TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KÌ MÔN

NHẬP MÔN XỬ LÝ ẢNH SỐ

Nhận diện ảnh chụp thẻ sinh viên

Người hướng dẫn: **TS TRỊNH HÙNG CƯỜNG**

Người thực hiện: **VÕ NGUYỄN ANH KHOA – 52100049**

ĐINH HOÀNG PHÚC – 52100087

Lớp : **21050201**

Khoá : **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KÌ MÔN

NHẬP MÔN XỬ LÝ ẢNH SỐ

Nhận diện ảnh chụp thẻ sinh viên

Người hướng dẫn: **TS TRỊNH HÙNG CƯỜNG**

Người thực hiện: **VÕ NGUYỄN ANH KHOA – 52100049**

ĐINH HOÀNG PHÚC – 52100087

Lớp : **21050201**

Khoá : **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành đến thầy Trịnh Hùng Cường đã giúp đỡ em hoàn thành đồ án cuối kì này. Em cũng rất biết ơn sự chỉ dẫn, hỗ trợ của các thầy cô giảng viên khoa Công Nghệ Thông Tin trong suốt quá trình thực hiện đồ án.

Qua môn học này, em đã học được rất nhiều kiến thức bổ ích. Tuy nhiên, do vốn kiến thức còn hạn chế và khả năng tiếp thu trong thực tế còn nhiều bỡ ngỡ, nên bài đồ án cuối kì của em vẫn còn nhiều sai sót, mong cô bỏ qua và góp ý cho chúng em để bài báo cáo của em hoàn thiện hơn. Chúng em xin chân thành cảm ơn!

ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm đồ án của chúng tôi và được sự hướng dẫn của TS Trịnh Hùng Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 1 tháng 1 năm 2023

Tác giả

(ký tên và ghi rõ họ tên)

Võ Nguyễn Anh Khoa

Đinh Hoàng Phúc

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

TÓM TẮT

Trình bày tóm tắt vấn đề nghiên cứu, các hướng tiếp cận, cách giải quyết vấn đề và một số kết quả đạt được, những phát hiện cơ bản trong vòng 1 -2 trang.

MỤC LỤC

DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

CÁC KÝ HIỆU

f *Tần số của dòng điện và điện áp (Hz)*

p *Mật độ điện tích khối (C/m3)*

CÁC CHỮ VIẾT TẮT

CSTD	Công suất tác dụng
MF	Máy phát điện
BER	Tỷ lệ bít lỗi

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

CHƯƠNG 1 – PHƯƠNG PHÁP GIẢI QUYẾT ĐỀ TÀI

1.1 Mục tiêu

Bài toán đặt ra là nhận diện và trích xuất các ký tự số từ ảnh chứa mã số sinh viên (MSSV) của sinh viên. Mục tiêu của chương này là tìm ra phươn pháp giải viết về một hệ thống nhận diện ký tự số và chữ từ ảnh, cuối cùng sẽ trích xuất MSSV từ các ký tự đã nhận diện.

1.2 Phương pháp giải quyết

1.1.1 Xây dựng và huấn luyện mô hình

1.1.1.1 Kiến trúc mô hình

- Sử dụng kiến trúc mạng neural theo mô hình AlexNet, được định nghĩa trong model.py.
- Mô hình bao gồm nhiều lớp Convolutional, lớp MaxPooling, và các lớp Fully Connected.

1.1.1.2 Huấn luyện mô hình

- Sử dụng dữ liệu từ thư mục './Fnt/' bao gồm ảnh chữ cái và số trong bảng chữ cái Latinh và các chữ số.
- Sử dụng ImageDataGenerator để tạo dữ liệu từ đĩa và thực hiện data augmentation.
- Tách 20% dữ liệu cho validation.
- Sử dụng optimizer Adam, hàm loss là 'categorical_crossentropy', và theo dõi độ chính xác.

1.1.2 Trích xuất và xác định khu vực chứa MSSV

1.1.2.1 Trích xuất ảnh lớn nhất

- Sử dụng OpenCV để đọc ảnh và chuyển đổi sang ảnh xám.
- Áp dụng threshold để tạo ảnh nhị phân.

- Tìm contours và lấy contour có diện tích lớn nhất, sau đó cắt ảnh theo bounding box của contour đó.

1.1.2.2 Vẽ hình vuông quan số

- Sử dụng hàm draw_square_around_number để vẽ hình vuông quanh từng ký tự.

1.1.3 Lưu ảnh từng ký tự riêng biệt

1.1.3.1 Tiền xử lý

- Tăng cường độ sáng và chuyển đổi ảnh sang đen trắng.

1.1.3.2 Phân đoạn ảnh

- Sử dụng phương pháp thresholding để tìm contours trong ảnh.

1.1.3.3 Lưu từng ký tự thành ảnh riên biệt

- Sắp xếp contours theo diện tích giảm dần.
- Giữ lại 8 contours có diện tích lớn nhất.
- Lưu mỗi contour thành một ảnh riêng biệt sau khi áp dụng các bước tiền xử lý.

1.1.4 Nhận diện ký tự

1.1.4.1 Chuẩn bị dữ liệu cho mô hình

- Chuyển đổi kích thước ảnh lớn nhất về kích thước mong muốn và chuẩn hóa pixel trong khoảng [0, 1].

1.1.4.2 Nhận diện ký tự ảnh đã cắt

- Sử dụng mô hình đã huấn luyện để dự đoán lớp của từng ký tự trên ảnh cắt.
- Xác định lớp của mỗi ký tự và ghi lại chuỗi các ký tự đã nhận diện.

1.1.5 Kết quả

1.1.5.1 Hiển thị kết quả

- Kết quả cuối cùng là ảnh đã cắt với các ký tự được nhận diện, được in lên và được vẽ hình vuông quanh.
- Chuỗi ký tự đã nhận diện được in ra màn hình và được đảo ngược trình tự để có kết quả cuối cùng.

1.3 Triển khai chi tiết

1.3.1 Xây dựng và huấn luyện mô hình

1.3.1.1 Kiến trúc mô hình

- Trong đoạn mã model.py, kiến trúc mô hình được xây dựng theo mô hình AlexNet, một trong những mô hình nổi tiếng đầu tiên trong lĩnh vực mạng neural học sâu. Dưới đây là giải thích chi tiết từng phần của kiến trúc:
  - Khai báo thư viện và lớp Model

```
from keras.models import Sequential, Model
from keras.layers import BatchNormalization
from keras.layers import Activation
from keras.layers import Flatten
from keras.layers import Dropout
from keras.layers import Dense
from keras.layers import Conv2D, MaxPooling2D
from keras import layers
```

    - Import các thư viện và lớp cần thiết từ Keras.
  - Định nghĩa hàm Model

```
# Define the model using AlexNet architectures
def model(num_classes, input_shape):
    model = Sequential()
```

    - Sử dụng kiến trúc mô hình tuần tự Sequential của Keras.
  - Lớp Convolutional đầu tiên

```
# 1st Convolutional Layer
model.add(Conv2D(filters=96, input_shape=input_shape, kernel_size=(11,11), strides=(4,4), padding='valid'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(3,3), strides=(2,2), padding='valid'))
```

  - Lớp Conv2D với 96 filters, kernel size là (11, 11), strides là (4, 4) và sử dụng hàm kích hoạt ReLU.
  - Lớp MaxPooling2D với pool size là (3, 3) và strides là (2, 2).

- Các lớp Convolutional tiếp theo

```
# 2nd Convolutional Layer
model.add(Conv2D(filters=256, kernel_size=(5,5), strides=(1,1), padding='same'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(3,3), strides=(2,2), padding='valid'))

# 3rd Convolutional Layer
model.add(Conv2D(filters=384, kernel_size=(3,3), strides=(1,1), padding='same'))
model.add(Activation('relu'))

# 4th Convolutional Layer
model.add(Conv2D(filters=384, kernel_size=(3,3), strides=(1,1), padding='same'))
model.add(Activation('relu'))

# 5th Convolutional Layer
model.add(Conv2D(filters=256, kernel_size=(3,3), strides=(1,1), padding='same'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(3,3), strides=(2,2), padding='valid'))
```

  - Các lớp Convolutional và MaxPooling2D tiếp theo theo mô hình AlexNet.

- Lớp Fully Connected

```
# Passing it to a Fully Connected layer
model.add(Flatten())
# 1st Fully Connected Layer
model.add(Dense(4096))
model.add(Activation('relu'))
# Add Dropout to prevent overfitting
model.add(Dropout(0.5))

# 2nd Fully Connected Layer
model.add(Dense(4096))
model.add(Activation('relu'))
# Add Dropout to prevent overfitting
model.add(Dropout(0.5))
```

  - Lớp Flatten để chuyển đổi tensor thành vector.
  - Hai lớp Fully Connected với 4096 units, hàm kích hoạt ReLU và lớp Dropout để ngăn chặn overfitting.

- Lớp Output

```
# Output Layer
model.add(Dense(num_classes))
model.add(Activation('softmax'))
```

  - Lớp Fully Connected cuối cùng với số units bằng số lớp đầu ra và hàm kích hoạt softmax để đưa ra xác suất dự đoán cho từng lớp.

- Trả về mô hình

```
return model
```

  - Trả về mô hình đã được định nghĩa.

1.3.1.2 Huấn luyện mô hình

- Huấn luyện mô hình:

```
# Set up the data generator to flow data from disk
print("[INFO] Setting up Data Generator...")
data_gen = ImageDataGenerator(validation_split=0.2, rescale=1./255)

train_generator = data_gen.flow_from_directory(
    DATASET_PATH,
    subset='training',
    target_size = (TARGET_WIDTH, TARGET_HEIGHT),
    batch_size = BATCH_SIZE
)

val_generator = data_gen.flow_from_directory(
    DATASET_PATH,
    subset='validation',
    target_size = (TARGET_WIDTH, TARGET_HEIGHT),
    batch_size = BATCH_SIZE
)
```

  - Dùng ImageDataGenerator để tạo dữ liệu từ thư mục DATASET_PATH.
  - validation_split=0.2 chia dữ liệu thành tập huấn luyện và tập validation (20%).
  - rescale=1./255 chuẩn hóa giá trị pixel về khoảng [0, 1].
- Xây dựng và huấn luyện mô hình

```
# Build model
print("[INFO] Compiling model...")
alexnet = model(train_generator.num_classes, (TARGET_WIDTH, TARGET_HEIGHT, TARGET_DEPTH))

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

# Train the network
print("[INFO] Training network ...")
# Set the learning rate decay
reduce_lr = ReduceLROnPlateau(monitor='loss', factor=0.2, patience=2, min_lr=0.001)
H = alexnet.fit_generator(
    train_generator,
    validation_data=val_generator,
    steps_per_epoch=train_generator.samples // BATCH_SIZE,
    validation_steps = val_generator.samples // BATCH_SIZE,
    epochs=EPOCHS, verbose=1, callbacks=[reduce_lr])
```

- Tạo mô hình sử dụng hàm model từ model.py.
- Compile mô hình với optimizer Adam, hàm loss là 'categorical_crossentropy', và theo dõi độ chính xác.
- Sử dụng fit_generator để huấn luyện mô hình với dữ liệu từ train_generator và val_generator.
- Sử dụng ReduceLROnPlateau để giảm learning rate nếu mất mát trên tập huấn luyện không giảm sau một số epochs.

- Lưu mô hình

```
# save the model to disk
print("[INFO] Serializing network...")
alexnet.save(MODEL_PATH + os.path.sep + "trained_model")

print("[INFO] Done!")
```

- Lưu mô hình đã huấn luyện vào đường dẫn ./trained_model.

1.3.2 Trích xuất và xác định khu vực chứa MSSV

1.3.2.1 Trích xuất ảnh lớn nhất

```python
def extract_largest_image(image_path):
    # Đọc ảnh từ file
    image = cv2.imread(image_path)
    # Chuyển đổi ảnh sang ảnh xám
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Áp dụng threshold để tạo ảnh nhị phân
    _, binary_image = cv2.threshold(gray_image, 128, 255, cv2.THRESH_BINARY)
    # Tìm contours trong ảnh nhị phân
    contours, _ = cv2.findContours(binary_image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    # Tìm contour có diện tích lớn nhất
    largest_contour = max(contours, key=cv2.contourArea)
    # Lấy bounding box của contour
    x, y, w, h = cv2.boundingRect(largest_contour)
    # Cắt ảnh từ bounding box
    largest_image = image[y:y+h, x:x+w]

    return largest_image
```

- Đọc ảnh từ file: Sử dụng OpenCV để đọc ảnh từ đường dẫn được chuyển vào hàm.
- Chuyển đổi ảnh sang ảnh xám: Sử dụng cv2.cvtColor để chuyển đổi ảnh màu sang ảnh xám.
- Áp dụng threshold để tạo ảnh nhị phân: Sử dụng cv2.threshold để áp dụng một ngưỡng cho ảnh xám, tạo thành ảnh nhị phân. Trong trường hợp này, ngưỡng được đặt là 128.
- Tìm contours trong ảnh nhị phân: Sử dụng cv2.findContours để tìm contours trong ảnh nhị phân.
- Tìm contour có diện tích lớn nhất: Sử dụng hàm max để tìm contour có diện tích lớn nhất dựa trên cv2.contourArea.
- Lấy bounding box của contour: Sử dụng cv2.boundingRect để lấy bounding box (hình chữ nhật bao quanh) của contour.

- Cắt ảnh từ bounding box: Sử dụng các thông số (x, y, w, h) của bounding box để cắt ảnh lớn nhất từ ảnh gốc.

1.3.2.2 Vẽ hình vuông quan số và ghép chuỗi MSSV

```
# Hàm để vẽ hình vuông quanh số
def draw_square_around_number(image):
    result = cv2.convertScaleAbs(image, alpha=1.5, beta=0)
    # Chuyển đổi ảnh sang ảnh đen trắng để dễ xử lý
    gray_image = cv2.cvtColor(result, cv2.COLOR_BGR2GRAY)
    # Phát hiện số trong ảnh bằng phương pháp thresholding
    _, threshold_image = cv2.threshold(gray_image, 180, 255, cv2.THRESH_BINARY_INV)
    # Tìm contours trong ảnh
    contours, _ = cv2.findContours(threshold_image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    # Duyệt qua contours và vẽ hình vuông quanh số
    for contour in contours:
        # Lấy bounding box của contour
        x, y, w, h = cv2.boundingRect(contour)
        # Vẽ hình vuông quanh số
        cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

- Chúng ta chuyển đổi ảnh gốc sang ảnh có độ tương phản cao hơn bằng hàm cv2.convertScaleAbs.
- Sau đó, chuyển đổi ảnh sang ảnh đen trắng để dễ xử lý.
- Sử dụng phương pháp thresholding để phát hiện các số trong ảnh.
- Tìm contours trong ảnh và sau đó vẽ hình vuông quanh mỗi contour bằng cách sử dụng hàm cv2.rectangle.

```
strr = " "
for i in range(0, 8):
    original_image = cv2.imread(f'output_folder/digit_{i}.png')
    find_mssv(original_image)
```

- Sử dụng vòng lặp để đọc từng ảnh ký tự được lưu từ hàm save_contours_as_images.
- Với mỗi ảnh ký tự, gọi hàm find_mssv để dự đoán và ghép vào chuỗi strr.

1.3.3 Nhận diện ký tự

1.3.3.1 Nhận diện ký tự ảnh đã cắt

```
def find_mssv(original_image):
    global strr
    # Define constants
    TARGET_WIDTH = 128
    TARGET_HEIGHT = 128
    MODEL_PATH = './trained_model'

    # Preprocessing the image
    image = cv2.resize(original_image, (TARGET_WIDTH, TARGET_HEIGHT))
    image = image.astype("float") / 255.0
    image = img_to_array(image)
    image = np.expand_dims(image, axis=0)

    # Load the trained convolutional neural network
    model = load_model(MODEL_PATH, compile=False)

    # Classify the input image then find the index of the class with the *largest* probability
    prob = model.predict(image)[0]
    idx = np.argsort(prob)[-1]
    strr += labels[idx]
```

- Định nghĩa các hằng số liên quan đến kích thước ảnh đầu vào (TARGET_WIDTH và TARGET_HEIGHT) và đường dẫn của mô hình đã được huấn luyện (MODEL_PATH).
- Thay đổi kích thước ảnh đầu vào để phù hợp với kích thước đầu vào của mô hình.
- Chuẩn hóa giá trị pixel của ảnh về khoảng từ 0 đến 1.
- Chuyển đổi ảnh thành mảng numpy và thêm một chiều mới để tạo thành batch (axis=0).
- Sử dụng hàm load_model từ Keras để tải mô hình đã được huấn luyện từ đường dẫn MODEL_PATH.
- Tham số compile=False để không cần thiết lập lại các thông số biên dịch của mô hình (vì đã được biên dịch khi huấn luyện).

- Sử dụng mô hình để dự đoán lớp của ảnh đầu vào bằng cách sử dụng hàm predict.
- prob là một vector xác suất cho mỗi lớp.
- np.argsort(prob) sắp xếp các xác suất theo thứ tự tăng dần và trả về các chỉ số tương ứng.
- idx = np.argsort(prob)[-1] lấy chỉ số của lớp có xác suất cao nhất.
- Lấy nhãn của lớp được dự đoán và thêm vào biến strr.
- Biến labels chứa danh sách các nhãn tương ứng với các lớp mô hình đã huấn luyện.

1.3.4 Kết quả

1.3.4.1 Hiển thị kết quả

```
cv2.putText(image, strr, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2 )
# Overlay the drawn squares onto the original image
cv2.imwrite('./output/' + nameFile, image)
```

- MSSV sẽ được in lên ảnh thẻ SV sau đó sẽ được lưu vào folder output với tên giống với file input

CHƯƠNG 2 – TỔNG QUAN LÝ THUYẾT

2.1 Kết quả tổng quát

Cho 5 ảnh đầu vào từ thư mục "input"

Sau khi thực hiện chương trình, các ảnh thành quả sẽ được lưu vào thư mục "output":

2.2 Kết quả chi tiết

- **Test 1:**

Input: 52100049.jpg

Output: 52100049.jpg

- **Test 2:**

Input: 52100087.jpg

Output: 52100087.jpg

- **Test 3:**

- Input: 52100844.jpg

Output: 52100844.jpg

- **Test 4:**

Input: 52100542.jpg

Output: 52100542.jpg

- **Test 5:**

Input: E22H0021.jpg

Output: E22H0021.jpg

TÀI LIỆU THAM KHẢO

Tiếng Việt

Tiếng Anh

PHỤ LỤC

Phần này bao gồm những nội dung cần thiết nhằm minh họa hoặc hỗ trợ cho nội dung luận văn như số liệu, biểu mẫu, tranh ảnh. . . . nếu sử dụng những câu trả lời cho một *bảng câu hỏi thì bảng câu hỏi mẫu này phải được đưa vào phần Phụ lục ở dạng nguyên bản* đã dùng để điều tra, thăm dò ý kiến; **không được tóm tắt hoặc sửa đổi**. Các tính toán mẫu trình bày tóm tắt trong các biểu mẫu cũng cần nêu trong Phụ lục của luận văn. Phụ lục không được dày hơn phần chính của luận văn