TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN

PHÁT TRIỂN PHẦN MỀM MÃ NGUỒN MỞ
Xử lý bài toán Nhận dạng MNist trên PYTHON

GVHD: Từ Lãng Phiêu
SV: Trương Chấn Đông - 3120410133
Mai Xuân Hiếu - 3121410193
Lê Anh Duy - 3121410117

TP. HỒ CHÍ MINH, THÁNG 4/2024

Mục lục

1 Phần lý thuyết **2**
1.1 Giới thiệu về công nghệ sử dụng và thuật toán 2
1.1.1 Ngôn ngữ và công nghệ web 2
1.1.2 Các thư viện sử dụng 2
1.2 Thuật toán LinearSCV (Thuật toán phân loại) 3
1.2.1 Nguyên lý hoạt động 3
1.2.2 Đánh giá một Classification Model 3

2 Mô hình nhận dạng chữ số viết tay **4**
2.1 Dataset MNIST: 4
2.2 Trích xuất đặc trưng với HOG (Histogram of Oriented Gradients) 4
2.3 Mô hình phân loại LinearSVC (Linear Support Vector Classifier): 4
2.4 Tiến trình xây dựng mô hình nhận dạng chữ số: 4
2.5 Tích hợp vào ứng dụng web với Flask: 4

3 Phần lập trình **5**
3.1 Giải thích code trong chương trình 5
3.2 Các bước thực hiện chương trình 7
3.2.1 Kết quả chương trình 8

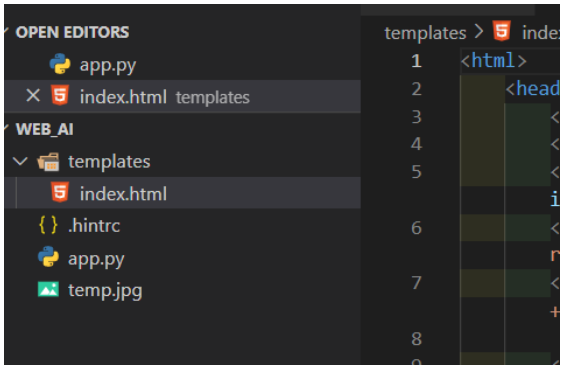
4 TÀI LIỆU THAM KHẢO **10**

1 Phần lý thuyết

1.1 Giới thiệu về công nghệ sử dụng và thuật toán

1.1.1 Ngôn ngữ và công nghệ web

Sử dụng thư viện Flask để tạo ra 1 trang web nhanh chóng. Cấu hình các router đường dẫn tạo ra trong miền. 1 file HTML được sử dụng để hiển thị chương trình, và sử dụng ngôn ngữ javascript để tạo các hiệu ứng động.



Hình 1: Caption 1

Hình 2: Caption 2

1.1.2 Các thư viện sử dụng

- **TensorFlow:** TensorFlow cho phép các lập trình viên tạo ra dataflow graph, cấu trúc mô tả làm thế nào dữ liệu có thể di chuyển qua 1 biểu đồ, hay 1 sê-ri các node đang xử lý. Mỗi node trong đồ thị đại diện 1 operation toán học, và mỗi kết nối hay edge giữa các node là 1 mảng dữ liệu đa chiều, hay còn được gọi là 'tensor'. TensorFlow cung cấp tất cả những điều này cho lập trình viên theo phương thức của ngôn ngữ Python.
- **Numpy:** Numpy sử dụng để sử dụng các câu tính toán phức tạp, vẽ đồ thị 3D và phân đoạn hình ảnh.
- **Sklearn data và dataset metrix:** Sklearn data được sử dụng để tạo dữ liệu từ mnist số, sau đó sử dụng metrix để phân bố dữ liệu.

1.2 Thuật toán LinearSCV (Thuật toán phân loại)

- Support Vector Machine (SVM) là một thuật toán giám sát được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồ thị dữ liệu là các điểm trong n chiều, với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó, chúng ta thực hiện tìm "đường bay" (hyper-plane) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.

1.2.1 Nguyên lý hoạt động

- **Hàm Ranh Giới:** LinearSVC cố gắng tìm ra một ranh giới tuyến tính trong không gian đặc trưng sao cho các điểm dữ liệu thuộc các lớp khác nhau được phân chia rõ ràng.
- **Hàm Mất Mát:** LinearSVC sử dụng một hàm mất mát để đo lường mức độ lỗi trong quá trình phân loại. Mục tiêu là tối thiểu hóa hàm mất mát này.
- **Tối Ưu Hóa:** Thuật toán sử dụng các phương pháp tối ưu hóa convex optimization để tìm ra ranh giới tốt nhất.

1.2.2 Đánh giá một Classification Model

- Các phương pháp đánh giá hiệu quả của một mô hình phân loại sau khi đã huấn luyện có thể bao gồm sử dụng confusion matrix, độ chính xác, precision, recall, F1-score, ROC Curve và AUC.

2 Mô hình nhận dạng chữ số viết tay

- Trong phần này, chúng ta sẽ đi sâu vào lý thuyết về cách xây dựng một mô hình nhận dạng chữ số sử dụng các thư viện OpenCV, scikit-image, scikit-learn và Flask.

2.1 Dataset MNIST:

- Tập dữ liệu MNIST là một tập dữ liệu lớn chứa các chữ số viết tay từ 0 đến 9. Mỗi hình ảnh có kích thước 28x28 pixel và được gắn nhãn với số tương ứng. Tập dữ liệu này thường được sử dụng làm bài toán thử nghiệm đầu tiên khi làm việc với mạng neural và các thuật toán nhận dạng chữ viết tay.

2.2 Trích xuất đặc trưng với HOG (Histogram of Oriented Gradients)

- HOG là một kỹ thuật phổ biến được sử dụng để trích xuất đặc trưng từ hình ảnh. Kỹ thuật này dựa trên việc tính toán các hướng gradient của các pixel trong ảnh và biểu diễn chúng dưới dạng histogram.
- Trong OpenCV và scikit-image, chúng ta có thể sử dụng các hàm như hog() để trích xuất đặc trưng HOG từ hình ảnh.

2.3 Mô hình phân loại LinearSVC (Linear Support Vector Classifier):

- LinearSVC là một mô hình học máy dựa trên hỗ trợ tuyến tính, thường được sử dụng cho các bài toán phân loại. Mô hình này tìm ra đường biên tuyến tính để phân chia các lớp dữ liệu.
- Trong scikit-learn, chúng ta có thể sử dụng lớp LinearSVC để tạo và huấn luyện mô hình phân loại LinearSVC.

2.4 Tiến trình xây dựng mô hình nhận dạng chữ số:

Bước 1: Chuẩn bị dữ liệu
- Sử dụng tập dữ liệu MNIST để huấn luyện và kiểm tra mô hình. Tập dữ liệu này chứa các hình ảnh chữ số viết tay từ 0 đến 9.
Bước 2: Trích xuất đặc trưng
- Sử dụng HOG để trích xuất đặc trưng từ các hình ảnh trong tập dữ liệu MNIST.
Bước 3: Huấn luyện mô hình
- Sử dụng LinearSVC để huấn luyện mô hình phân loại dựa trên các đặc trưng HOG đã trích xuất.
Bước 4: Đánh giá mô hình
- Sử dụng tập dữ liệu kiểm tra để đánh giá hiệu suất của mô hình dự đoán.

2.5 Tích hợp vào ứng dụng web với Flask:

- Sử dụng Flask để xây dựng một ứng dụng web đơn giản cho phép người dùng viết chữ số trên giao diện.
- Khi người dùng hoàn thành việc viết, ảnh của chữ số được chụp và gửi đến máy chủ.
- Máy chủ sử dụng mô hình LinearSVC đã được huấn luyện để nhận diện chữ số từ hình ảnh và trả về kết quả cho người dùng.

3 Phần lập trình

3.1 Giải thích code trong chương trình

Khai báo thư viện trong chương trình:

```
from flask import Flask, render_template, request , jsonify
import tensorflow as tf
import base64


import cv2
import numpy as np
from skimage.feature import hog
from sklearn.svm import LinearSVC
from keras.datasets import mnist
from sklearn.metrics import accuracy_score
```

Hình 3: Caption 3

Sau khi khai báo thư viện cần thiết cho chương trình, bước tiếp theo là tạo data từ mnist:

```
(X_train,y_train),(X_test,y_test) = mnist.load_data()
#cho x_train
X_train_feature = []
for i in range(len(X_train)):
    feature = hog(X_train[i],orientations=9,pixels_per_cell=(14,14),cells_per_block=(1,1),block_norm="L2")
    X_train_feature.append(feature)
X_train_feature = np.array(X_train_feature,dtype = np.float32)
```

Hình 4: Caption 4

Tách dữ liệu thành hai tập train và test từ bộ dữ liệu mnist được cung cấp bởi thư viện sklearn. Sau khi tách xong, chúng tôi tiếp tục tách nhãn thành một tệp riêng, nhãn này được sử dụng để dự đoán ($x_{train\_feature}$) và làm tương tự cho tập test ($x_{test\_feature}$). Sau đó, chúng tôi tiến hành khai báo model LinearSVC và cài đặt các thông số cho nó.

```
#cho x_test
X_test_feature = []
for i in range(len(X_test)):
    feature = hog(X_test[i],orientations=9,pixels_per_cell=(14,14),cells_per_block=(1,1),block_norm="L2")
    X_test_feature.append(feature)
X_test_feature = np.array(X_test_feature,dtype=np.float32)
model = LinearSVC(C=10)
model.fit(X_train_feature,y_train)
y_pre = model.predict(X_test_feature)
print(accuracy_score(y_test,y_pre))
```

Hình 5: Caption 5

Khai báo model LinearSVC với thông số C là 10, sau đó ta sẽ train model đó với $x_{train\_feature}$ và y_{train}. Sau đó tạo thành 1 file y-pre từ file feature rồi tính độ chính xác. Tạo trang web và đường dẫn tên miền bằng Flask:

```python
app = Flask(__name__)



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

@app.route('/recognize' , methods = ['POST'])
def recognize():
    if request.method == 'POST':
        print('adsd')
        data = request.get_json()
        imageBase64 = data['image']
        imgBytes = base64.b64decode(imageBase64)
```

Hình 6

Tạo 2 trang có tên miền khác nhau những cũng truy cấp 1 file html , trang nhận dạng kết quả là trang method bằng post. Trang nhận diện post có 1 file image lấy từ trang giao diện khi người dùng vẽ. Hình ảnh sẽ được truyền cho file test rồi model sẽ nhận diện ảnh đó rồi đưa ra kết quả

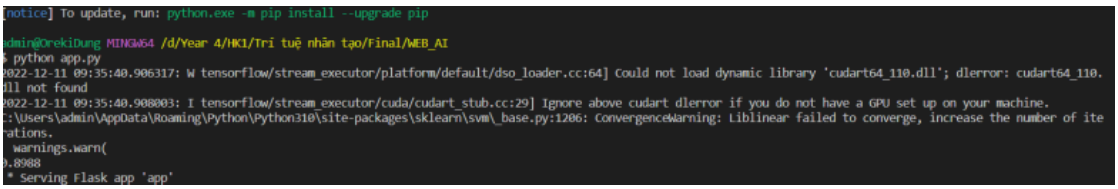
```python
    image = cv2.imread("temp.jpg")
    im_gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
    im_blur = cv2.GaussianBlur(im_gray,(5,5),0)
    im,thre = cv2.threshold(im_blur,90,255,cv2.THRESH_BINARY_INV)
    contours,hierachy = cv2.findContours(thre,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
    rects = [cv2.boundingRect(cnt) for cnt in contours]
    for i in contours:
        (x,y,w,h) = cv2.boundingRect(i)
        cv2.rectangle(image,(x,y),(x+w,y+h),(0,255,0),3)
        roi = thre[y:y+h,x:x+w]
        roi = np.pad(roi,(20,20),'constant',constant_values=(0,0))
        roi = cv2.resize(roi, (28, 28), interpolation=cv2.INTER_AREA)
        roi = cv2.dilate(roi, (3, 3))

        # Calculate the HOG features
        roi_hog_fd = hog(roi, orientations=9, pixels_per_cell=(14, 14), cells_per_block=(1, 1),block_norm="L2")
        nbr = model.predict(np.array([roi_hog_fd], np.float32))
        cv2.putText(image, str(int(nbr[0])), (x, y),cv2.FONT_HERSHEY_DUPLEX, 2, (0, 255, 255), 3)
        cv2.imshow("image",image)
```

Hình 7

3.2 Các bước thực hiện chương trình

- B1: Chạy chương trình bằng python app.py trong file :
+ Chương trình sẽ lấy file html để hiện thị , lấy port là 5000 được cấu hình bằng flask.
+ Nếu chưa tải Flask thì nên tải Flask trước bằng câu lệnh : pip3 install flask



Hình 8

B2: Hình ảnh giao diện chương trình sẽ hiển thị , người dùng sẽ ấn vào clear để xóa khoảng đen trong hình ảnh.

Hình 9

B3: Sau khi ấn clear , người dùng vẽ các chữ số từ 1 đến 9 trong ô vuông và nhấn nút màu xanh để chương trình nhận diện:

Hình 10: Enter Caption

- Người dùng có thể thay đổi cỡ chữ , tô dậm chữ và thay đổi màu sắc của nét vẽ

3.2.1 Kết quả chương trình

- Dự đoán số 3 :

Hình 11

- Vẽ số 9

Hình 12

4 TÀI LIỆU THAM KHẢO

[1] Phân nhóm các thuật toán Machine Learning). Được truy cập tại :
https://thorpham.github.io/blog/Nh%E1%BA%ADn-d%E1%BA%A1ngch%E1%BB%AF-s%E1%BB%91vi%E1%BA%BFttayfbclid=IwAR2qYQdeY4IGc589cOwoxNfbMUlfBOUuIM7vxflorC17w_DiQKFtL5p5Q#:~:text=C%C3%B3%20r%E1%BA%A5t%20nhi%E1%BB%81u%20thu%E1%BA%ADt%20to%C3%A1n,chi%20ph%C3%AD%20t%C3%ADnh%20to%C3%A1n%20th%C3%A2p&utm_source=zalo&utm_medium=zalo&utm_campaign=zalo.