1. Project 1

Nội dung:

Xây dựng ứng dụng nhận dạng chữ viết tay qua giao diện web

Yêu cầu:

Cần nhận dạng 26 chữ cái thường và 26 chữ cái hoa qua ảnh vẽ tay của người dùng trên website. Không có trước dữ liệu training, do đó ứng dụng phải tự có phần thu thập ảnh viết tay qua giao diện website.

Một số hướng dẫn:

Bước 1:
Xây dựng giao diện cho phép người dùng vẽ trên website và thu thập ảnh chữ viết về server.

Một số hướng dẫn :

- Phía client : Dùng canvas của html5 để vẽ ảnh, tham khảo chương trình sau:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
<canvas id="myCanvas" width="200" height="200" style="border:2px solid black"></canvas>
<br/>
<input type="button" value="Clear" onclick="clearArea()"/>
<input type="button" value="Save" onclick="saveImage()"/>
<input type="button" value="Recognize" onclick="recognizeChar()"/>

<script>
   var canvas = document.getElementById('myCanvas');
   var mousePressed = false;
   var lastX, lastY;
   var ctx = canvas.getContext("2d");

   $(document).ready(function() {
        $('#myCanvas').mousedown(function (e) {
             mousePressed = true;
             Draw(e.pageX - $(this).offset().left, e.pageY - $(this).offset().top, false);
        });

        $('#myCanvas').mousemove(function (e) {
             if (mousePressed) {
                  Draw(e.pageX - $(this).offset().left, e.pageY - $(this).offset().top, true);
             }
        });

        $('#myCanvas').mouseup(function (e) {
             mousePressed = false;
        });

        $('#myCanvas').mouseleave(function (e) {
             mousePressed = false;
        });
   });

   function Draw(x, y, isDown) {
        if (isDown) {
             ctx.beginPath();
             ctx.strokeStyle = 'black';
             ctx.lineWidth = 10;
```

```
            ctx.lineJoin = "round";
            ctx.moveTo(lastX, lastY);
            ctx.lineTo(x, y);
            ctx.closePath();
            ctx.stroke();
        }
        lastX = x; lastY = y;
    }

    function clearArea() {
        ctx.setTransform(1, 0, 0, 1, 0, 0);
        ctx.clearRect(0, 0, ctx.canvas.width, ctx.canvas.height);
    }

    function saveImage() {
        var image_data = canvas.toDataURL();
        data = JSON.stringify({image_data : image_data});

        $.ajax({
            url: "http://localhost:5000/images/save",
            contentType: "application/json",
            crossDomain:true,
            data : data,
            method: "POST",
            success: function(result){
              clearArea();
            }
         });
    }

    function recognizeChar() {
        // TODO
    }
</script>
```

Giao diện vẽ và thu thập ảnh phía client

- Phiá server : Sử dụng web framework flask của python để thu thập ảnh do client gửi về

Cài đăt flask (nếu chưa có sẵn) :

```
import pip
pip.main(['install', 'flask'])
pip.main(['install', 'flask_cors'])
```

Tham khảo ứng dụng server để lấy ảnh do client gửi về:

```
from flask import Flask, request, jsonify
from flask_cors import CORS
import os
```

```
import base64

app = Flask(__name__)
CORS(app)

images_folder = "images"
image_index = 1

@app.route('/images/save', methods=['POST'])
def save_image():
    global image_index
    img_path = os.path.join(images_folder, str(image_index) + ".png")
    image_index += 1
    image_data = request.json['image_data']
    image_data = image_data[22:]                # remove header padding
    f = open(img_path, 'wb')
    f.write(base64.b64decode(image_data))
    return jsonify('OK')


if __name__ == '__main__':
    app.run(host="0.0.0.0", port=5000, threaded=False)
```

Bước 2:
Tạo mô hình neural network, dùng dữ liệu để huấn luyện mạng neuron, lưu mô hình sau huấn luyện lại thành file để sử dụng về sau. Cách thực hiện tương tự với bài toán MNIST. Có thể dùng mạng MLP hoặc CNN.

Bước 3:
Viết thêm API cho server để thực hiện nhận dạng ảnh do client gửi về. Cách load model và dùng model tham khảo bài toán MNIST. Cách gửi ảnh từ client về server tham khảo bước 1.

2. Project 2

Nội dung

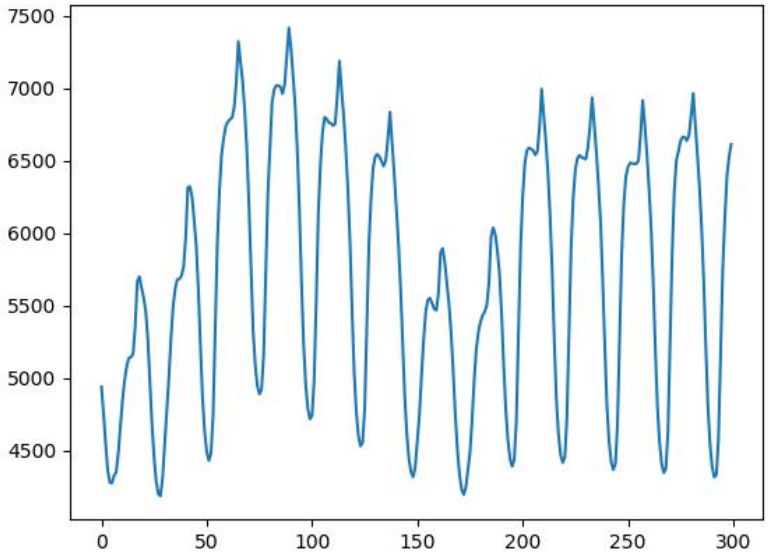
Xây dựng mô hình mạng neuron cho bài toán dự đoán tín hiệu chuỗi thời gian (Time series)

Yêu cầu:

Dự đoán nhu cầu tiêu thụ điện theo từng giờ của một thành phố dựa trên dữ liệu đã có trong quá khứ và thông tin dự báo nhiệt độ từ trung tâm dự báo thời tiết.

File dữ liệu có thể download về : nyc.zip, khi giải nén ra sẽ gồm 2 file:

- nyc_weather.csv : chứa thông tin về nhiệt độ theo từng giờ của thành phố NewYork từ năm 2012 đến năm 2017
- nyc_demand.csv : lượng điện năng tiêu thụ theo từng giờ của thành phố, cũng từ năm 2012 đến năm 2017



Đồ thị điện năng tiêu thụ theo từng giờ

Lưu ý : Hai file dữ liệu do 2 nguồn khác nhau cung cấp, do đó có một số ngày/giờ không có dữ liệu ở một trong hai file, vấn đề này cần xử lý khi chuẩn bị dữ liệu

Cần xây dựng một mô hình cho phép dự đoán lượng điện năng thành phố tiêu thụ trong vòng 24 giờ sắp tới (yêu cầu dự đoán cho từng giờ một). Thời điểm đưa ra dự báo là 0h mỗi ngày. Tại thời điểm này, các dữ liệu có thể dùng cho việc dự đoán gồm:

- Toàn bộ dữ liệu lịch sử trước thời điểm đang xét
- Thông tin về nhiệt độ do trung tâm dự báo thời tiết đưa ra trong 24 giờ tới

Để giảm thời gian chạy chương trình, chỉ yêu cầu dự đoán điện năng tiêu thụ cho các ngày/giờ trong năm 2017.

Một số gợi ý:

- Lựa chọn đầu vào:
  - Giờ trong ngày : mã hóa *onehot* với độ dài bằng 24
  - Ngày trong tuần : mã hóa binary 0/1 với 0 là ngày thường, 1 là ngày cuối tuần
  - Tháng trong năm : mã hóa *onehot* với độ dài 12
  - Nhiệt độ từ thông tin dự báo thời tiết : chỉ dùng thông tin nhiệt độ tương ứng với giờ cần dự báo, hoặc trong lân cận 1-2 giờ
  - Dữ liệu lịch sử : lấy trung bình điện năng tiêu thụ của 1-2 ngày trước. Nếu ngày hôm trước là cuối tuần (thứ 7/ chủ nhật) thì lùi lại tiếp trước ngày cuối tuần đó

- Chọn mô hình :
  - Có thể thử nghiệm với Linear Regression để có ước lượng sơ bộ
  - Sử dụng mạng neuron nhiều lớp để cho kết quả chính xác cao hơn