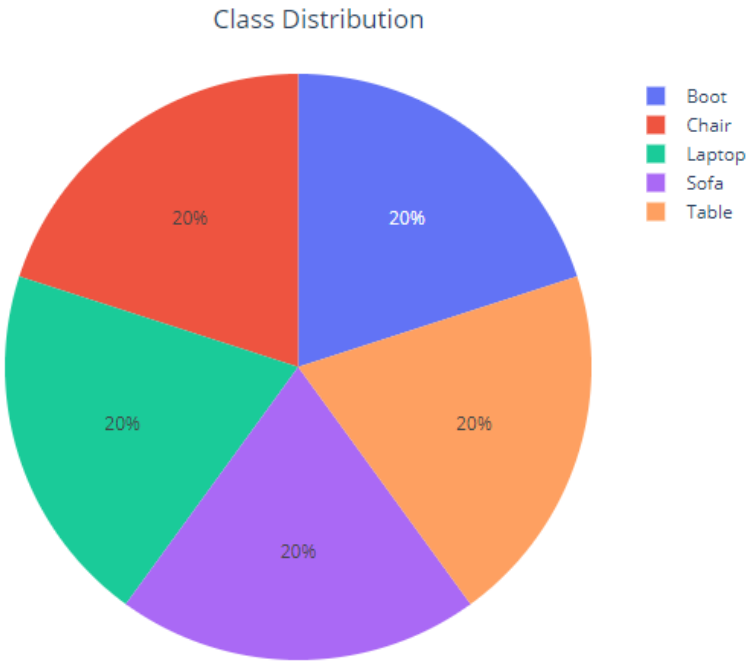
Chương 2. Giải pháp đề xuất

2.1 Phân tích dữ liệu

Khi bắt đầu thu nhập dữ liệu đây chỉ là những hình ảnh riêng lẻ không liên quan đến nhau. Tiến hành bước tiền xử lý dữ liệu, tiến hành lọc và chia dữ liệu thành 5 class khác nhau tương ứng cho từng vật thể. Chia đều tập data thành 5 class (như hình minh họa).

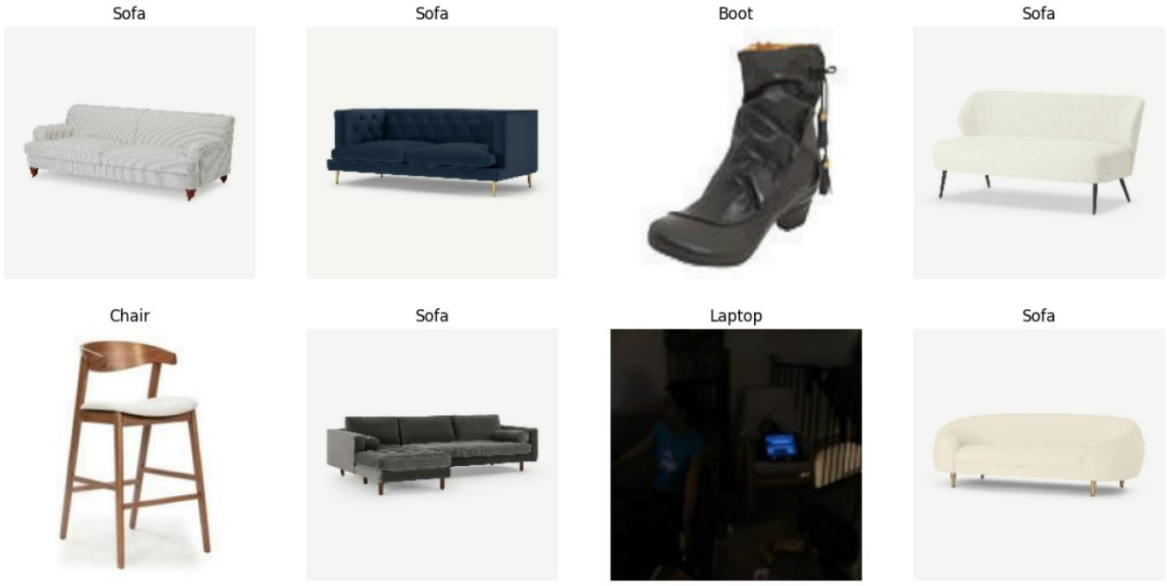
```
class_dis = [len(os.listdir('/content/data' + f"/{name}")) for name in class_names]
lk_dis = dict(zip(class_names, class_dis))
print(f"Class Distribution : \n{lk_dis}")

fig = px.pie(names=class_names, values=class_dis, width=600)
fig.update_layout({"title":{'text':"Class Distribution","x":0.5}})
fig.show()
```

```
Class Distribution :
{'Boot': 1000, 'Chair': 1000, 'Laptop': 1000, 'Sofa': 1000, 'Table': 1000}
```



- Tiến hành xuất ra một vài ví dụ

```
dataset = image_dataset_from_directory(
  path_to_data,
  image_size=(224, 224),
  batch_size=64,
  shuffle=True)

def show_images(dataset):
  plt.figure(figsize=(16,16))
  for images, labels in dataset.take(1):
    for i in range(16):
      ax = plt.subplot(4, 4, i + 1)
      ax.imshow(images[i].numpy().astype("uint8"))
      ax.set_title(dataset.class_names[labels[i]])
      ax.axis("off")
  plt.title("Samples of object")
  plt.show()

show_images(dataset)
```



2.2 Mô hình

- Chia tập dữ liệu với 70% train và 30% test

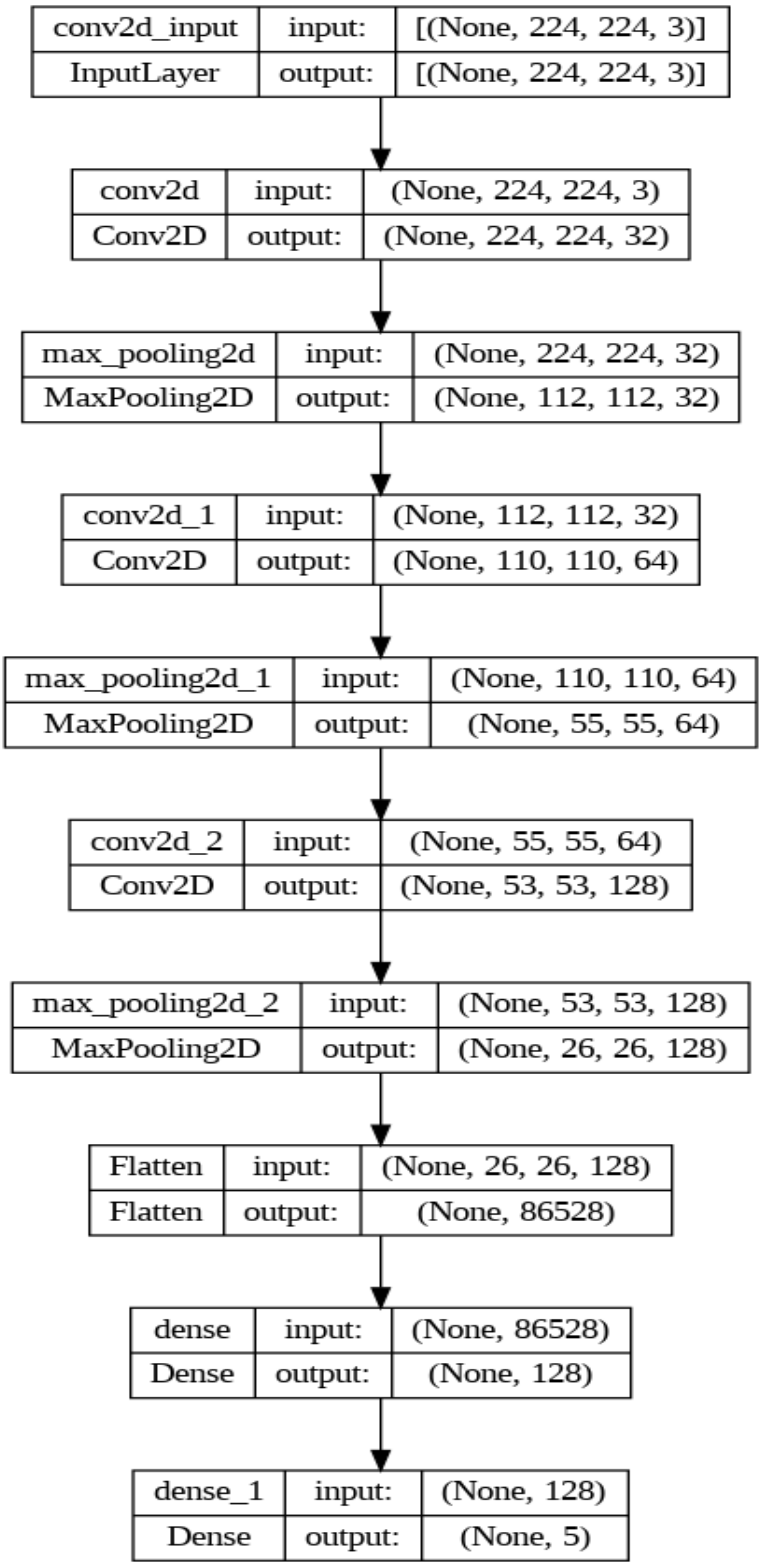
```
data_directory = './data/'

data = []
labels = []

# Lặp qua từng thư mục con trong thư mục data
for label in os.listdir(data_directory):
    label_directory = os.path.join(data_directory, label)
    if os.path.isdir(label_directory):
        # Lặp qua từng tệp tin trong thư mục con và thu thập dữ liệu
        for file in os.listdir(label_directory)[:1000]:
            file_path = os.path.join(label_directory, file)
            data.append(file_path)
            labels.append(label)

# Chia dữ liệu thành tập huấn luyện và kiểm tra
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.3, random_state=42)
```

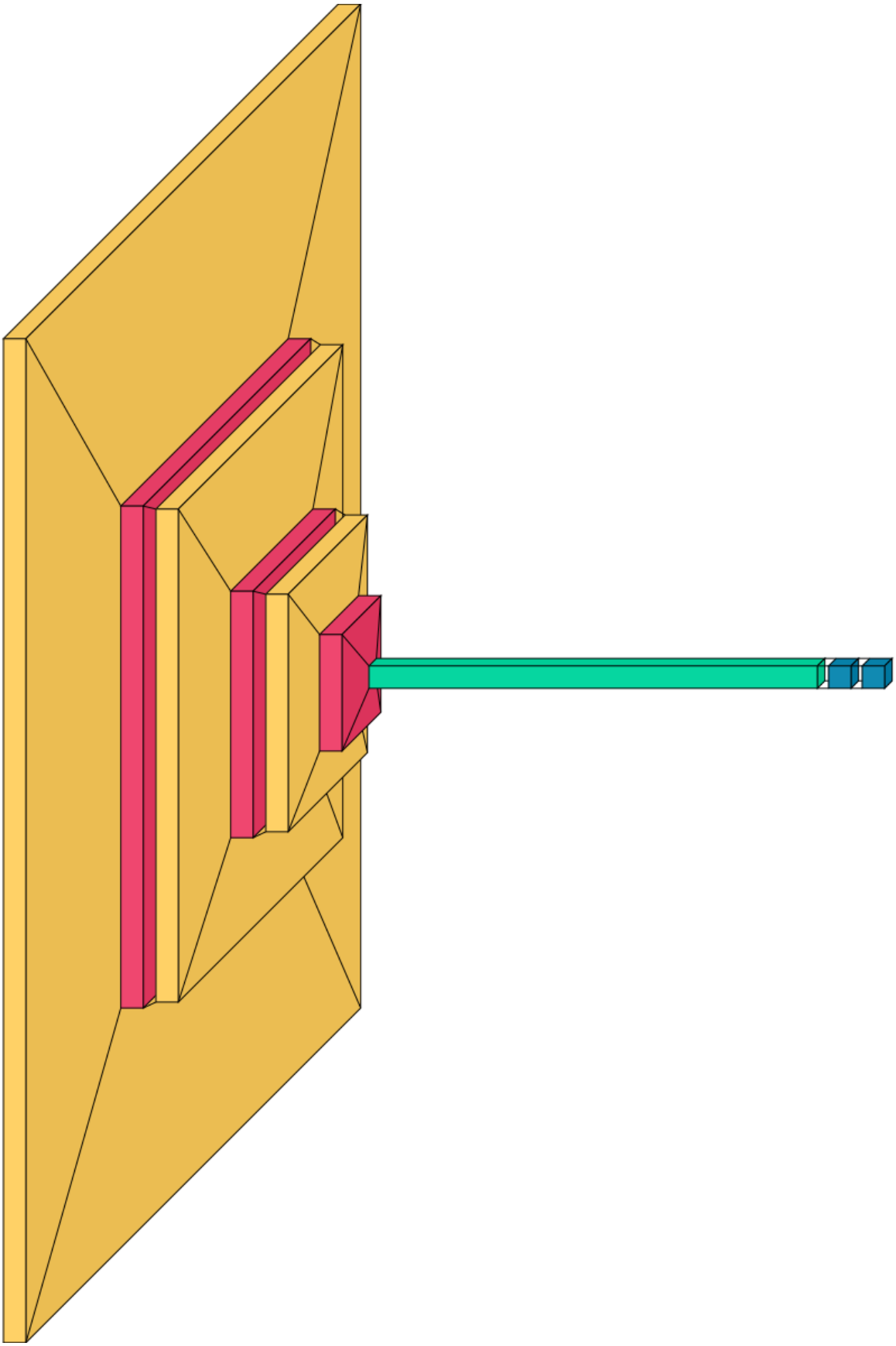
- Tiến hành tạo và huấn luyện mô hình

```
def five_oject():
    model = Sequential()
    model.add(Conv2D(filters=32, kernel_size=(3, 3), padding='same', activation='relu', input_shape=(224, 224, 3)))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=2))

    model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='valid', activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=2))

    model.add(Conv2D(filters=128, kernel_size=(3, 3), padding='valid', activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=2))

    model.add(Flatten(name='Flatten'))

    model.add(Dense(128, activation='relu'))
    model.add(Dense(5, activation='softmax'))
    return model
```

```
# Hàm tiền xử lý ảnh
def preprocess_image(image_path):
    img = load_img(image_path, target_size=(224, 224))
    img_array = img_to_array(img)
    return img_array

# Tiền xử lý dữ liệu ảnh trong X_train và X_test
X_train_processed = np.array([preprocess_image(img_path) for img_path in X_train])
X_test_processed = np.array([preprocess_image(img_path) for img_path in X_test])

label_encoder = LabelEncoder()
y_train_encoded = to_categorical(label_encoder.fit_transform(y_train))
y_test_encoded = to_categorical(label_encoder.transform(y_test))

# Tạo model và huấn luyện
model = five_oject()
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

history = model.fit(X_train_processed, y_train_encoded, epochs=100, validation_data=(X_test_processed, y_test_encoded))
```

- Trực quan hoá mô hình



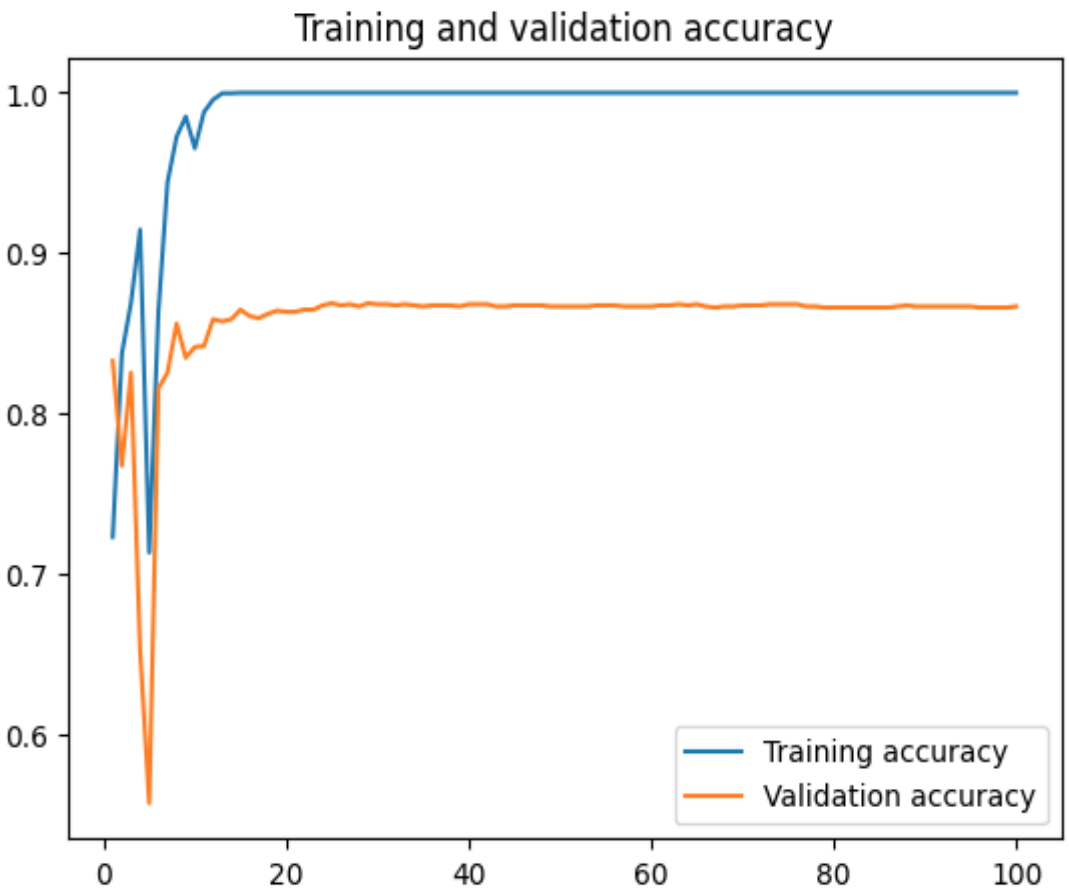


Chương 3. KẾT QUẢ

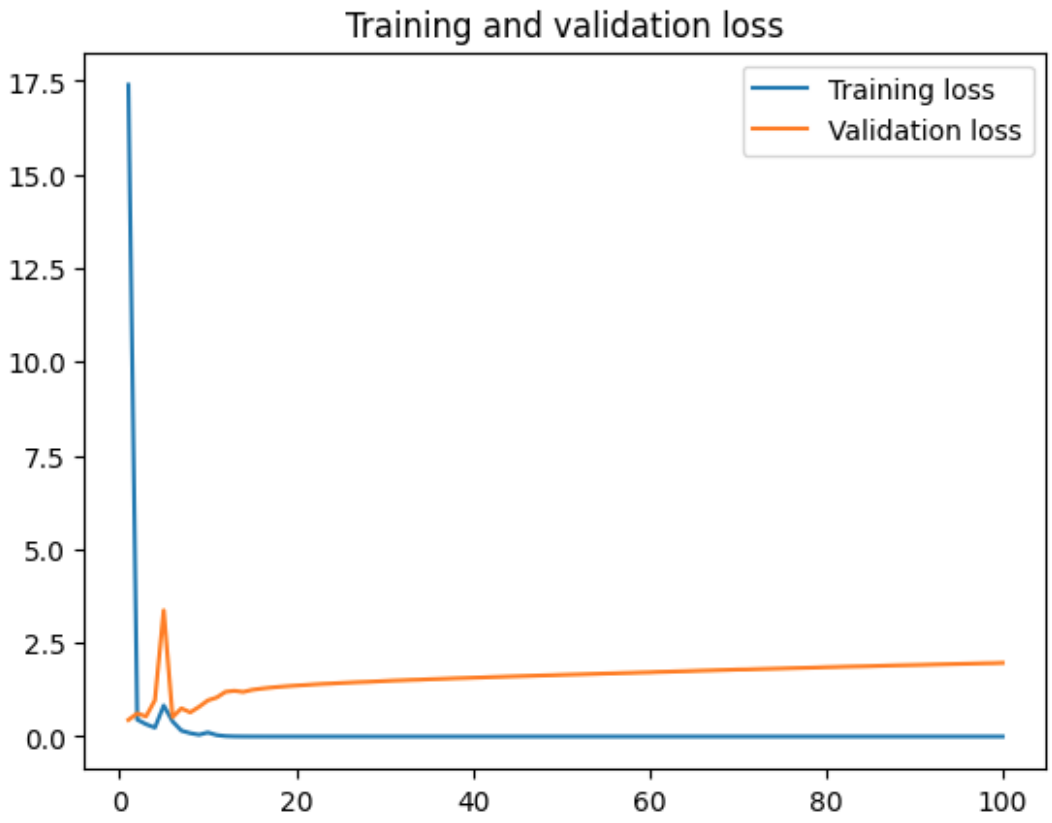
- Tiến hành trực quan hóa kết quả sau khi đã chạy qua model train

```
# Vẽ biểu đồ huấn luyện
accuracy = history.history["accuracy"]
val_accuracy = history.history["val_accuracy"]
loss = history.history["loss"]
val_loss = history.history["val_loss"]
epochs = range(1, len(accuracy) + 1)
plt.plot(epochs, accuracy, label="Training accuracy")
plt.plot(epochs, val_accuracy, label="Validation accuracy")
plt.title("Training and validation accuracy")
plt.legend()

plt.figure()
plt.plot(epochs, loss, label="Training loss")
plt.plot(epochs, val_loss, label="Validation loss")
plt.title("Training and validation loss")
plt.legend()
plt.show()
```

- Tiến hành so sánh tập train với tập validation



- Tiến hành so sánh tập train với tập validation loss



- Tiến hành kiểm tra độ chính xác của mô hình

```
# Kiểm tra mô hình
test_model = load_model("/content/drive/MyDrive/DeepLearning/Final/five_object.h5")
test_loss, test_acc = test_model.evaluate(X_test_processed, y_test_encoded)
print(f"Test accuracy: {test_acc:.3f}")
```

```
47/47 [==============================] - 1s 22ms/step - loss: 0.4020 - accuracy: 0.8613
Test accuracy: 0.861
```

- Train mô hình VGG16:

```
# Tạo mô hình VGG16
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Freeze convolutional layers
for layer in base_model.layers:
    layer.trainable = False

# Thêm top layers
model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dense(1024, activation='relu'),
    Dense(num_classes, activation='softmax')
])

# Compile mô hình
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Huấn luyện mô hình
model.fit(train_generator, epochs=5, validation_data=val_generator)
```

- Train mô hình ResNet

```
# Tạo mô hình ResNet50
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Freeze convolutional layers
for layer in base_model.layers:
    layer.trainable = False

# Thêm top layers
model_resnet50 = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dense(1024, activation='relu'),
    Dense(len(labels), activation='softmax')
])

# Compile mô hình
model_resnet50.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Huấn luyện mô hình
model_resnet50.fit(train_generator, epochs=5, validation_data=val_generator)
```

- Lập bảng so sánh các mô hình đã huấn luyện

```
# Tạo dữ liệu
data = {
    'Name': ['model','VGG16', 'RESNET50'],
    'Test accuracy': [0.845,eval_result_vgg16[1], eval_result_resnet50[1]]
}

# Tạo DataFrame từ dữ liệu và không hiển thị số thứ tự
df = pd.DataFrame(data)
df.index = [''] * len(df)  # Đặt index thành một list rỗng
print(df)
```

```
      Name  Test accuracy
     model          0.845
     VGG16          0.937
  RESNET50          0.707
```

Nhận xét:

- VGG16 có accuracy cao nhất: 0.937
- RESNET50 có accuracy thấp nhất: 0.707
- Model tự viết có accuracy khá cao vượt hơn RESNET50

Chương 4. Tìm góc quay của vật

- Tiến hành tải lên tập dữ liệu

```
#https://drive.google.com/file/d/1Br-yENtHdpWwTQ-xjewhAI1NoDfq8w78/view?usp=sharing
%%shell
pip -q install --upgrade --no-cache-dir gdown
gdown -q 1Br-yENtHdpWwTQ-xjewhAI1NoDfq8w78
echo 'download: done => file: data.zip'
rm -rf ./data/
unzip -q data.zip
echo 'Data files in: data/'
```

- Tiến hành xoay ảnh trong tập dữ liệu để làm ví dụ cho bài toán

```
import cv2
from google.colab.patches import cv2_imshow
# Đọc ảnh
image = cv2.imread("/content/data/test/Chair/130d117b2688f64a13ad30adcc7fad.jpg")

# Xoay ảnh 90 độ
rotated_90 = cv2.rotate(image, cv2.ROTATE_90_CLOCKWISE)
rotated_90 = cv2.cvtColor(rotated_90, cv2.COLOR_BGR2GRAY)
# Xoay ảnh 180 độ
rotated_180 = cv2.rotate(image, cv2.ROTATE_180)
rotated_180 = cv2.cvtColor(rotated_180, cv2.COLOR_BGR2GRAY)
# Xoay ảnh 270 độ
rotated_270 = cv2.rotate(image, cv2.ROTATE_90_COUNTERCLOCKWISE)
rotated_270 = cv2.cvtColor(rotated_270, cv2.COLOR_BGR2GRAY)
# Hiển thị ảnh gốc
print("Original Image")
cv2_imshow(image)

# Hiển thị ảnh xoay 90 độ
print("Rotated 90 Degrees")
cv2_imshow(rotated_90)

# Hiển thị ảnh xoay 180 độ
print("Rotated 180 Degrees")
cv2_imshow(rotated_180)

# Hiển thị ảnh xoay 270 độ
print("Rotated 270 Degrees")
cv2_imshow(rotated_270)
```

```
# Chờ người dùng nhấn phím bất kỳ để đóng cửa sổ
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Rotated 90 Degrees

Rotated 180 Degrees

Rotated 270 Degrees

- Thuật toán được sử dụng để nhận dạng: +ORB(Oriented FAST and Rotated BRIEF)

```
def tim_goc_quay(anh_goc, anh_quay):
    # Phát hiện key points và tính toán descriptors bằng thuật toán ORB
    orb = cv2.ORB_create()
    key_points_goc, des_goc = orb.detectAndCompute(anh_goc, None)
    key_points_quay, des_quay = orb.detectAndCompute(anh_quay, None)

    # Sử dụng BFMatcher để tìm các điểm tương đồng
    bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    matches = bf.match(des_goc, des_quay)

    # Sắp xếp các điểm dựa trên khoảng cách
    matches = sorted(matches, key=lambda x: x.distance)

    # Lấy 10 điểm tốt nhất
    good_matches = matches[:100]

    # Tính ma trận biến đổi (homography) dựa trên điểm tương đồng
    src_pts = np.float32([key_points_goc[m.queryIdx].pt for m in good_matches]).reshape(-1, 1, 2)
    dst_pts = np.float32([key_points_quay[m.trainIdx].pt for m in good_matches]).reshape(-1, 1, 2)
    ma_tran_bien_doi, _ = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)

    # Tính góc quay
    goc_quay_rad = np.arctan2(ma_tran_bien_doi[0, 1], ma_tran_bien_doi[0, 0])
    goc_quay_deg = np.degrees(goc_quay_rad)
    goc_quay_deg = (goc_quay_deg + 360) % 360
    cv2_imshow(anh_goc)
    cv2_imshow(anh_quay)
    return goc_quay_deg

# Tính góc quay
goc_quay = tim_goc_quay(image, rotated_270)

print(f"Góc quay của ảnh: {goc_quay} độ")
```

KẾT LUẬN

Những điều đã làm được

- Đã tạo ra được model train và nhận biết các đồ vật có trong mô hình.
- Tìm hiểu được thêm nhiều thuật toán để giải quyết bài toán.
- Áp dụng việc rút trích đặc trưng để giải quyết bài toán tìm góc quay của vật.
- Giải quyết được cả 2 bài toán đề ra.

Những điều chưa làm được

- Có thể các model vẫn chưa phải là model tối ưu nhất để giải quyết bài toán.

Hướng phát triển

Phát triển về thiên mô hình: Hướng tiếp cận thiên mô hình là một xu hướng nghiên cứu động lực và phát triển mạnh mẽ trong lĩnh vực khoa học dữ liệu và máy học. Thiên mô hình đặt trọng tâm vào việc tích hợp kiến thức chuyên sâu về đối tượng nghiên cứu vào các mô hình máy học, nhằm tối ưu hóa hiểu biết và dự đoán về các hiện tượng tự nhiên. Thay vì dựa hoàn toàn vào dữ liệu thống kê, thiên mô hình kết hợp tri thức chuyên gia và sự hiểu biết sâu rộng về lĩnh vực cụ thể. Điều này giúp mô hình không chỉ là một công cụ dự đoán, mà còn là một phương tiện để hiểu rõ hơn về các quy luật và nguyên lý tự nhiên. Điều này có thể bao gồm việc tích hợp kiến thức từ các lĩnh vực như vật lý, sinh học, hoặc kinh tế vào mô hình máy học. Hướng tiếp cận này thường được áp dụng trong các lĩnh vực nghiên cứu yêu cầu sự hiểu biết sâu sắc về hệ thống phức tạp, nơi mà các mô hình truyền thống dựa trên dữ liệu có thể không đạt được kết quả chính xác và diễn giải. Thiên mô hình mang lại sự minh bạch và giải thích có giá trị, giúp các chuyên gia đưa ra quyết định thông tin và đồng thời cung cấp một cách tiếp cận độc đáo để nghiên cứu và khám phá trong các lĩnh vực phức tạp.

TÀI LIỆU THAM KHẢO

[1] VBD, "Bạn đã biết các thuật toán phát hiện và đối sánh đặc trưng của ảnh chưa?," 22 September 2022. [Online]. Available: https://vinbigdata.com/kham-pha/ban-da-biet-cac-thuat-toan-phat-hien-va-doi-sanh-dac-trung-cua-anh-chua.html?fbclid=IwAR3qSMDvdVKNl9ya9a8RGmr9PohHnD3-QRxtqb7g4HQyqD3zHCZZ3oycmD8.