**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề tài số 08: Xây dựng hệ thống nhận diện biển báo giao thông bằng CNN và Keras**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20211651 | Nguyễn Thiện Nhật Nam | DCCNTT12.10.6 |
| **2** | 20211676 | Nguyễn Đức Thịnh | DCCNTT12.10.6 |

**Bắc Ninh, năm 2024**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề tài số 08: Xây dựng hệ thống nhận diện biển báo giao thông bằng CNN và Keras**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20211651 | Nguyễn Thiện Nhật Nam | DCCNTT12.10.6 |
| **2** | 20211676 | Nguyễn Đức Thịnh | DCCNTT12.10.6 |

**Bắc Ninh, năm 2024**

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á  **KHOA CÔNG NGHỆ THÔNG TIN** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ 1, NĂM HỌC 2024** – **2025** |

|  |  |
| --- | --- |
| **PHIẾU CHẤM THI BÀI TẬP LỚN KẾT THÚC HỌC PHẦN**  **Mã đề thi: 08**  **Tên học phần: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ:**  **XATGMT.03.K12.06.LH.C04.1\_LT** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)* | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Nguyễn Thiện Nhật Nam** | **Nguyễn Đức Thịnh** |
| --- | --- | --- | --- | --- |
| 20211651 | 20211676 |
| **1** | **Nội dung báo cáo trên Word đầy đủ** | **3.5** |  |  |
| 1.1 | Có bố cục rõ ràng (mục lục, phần mở đầu, nội dung chính, kết luận). | 0,5 |  |  |
| 1.2 | Nội dung phân tích rõ ràng, logic. | 0,5 |  |  |
| 1.3 | Có dẫn chứng, số liệu minh họa đầy đủ. | 0,5 |  |  |
| 1.4 | Ngôn ngữ và trình bày chuẩn, không lỗi chính tả. | 0,5 |  |  |
| 1.5 | Có trích dẫn tài liệu tham khảo đúng quy cách. | 0,5 |  |  |
| 1.6 | Được trình bày chuyên nghiệp (canh lề, font chữ, khoảng cách dòng hợp lý). | 0,5 |  |  |
| 1.7 | Tài liệu đầy đủ, bám sát yêu cầu của đề bài. | 0,5 |  |  |
| **2** | **Nội dung thuyết trình đầy đủ** | **1.0** |  |  |
| 2.1 | Trình bày tự tin, phát âm rõ ràng, mạch lạc. | 0,5 |  |  |
| 2.2 | Nội dung thuyết trình đúng trọng tâm, không lan man. | 0,5 |  |  |
| **3** | **Slides báo cáo đầy đủ nội dung + Hỏi đáp** | **3.0** |  |  |
| 3.1 | Slides có bố cục rõ ràng (mở đầu, nội dung, kết luận). | 0,5 |  |  |
| 3.2 | Thiết kế slides đẹp, chuyên nghiệp (màu sắc, hình ảnh minh họa). | 0,5 |  |  |
| 3.3 | Nội dung trên slides ngắn gọn, dễ hiểu, súc tích. | 0,5 |  |  |
| 3.4 | Nội dung slides phù hợp với nội dung báo cáo. | 0,5 |  |  |
| 3.5 | Trả lời câu hỏi đầy đủ, chính xác. | 0,5 |  |  |
| 3.6 | Trả lời câu hỏi tự tin, thuyết phục. | 0,5 |  |  |
| **4** | **Code đầy đủ** | **2.5** |  |  |
| 1.1 | Code được trình bày rõ ràng, có chú thích đầy đủ. | 0,5 |  |  |
| 1.2 | Code chạy đúng, không lỗi. | 0,5 |  |  |
| 1.3 | Code tối ưu, không dư thừa. | 0,5 |  |  |
| 1.4 | Đáp ứng đầy đủ các yêu cầu chức năng theo đề bài. | 0,5 |  |  |
| 1.5 | Có tính sáng tạo hoặc cải thiện so với yêu cầu. | 0,5 |  |  |
| **TỔNG ĐIỂM BẰNG SỐ:** | | **10** |  |  |
| **TỔNG ĐIỂM BẰNG CHỮ:** | | *Mười tròn* |  |  |

MỤC LỤC

[LỜI CẢM ƠN 8](#_Toc184471940)

[LỜI NÓI ĐẦU 9](#_Toc184471941)

[CHƯƠNG I: CƠ SỞ LÝ THUYẾT 10](#_Toc184471942)

[1.1. Xử lý ảnh 10](#_Toc184471943)

[1.2. Biểu diễn ảnh trong máy tính 10](#_Toc184471944)

[1.2.1. Thu nhận ảnh 11](#_Toc184471945)

[1.2.2. Các hệ màu trên máy tính 11](#_Toc184471946)

[1.3. Tổng quan về học máy 13](#_Toc184471947)

[1.3.1. Học máy có giám sát 14](#_Toc184471948)

[1.3.1.1. K-Nearest Neighbors (KNN) 15](#_Toc184471949)

[1.3.1.2. Support Vector Machines (SVM) 16](#_Toc184471950)

[1.3.2. Học máy không giám sát 17](#_Toc184471951)

[1.3.2.1. Thuật toán K-means 18](#_Toc184471952)

[1.3.3. Học máy bán giám sát 21](#_Toc184471953)

[CHƯƠNG II: XÂY DỰNG HỆ THỐNG NHẬN DIỆN BIỂN BÁO GIAO THÔNG BẰNG CNN VÀ KERAS 23](#_Toc184471954)

[2.1. Thư viện 23](#_Toc184471957)

[2.1.1. Keras 23](#_Toc184471958)

[2.1.2. Sklearn 24](#_Toc184471959)

[2.1.3. Matplotlib 26](#_Toc184471960)

[2.1.4. Pil 27](#_Toc184471961)

[2.1.5. Numpy 28](#_Toc184471962)

[2.1.6. Os 30](#_Toc184471963)

[2.1.7. Tkinter 31](#_Toc184471964)

[2.2. Xây dựng hệ thống nhận diện biển báo giao thông 32](#_Toc184471965)

[2.2.1. Vấn đề bài toán 32](#_Toc184471966)

[2.2.2. Ý tưởng thực hiện 33](#_Toc184471967)

[2.2.3. Xây dựng hệ thống 35](#_Toc184471968)

[CHƯƠNG III: THỰC NGHIỆM 41](#_Toc184471969)

[3.1. Độ đánh giá 41](#_Toc184471973)

[3.2. Kết quả thực nghiệm 42](#_Toc184471974)

[DANH MỤC THAM KHẢO 45](#_Toc184471975)

# LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, sự giúp đỡ dù ít hay nhiều, dù là trực tiếp hay gián tiếp của người khác. Trong suốt thời gian từ khi bắt đầu học tập, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của thầy cô, gia đình và bạn bè.

Chúng em xin chân thành cảm ơn cô Lương Thị Hồng Lan đã tận tâm hướng dẫn chúng em qua từng buổi học trên lớp cũng như những buổi nói chuyện, thảo luận về môn học. Trong thời gian được học tập và thực hành dưới sự hướng dẫn của thầy cô, chúng em không những thu được rất nhiều kiến thức bổ ích, mà còn được truyền sự say mê và thích thú. Nếu không có những lời hướng dẫn, dạy bảo của cô thì chúng em nghĩ báo cáo này của chúng em rất khó có thể hoàn thành được.

Chúng em xin chân thành gửi lời cảm ơn sâu sắc tới cô đã tận tình giảng dạy và truyền đạt cho chúng em những kiến thức, kinh nghiệm quý báu trong suốt thời gian học tập và rèn luyện.

Sau cùng, chúng em xin kính chúc các thầy cô trong Khoa Công Nghệ Thông Tin dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

*Chúng em xin chân thành cảm ơn!*

# LỜI NÓI ĐẦU

Ngày nay với các tiến bộ của khoa học kỹ thuật thì mọi công việc hầu như đều có thể tiến hành trên máy tinh một cách tự động hóa hoàn toàn hoặc một phần. Một trong những sự thay đổi lớn đó là cách thức chúng ta thu nhận và xử lý dữ liệu. Các công cụ nhập liệu như bàn phím hay máy scan dần bị thay thế bằng các thiết bị tiện lợi hơn như màn hình cảm ứng, camera...

Hơn thế nữa, các máy tính để bàn không còn là công cụ duy nhất có thê hồ trợ cho con người. Chúng ta bước sang thế kỷ 21 với sự phát triển mạnh mẽ của các thiết bị di động, giải trí câm tay hay smartphone. Với kích thước ngày càng nhỏ gọn và hiệu suất làm việc thì không ngừng được cải tiền, các công cụ mini này hứa hẹn sẽ là một phân không thể thiếu trong xã hội hiện đại. Và do đó, phát triển các ứng dụng trên các thiết bị này cũng là một xu thế tất yếu.

Công nghệ nhận dạng là một trong các công nghệ đang được áp dụng cho các thiết bị di động hiện nay. Nhận dạng có thế bao gồm nhận dạng âm thanh, hình ảnh. Các đối tượng nhận dạng có nhiều kiểu như tiếng nói, chữ viết, khuôn mặt, mã vạch ... và biển báo giao thông cùng là một trong số đó. Chương trình nhận dạng biển báo giao thông thường phức tạp và được cài đặt trên những hệ thống có bộ xử lý lớn, camera chất lượng cao. Mục tiêu của báo cáo là giúp chúng ta phát hiện biển báo và nhận dạng nó một cách nhanh nhất.​

# CHƯƠNG I: CƠ SỞ LÝ THUYẾT

## Xử lý ảnh

Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ hoạ đó phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Xử lý ảnh và đồ hoạ đóng một vai trò quan trọng trong tương tác người máy.

Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh “tốt hơn” hoặc một kết luận.

A black and white rectangular sign with black text

Description automatically generated

Hình 1.1: Quá trình xử lý ảnh

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến P(c1, c2,..., cn). Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều.

Sơ đồ tổng quát của một hệ thống xử lý ảnh:

A diagram of a flowchart

Description automatically generated

Hình 1.2: Các bước cơ bản trong một hệ thống xử lý ảnh

## Biểu diễn ảnh trong máy tính

Ảnh số trong máy tính là một tập hợp các giá trị số mô tả đặc điểm của mỗi điểm ảnh (pixel). Những giá trị này được tổ chức và xử lý dưới dạng ma trận, giúp máy tính lưu trữ, hiển thị và xử lý hình ảnh. Để hiểu rõ hơn, cần xem xét quy trình từ thu nhận ảnh đến biểu diễn ảnh thông qua các hệ màu.

### Thu nhận ảnh

Thu nhận ảnh là bước đầu tiên trong quá trình số hóa hình ảnh từ thế giới thực. Quá trình này bao gồm:

* **Chụp ảnh từ cảnh vật thực tế**:
* Ánh sáng từ cảnh vật đi qua ống kính, được điều chỉnh bởi khẩu độ và tốc độ màn trập.
* Hệ thống quang học tập trung ánh sáng lên bề mặt cảm biến.
* **Cảm biến ánh sáng (CCD hoặc CMOS)**:
* **CCD (Charge-Coupled Device)**: Sử dụng các cảm biến nhạy sáng để chuyển đổi ánh sáng thành điện tích.
* **CMOS (Complementary Metal-Oxide Semiconductor)**: Công nghệ hiện đại hơn, tích hợp nhiều chức năng trên cùng một con chip.
* Kết quả: Ánh sáng được biến đổi thành tín hiệu điện tử tương ứng.
* **Mẫu hóa (Sampling)**:
* Hình ảnh được chia thành lưới các điểm ảnh (pixel). Mỗi pixel là đơn vị cơ bản của hình ảnh số.
* Độ phân giải của ảnh (ví dụ: 1920x1080) xác định số lượng pixel trong hình.
* **Lượng tử hóa (Quantization)**:
* Mỗi pixel được gán một giá trị số dựa trên mức độ sáng (đối với ảnh xám) hoặc tổ hợp các mức sáng của các kênh màu (đối với ảnh màu).
* Ví dụ:
  + Ảnh xám 8-bit: Giá trị từ 0 (đen) đến 255 (trắng).
  + Ảnh màu: Giá trị của từng kênh (R, G, B) trong khoảng [0, 255].
* **Lưu trữ và xử lý dữ liệu**:
* Ảnh sau khi số hóa được lưu trữ dưới các định dạng như JPEG, PNG, BMP, hoặc RAW. Các định dạng này có thể nén dữ liệu hoặc giữ nguyên chất lượng tùy vào yêu cầu sử dụng.

### Các hệ màu trên máy tính

Hệ màu là một cách biểu diễn và tổ chức thông tin màu sắc. Mỗi hệ màu có cách định nghĩa và ứng dụng khác nhau, dựa trên cách mắt người hoặc thiết bị nhận biết màu sắc.

* **RGB (Red, Green, Blue)**
* **Đặc điểm**:
  + RGB là hệ màu phổ biến nhất, sử dụng ba kênh màu chính: Đỏ (R), Xanh lá (G), và Xanh lam (B).
  + Màu sắc được tạo ra bằng cách kết hợp các mức sáng khác nhau của ba kênh.
* **Biểu diễn**:
  + Mỗi kênh có giá trị từ 0 đến 255 (8-bit), hoặc 0 đến 1 (chuẩn hóa).
  + Ví dụ: (255, 0, 0) biểu diễn màu đỏ thuần túy; (0, 255, 0) là màu xanh lá.
* **Ứng dụng**:
  + Dùng trong hiển thị màn hình (monitor, TV).
  + Là hệ màu cơ bản cho đồ họa máy tính và xử lý ảnh.
* **CMYK (Cyan, Magenta, Yellow, Black)**
* **Đặc điểm**:
  + Hệ màu dựa trên nguyên tắc trừ màu (subtractive color), nghĩa là ánh sáng bị hấp thụ thay vì phát ra.
  + Dùng chủ yếu trong in ấn vì phù hợp với vật liệu mực in.
* **Biểu diễn**:
  + Sử dụng bốn kênh: Cyan (Xanh lơ), Magenta (Hồng cánh sen), Yellow (Vàng), và Black (Đen).
  + Mực đen (K) được thêm vào để tăng cường độ tương phản và giảm lượng mực cần dùng.
* **Ứng dụng**:
  + Trong ngành công nghiệp in ấn (sách, báo, tạp chí).
* **Grayscale (Ảnh xám)**
* **Đặc điểm**:
  + Chỉ biểu diễn độ sáng của hình ảnh, không có thông tin màu sắc.
  + Mỗi pixel chỉ chứa một giá trị, từ 0 (đen) đến 255 (trắng).
* **Ứng dụng**:
  + Dùng trong xử lý ảnh ban đầu (trích xuất đặc trưng, nhận diện biên).
  + Phân tích các ảnh không yêu cầu thông tin màu sắc, như ảnh X-quang.
* **HSV/HSB (Hue, Saturation, Value/Brightness)**
* **Đặc điểm**:
  + Là hệ màu trực quan, mô tả cách mắt người cảm nhận màu sắc.
  + Thành phần:
    - **Hue (Sắc độ)**: Góc trên vòng tròn màu (0° - 360°), xác định loại màu (đỏ, vàng, xanh, v.v.).
    - **Saturation (Độ bão hòa)**: Độ tinh khiết của màu, từ nhạt đến đậm.
    - **Value/Brightness (Độ sáng)**: Độ sáng của màu, từ tối đến sáng.
* **Ứng dụng**:
  + Phù hợp cho chỉnh sửa ảnh và thiết kế đồ họa.
  + Dễ dàng tách biệt sắc độ để phân tích hoặc chỉnh sửa.
* ***LAB (CIE Lab)***
* **Đặc điểm**:
  + LAB là hệ màu được thiết kế dựa trên nhận thức màu của mắt người.
  + Thành phần:
    - **L**\*: Độ sáng (0 là đen, 100 là trắng).
    - **a**\*: Từ xanh lá (-) đến đỏ (+).
    - **b**\*: Từ xanh lam (-) đến vàng (+).
* **Ưu điểm**:
  + Bao phủ không gian màu rộng hơn RGB hoặc CMYK.
  + Không bị phụ thuộc vào thiết bị hiển thị (device-independent).
* **Ứng dụng**:
  + Nghiên cứu màu sắc, xử lý hình ảnh yêu cầu độ chính xác cao.

**Kết luận**: Biểu diễn ảnh trong máy tính không chỉ phụ thuộc vào dữ liệu số của từng pixel mà còn gắn liền với cách mã hóa màu sắc thông qua các hệ màu khác nhau. Lựa chọn hệ màu phụ thuộc vào mục đích sử dụng, từ hiển thị, in ấn, đến phân tích dữ liệu.

## Tổng quan về học máy

Học máy (Machine Learning) là một lĩnh vực của trí tuệ nhân tạo (AI), trong đó các hệ thống máy tính học từ dữ liệu để tự cải thiện hiệu suất mà không cần lập trình rõ ràng. Học máy dựa trên việc xây dựng các mô hình toán học từ dữ liệu và áp dụng chúng để đưa ra dự đoán, phân loại, hoặc tìm hiểu các cấu trúc ẩn trong dữ liệu.

### Học máy có giám sát

* **Khái niệm:**

Học có giám sát là quá trình học tập với sự hướng dẫn từ dữ liệu đã được gắn nhãn (labelled data). Máy học một ánh xạ từ đầu vào (X) đến đầu ra mong muốn (Y) bằng cách phân tích các cặp dữ liệu trong tập huấn luyện.

* **Cấu trúc của dữ liệu:**
* **Dữ liệu đầu vào (X)**: Là các đặc trưng hoặc thuộc tính (features) của vấn đề cần giải quyết.
* **Dữ liệu đầu ra (Y)**: Là nhãn hoặc giá trị tương ứng cần dự đoán, ví dụ:
  + Phân loại (Classification): Y là nhãn rời rạc, như "mèo" hoặc "chó".
  + Hồi quy (Regression): Y là giá trị liên tục, như giá nhà hoặc nhiệt độ.
* **Quy trình học có giám sát:**

1. **Thu thập dữ liệu**:
   * Chuẩn bị một tập dữ liệu bao gồm cả X (đầu vào) và Y (đầu ra).
2. **Chia tập dữ liệu**:
   * Chia tập dữ liệu thành hai phần: **Tập huấn luyện** (training set) và **tập kiểm tra** (test set).
3. **Huấn luyện mô hình**:
   * Áp dụng thuật toán học máy để học một ánh xạ từ X đến Y.
4. **Đánh giá mô hình**:
   * Sử dụng tập kiểm tra để đo lường độ chính xác hoặc hiệu suất của mô hình.
5. **Tối ưu hóa**:
   * Điều chỉnh mô hình để giảm sai số và cải thiện độ tổng quát.

* **Các bài toán phổ biến trong học có giám sát:**

1. **Phân loại (Classification)**:
   * Mục tiêu: Phân loại các mẫu dữ liệu vào một trong các nhóm hoặc lớp (categories).
   * Ví dụ:
     + Nhận diện email là "thư rác" (spam) hoặc "hợp lệ" (not spam).
     + Dự đoán bệnh từ các chỉ số sức khỏe.
   * Thuật toán: Logistic Regression, Decision Tree, Random Forest, SVM, Neural Networks.
2. **Hồi quy (Regression)**:
   * Mục tiêu: Dự đoán một giá trị liên tục dựa trên dữ liệu đầu vào.
   * Ví dụ:
     + Dự đoán giá nhà dựa trên diện tích và vị trí.
     + Ước tính nhiệt độ của ngày mai.
   * Thuật toán: Linear Regression, Ridge Regression, Lasso Regression, Support Vector Regression.

#### K-Nearest Neighbors (KNN)

**K-Nearest Neighbors (KNN)** là một thuật toán học máy thuộc nhóm phân loại (classification) và hồi quy (regression). KNN không yêu cầu quá trình huấn luyện phức tạp, và có thể được coi là một thuật toán học máy "phi tham số" (non-parametric), có nghĩa là nó không giả định về phân phối của dữ liệu.

* **Cách thức hoạt động của KNN:**

1. **Dự đoán cho một điểm mới**:
   * Để phân loại (hoặc dự đoán giá trị) cho một điểm mới, KNN tìm **K điểm gần nhất** trong tập huấn luyện (theo một tiêu chí đo lường khoảng cách, thường là khoảng cách Euclidean).
   * Sau đó, thuật toán sẽ phân loại điểm mới dựa trên nhãn của những điểm gần nhất (phân loại) hoặc tính giá trị trung bình của các nhãn gần nhất (hồi quy).
2. **Lựa chọn giá trị K**:
   * K là một tham số quan trọng quyết định số lượng điểm lân cận được sử dụng trong việc đưa ra dự đoán. K phải được chọn sao cho mô hình không bị overfitting (quá khớp) hoặc underfitting (chưa khớp).
3. **Đo khoảng cách**:
   * Các loại khoảng cách phổ biến thường được sử dụng là:
     + **Khoảng cách Euclidean** (đo khoảng cách thẳng giữa hai điểm trong không gian).
     + **Khoảng cách Manhattan** (tính theo trục x, y, z, v.v.).
     + **Khoảng cách Cosine** (thường dùng trong văn bản).
4. **Ưu điểm**:
   * Đơn giản, dễ hiểu và dễ triển khai.
   * Không yêu cầu huấn luyện phức tạp (tính toán sau khi nhận dữ liệu).
   * Mạnh mẽ trong các bài toán phân loại không có phân phối tuyến tính rõ ràng.
5. **Nhược điểm**:
   * **Chi phí tính toán cao** khi tập dữ liệu lớn, vì phải tính khoảng cách với tất cả các điểm trong tập huấn luyện.
   * Cần phải lựa chọn giá trị K phù hợp.
   * KNN không hoạt động tốt khi dữ liệu có nhiều đặc trưng không quan trọng (đặc trưng nhiễu).

#### Support Vector Machines (SVM)

**Support Vector Machines (SVM)** là một thuật toán học máy mạnh mẽ, đặc biệt trong các bài toán phân loại với không gian đặc trưng có chiều cao. SVM được phát triển dựa trên lý thuyết không gian đặc trưng và có khả năng tìm ra siêu phẳng phân tách dữ liệu tốt nhất.

* **Cách thức hoạt động của SVM:**

1. **Siêu phẳng (Hyperplane)**:
   * SVM tìm một siêu phẳng phân tách dữ liệu sao cho khoảng cách giữa các điểm gần nhất của mỗi lớp (gọi là **Support Vectors**) và siêu phẳng này là lớn nhất.
   * Trong không gian 2D, siêu phẳng là một đường thẳng, trong không gian 3D, siêu phẳng là một mặt phẳng, và trong không gian có chiều cao hơn, nó là một hyperplane.
2. **Tối ưu hóa và Margin**:
   * Mục tiêu của SVM là tối đa hóa **margin**, tức là khoảng cách giữa siêu phẳng và các điểm gần nhất của mỗi lớp (Support Vectors).
   * Độ rộng của margin càng lớn thì mô hình càng có khả năng tổng quát tốt hơn.
3. **Kernel Trick**:
   * SVM có thể sử dụng kỹ thuật **Kernel Trick** để chuyển các dữ liệu không phân tách được trong không gian đầu vào thành không gian đặc trưng cao hơn, giúp phân tách dữ liệu theo một siêu phẳng.
   * Các kernel phổ biến: **Linear**, **Polynomial**, **Radial Basis Function (RBF)**.
4. **Hỗ trợ cho nhiều lớp (Multi-class)**:
   * Mặc dù SVM ban đầu được thiết kế cho bài toán phân loại nhị phân, nhưng nó cũng có thể mở rộng cho các bài toán phân loại nhiều lớp bằng cách sử dụng các chiến lược như **One-vs-All** hoặc **One-vs-One**.
5. **Ưu điểm**:
   * Hiệu suất cao, đặc biệt trong các bài toán phân loại với không gian đặc trưng cao.
   * Có thể sử dụng kernel để giải quyết các bài toán phân loại phi tuyến tính.
   * SVM có thể làm việc tốt với dữ liệu nhỏ và phân tách rõ ràng.
6. **Nhược điểm**:
   * **Chi phí tính toán cao** trong các bài toán có dữ liệu lớn, vì cần phải tính toán ma trận nhân nội tại (inner product) cho các cặp điểm.
   * Các tham số như C (số lượng sai lệch cho phép) và kernel cần được điều chỉnh tốt.
   * Không hoạt động tốt khi dữ liệu có rất nhiều nhiễu hoặc không phân tách rõ ràng.

### Học máy không giám sát

* **Khái niệm:**

Học không giám sát là phương pháp học trên dữ liệu không được gắn nhãn (X không có Y). Hệ thống tự tìm kiếm các mẫu, cấu trúc, hoặc mối quan hệ tiềm ẩn trong dữ liệu.

* **Mục tiêu:**
* Phát hiện các cấu trúc ẩn hoặc mối quan hệ trong dữ liệu.
* Tóm tắt, giảm kích thước hoặc trực quan hóa dữ liệu.
* **Quy trình học không giám sát:**

1. **Thu thập dữ liệu**:
   * Dữ liệu đầu vào chỉ bao gồm các đặc trưng X.
2. **Áp dụng thuật toán**:
   * Sử dụng các thuật toán để tìm hiểu dữ liệu, ví dụ:
     + Nhóm các mẫu tương tự nhau.
     + Xác định cấu trúc dữ liệu.
3. **Đánh giá và phân tích**:
   * Đánh giá dựa trên tính hợp lý của các nhóm hoặc cấu trúc được phát hiện.

* **Các bài toán phổ biến trong học không giám sát:**

1. **Phân cụm (Clustering)**:
   * Mục tiêu: Nhóm các đối tượng tương tự vào cùng một cụm.
   * Ví dụ:
     + Phân nhóm khách hàng dựa trên hành vi mua hàng.
     + Nhận diện các nhóm tin tức có nội dung tương tự.
   * Thuật toán: K-Means, DBSCAN, Hierarchical Clustering.
2. **Giảm chiều dữ liệu (Dimensionality Reduction)**:
   * Mục tiêu: Tóm gọn dữ liệu nhiều chiều thành ít chiều hơn mà vẫn giữ được thông tin quan trọng.
   * Ví dụ:
     + Trực quan hóa dữ liệu phức tạp trên mặt phẳng 2D hoặc 3D.
     + Tăng hiệu suất cho các mô hình học máy.
   * Thuật toán: PCA (Principal Component Analysis), t-SNE, Autoencoders.
3. **Phát hiện bất thường (Anomaly Detection)**:
   * Mục tiêu: Phát hiện các điểm dữ liệu khác thường hoặc không phù hợp với phần còn lại.
   * Ví dụ:
     + Phát hiện gian lận thẻ tín dụng.
     + Tìm kiếm lỗi trong sản xuất.

#### Thuật toán K-means

**K-means** là một thuật toán phân cụm (clustering) phổ biến và đơn giản trong học máy không giám sát. Mục tiêu của thuật toán này là phân nhóm các điểm dữ liệu thành **K** cụm sao cho các điểm trong cùng một cụm có sự tương đồng cao, và sự khác biệt giữa các cụm là lớn nhất. Thuật toán K-means rất hiệu quả trong việc phân nhóm dữ liệu, đặc biệt là trong các bài toán phân tích dữ liệu lớn.

* **Cách thức hoạt động của K-means:**

1. **Chọn số cụm K**:
   * Thuật toán bắt đầu với việc chọn một số **K** cụm mà bạn muốn phân chia dữ liệu. Số cụm K là một tham số được xác định trước, và bạn phải thử nghiệm với các giá trị khác nhau của K để tìm ra số cụm tối ưu.
2. **Khởi tạo các centroid**:
   * Chọn ngẫu nhiên **K** điểm trong không gian dữ liệu làm các "centroid" ban đầu. Centroid là điểm trung tâm của mỗi cụm, được tính bằng trung bình cộng của tất cả các điểm trong cụm đó.
3. **Gán các điểm dữ liệu vào cụm**:
   * Sau khi khởi tạo các centroid, thuật toán sẽ gán mỗi điểm dữ liệu vào cụm mà nó có khoảng cách gần nhất với centroid. Khoảng cách thường được đo bằng **khoảng cách Euclidean**.

d(x,c)=(x1​−c1​)2+(x2​−c2​)2+⋯+(xn​−cn​)2

trong đó x là điểm dữ liệu và ccc là centroid.

1. **Cập nhật centroid**:
   * Sau khi các điểm dữ liệu đã được phân nhóm vào các cụm, thuật toán sẽ tính lại centroid cho mỗi cụm bằng cách tính trung bình của tất cả các điểm dữ liệu trong cụm đó.
2. **Lặp lại**:
   * Thuật toán sẽ tiếp tục lặp lại các bước 3 và 4 (gán điểm dữ liệu vào cụm gần nhất và cập nhật centroid) cho đến khi không có sự thay đổi lớn giữa các centroid trong các lần lặp tiếp theo, hoặc thuật toán đạt được một số lần lặp tối đa.

* **Ưu điểm của K-means:**

1. **Đơn giản và dễ hiểu**:
   * K-means là một thuật toán rất đơn giản, dễ triển khai và dễ hiểu. Nó có thể áp dụng cho nhiều loại dữ liệu khác nhau.
2. **Hiệu quả tính toán**:
   * K-means có độ phức tạp tính toán thấp, với thời gian thực thi O(n \* K \* t), trong đó nnn là số điểm dữ liệu, KKK là số cụm và ttt là số lần lặp. Điều này làm cho thuật toán phù hợp với các tập dữ liệu lớn.
3. **Tự động phân nhóm**:
   * K-means có thể tự động phân nhóm các điểm dữ liệu mà không cần phải biết trước về phân bố dữ liệu hay các nhãn.

* **Nhược điểm của K-means:**

1. **Cần xác định số cụm K trước**:
   * Một trong những nhược điểm lớn nhất của K-means là bạn phải xác định số cụm K trước khi chạy thuật toán. Điều này có thể khó khăn nếu bạn không biết số lượng cụm chính xác trong dữ liệu.
2. **Nhạy cảm với khởi tạo centroid**:
   * K-means rất nhạy cảm với cách khởi tạo các centroid ban đầu. Nếu centroid ban đầu được chọn không hợp lý, thuật toán có thể hội tụ đến một kết quả kém. Các phương pháp như **K-means++** giúp khởi tạo centroid một cách thông minh hơn và giảm thiểu vấn đề này.
3. **Không hiệu quả với các cụm không đều hình dạng**:
   * K-means giả định rằng các cụm có hình dạng hình cầu (spherical clusters) và có cùng kích thước, vì vậy nó không hoạt động tốt với các cụm có hình dạng phức tạp hoặc không đều.
4. **Nhạy cảm với điểm ngoại lai (outliers)**:
   * K-means dễ bị ảnh hưởng bởi các điểm ngoại lai, vì centroid có thể bị kéo về phía các điểm ngoại lai, làm giảm độ chính xác của phân cụm.

* **Ứng dụng của K-means:**

1. **Phân cụm khách hàng**:
   * Trong marketing, K-means có thể được sử dụng để phân nhóm khách hàng dựa trên các đặc điểm như hành vi mua hàng, độ tuổi, thu nhập, v.v. Điều này giúp doanh nghiệp xác định các nhóm khách hàng tiềm năng để áp dụng các chiến lược marketing khác nhau.
2. **Phân tích ảnh**:
   * K-means có thể được sử dụng để phân nhóm các pixel trong ảnh, chẳng hạn như trong phân đoạn ảnh (image segmentation), nơi các vùng ảnh tương tự được nhóm lại với nhau.
3. **Phân tích văn bản**:
   * K-means có thể phân nhóm các tài liệu văn bản dựa trên nội dung của chúng, giúp tổ chức các tài liệu theo các chủ đề hoặc lĩnh vực.
4. **Nhận diện bất thường**:
   * K-means cũng có thể được sử dụng để phát hiện các điểm bất thường trong dữ liệu. Nếu một điểm dữ liệu không thuộc vào cụm nào, nó có thể là một điểm ngoại lai.

### Học máy bán giám sát

* **Khái niệm:**

Học bán giám sát là sự kết hợp giữa học có giám sát và học không giám sát. Phương pháp này tận dụng một lượng nhỏ dữ liệu có nhãn và một lượng lớn dữ liệu không nhãn để xây dựng mô hình.

* **Đặc điểm:**
* Dữ liệu có nhãn thường đắt đỏ hoặc khó thu thập, nhưng dữ liệu không nhãn lại dễ thu thập hơn.
* Học bán giám sát giúp cải thiện hiệu suất của mô hình khi dữ liệu có nhãn bị hạn chế.
* **Quy trình học bán giám sát:**

1. **Chuẩn bị dữ liệu**:
   * Gồm hai phần:
     + Dữ liệu có nhãn (X, Y): Nhỏ, chất lượng cao.
     + Dữ liệu không nhãn X': Lớn, phong phú.
2. **Huấn luyện sơ bộ**:
   * Sử dụng dữ liệu có nhãn để xây dựng mô hình ban đầu.
3. **Tận dụng dữ liệu không nhãn**:
   * Mô hình dự đoán nhãn cho dữ liệu không nhãn và sử dụng chúng để tiếp tục huấn luyện.
4. **Tối ưu hóa và đánh giá**:
   * Lặp lại quy trình để cải thiện hiệu suất.

* **Ứng dụng phổ biến trong học bán giám sát:**

1. **Nhận diện hình ảnh**:
   * Ví dụ: Nhận diện khuôn mặt khi chỉ một số hình ảnh được gắn nhãn.
2. **Phân loại văn bản**:
   * Ví dụ: Gắn nhãn chủ đề cho tài liệu với ít dữ liệu gắn nhãn.
3. **Xử lý ngôn ngữ tự nhiên (NLP)**:
   * Ví dụ: Dịch ngôn ngữ khi chỉ một số ít câu được dịch sẵn.

* **Thuật toán phổ biến trong học bán giám sát:**

1. **Self-training**:
   * Dùng mô hình ban đầu để dự đoán nhãn cho dữ liệu không nhãn, sau đó thêm vào tập huấn luyện.
2. **Co-training**:
   * Sử dụng nhiều mô hình độc lập để hỗ trợ gán nhãn cho dữ liệu không nhãn.
3. **Mô hình đồ thị (Graph-based Models)**:
   * Xây dựng đồ thị biểu diễn quan hệ giữa các điểm dữ liệu và học từ đồ thị này.

* **Tổng kết các phương pháp học máy:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Đặc điểm** | **Học có giám sát** | **Học không giám sát** | **Học bán giám sát** |
| **Dữ liệu yêu cầu** | Có nhãn | Không nhãn | Kết hợp có nhãn và không nhãn |
| **Mục tiêu** | Dự đoán, phân loại | Tìm cấu trúc dữ liệu | Tăng cường dự đoán từ ít dữ liệu nhãn |
| **Ưu điểm** | Độ chính xác cao | Không cần gắn nhãn | Tận dụng dữ liệu không nhãn |
| **Thách thức** | Phụ thuộc vào dữ liệu nhãn | Khó đánh giá kết quả | Cần cân đối giữa hai loại dữ liệu |

Bảng 1.1: Bảng so sánh các phương pháp học máy

# CHƯƠNG II: XÂY DỰNG HỆ THỐNG NHẬN DIỆN BIỂN BÁO GIAO THÔNG BẰNG CNN VÀ KERAS



## Thư viện

### Keras

**Keras** là một thư viện mã nguồn mở để xây dựng và huấn luyện các mô hình học sâu (Deep Learning). Trước đây, Keras là một API độc lập và có thể chạy trên các backend khác nhau như **TensorFlow**, **Theano**, và **CNTK**. Tuy nhiên, hiện tại, Keras được tích hợp trực tiếp vào **TensorFlow** và trở thành API chính của TensorFlow để xây dựng các mô hình học sâu.

Keras cung cấp giao diện đơn giản, dễ sử dụng và dễ dàng tùy chỉnh, giúp người dùng tạo ra các mô hình phức tạp một cách nhanh chóng mà không cần phải viết quá nhiều mã. Thư viện này được ưa chuộng bởi tính dễ sử dụng và khả năng tối ưu hóa tốt cho các mô hình học sâu.

* **Tính năng chính của Keras**

1. **Tích hợp tốt với TensorFlow**:
   * Keras hiện là phần của TensorFlow, hỗ trợ xây dựng các mô hình học sâu phức tạp và có thể sử dụng GPU.
2. **API dễ sử dụng**:
   * Keras cung cấp 3 cách chính để xây dựng mô hình: Sequential API, Functional API, và Subclassing API.
3. **Hỗ trợ các mô hình học sâu phổ biến**:
   * Hỗ trợ các mô hình học sâu phổ biến như mạng nơ-ron tích chập (CNN), mạng nơ-ron hồi tiếp (RNN), mạng nơ-ron đối kháng (GAN), và nhiều hơn nữa.
4. **Hỗ trợ huấn luyện hiệu quả**:
   * Các công cụ hỗ trợ huấn luyện mô hình với nhiều tùy chọn như tối ưu hóa, mất mát (loss), và metrics.
5. **Dễ dàng mở rộng và tùy chỉnh**:
   * Bạn có thể xây dựng các lớp, mô hình, và thuật toán tùy chỉnh một cách dễ dàng.

* **Ứng dụng thực tế của Keras**

1. **Nhận dạng ảnh**:
   * Sử dụng CNN để phân loại hình ảnh, ví dụ như phân loại chữ số viết tay (MNIST), nhận dạng khuôn mặt, phân loại động vật, v.v.
2. **Dự đoán chuỗi thời gian**:
   * Sử dụng RNN hoặc LSTM để dự đoán giá trị trong tương lai từ các chuỗi thời gian (ví dụ: dự đoán giá cổ phiếu).
3. **Nâng cao chất lượng hình ảnh**:
   * Sử dụng GANs (Generative Adversarial Networks) để tạo ra hình ảnh mới hoặc cải thiện chất lượng hình ảnh.
4. **Tạo chatbot và hệ thống đối thoại**:
   * Sử dụng mô hình seq2seq hoặc Transformer để tạo ra các chatbot hoặc hệ thống đối thoại.

* **Ưu điểm và nhược điểm của Keras**
* **Ưu điểm**
* **Dễ sử dụng và học hỏi**: Keras có giao diện rất dễ sử dụng và là lựa chọn phổ biến cho những người mới bắt đầu học về học sâu.
* **Tính khả mở**: Mặc dù dễ sử dụng, nhưng Keras cũng cho phép bạn tạo các mô hình phức tạp.
* **Tích hợp tốt với TensorFlow**: Keras dễ dàng được sử dụng cùng với TensorFlow để tận dụng GPU.
* **Nhược điểm**
* **Ít linh hoạt so với các thư viện khác**: Mặc dù Keras rất dễ sử dụng, nhưng đối với các mô hình quá phức tạp hoặc yêu cầu tính linh hoạt cao, Keras có thể không cung cấp đủ công cụ.

### Sklearn

**Scikit-learn (sklearn)** là một thư viện Python phổ biến và mạnh mẽ dành cho học máy (machine learning). Thư viện này cung cấp nhiều công cụ để xây dựng, huấn luyện, và đánh giá các mô hình học máy, bao gồm các thuật toán học có giám sát và không giám sát, đánh giá mô hình, và tiền xử lý dữ liệu.

* **Tính năng chính của Scikit-learn**

1. **Thuật toán học máy**:
   * Cung cấp nhiều thuật toán học máy có giám sát (supervised learning) và không giám sát (unsupervised learning), ví dụ: hồi quy, phân loại, phân cụm, giảm chiều dữ liệu.
2. **Tiền xử lý dữ liệu**:
   * Các công cụ tiền xử lý như chuẩn hóa, chuẩn hóa, mã hóa phân loại, phân chia dữ liệu thành tập huấn luyện và kiểm tra.
3. **Chọn mô hình và đánh giá**:
   * Cung cấp các công cụ để đánh giá mô hình (accuracy, precision, recall, F1-score, ROC AUC) và lựa chọn mô hình tốt nhất.
4. **Dễ dàng tích hợp với các thư viện khác**:
   * Tích hợp tốt với NumPy, Pandas, và các thư viện xử lý khoa học khác như Matplotlib.

* **Ứng dụng thực tế của Scikit-learn**

1. **Phân loại**:
   * Dự đoán nhãn cho các đối tượng (ví dụ: phân loại email là spam hay không spam).
2. **Hồi quy**:
   * Dự đoán giá trị liên tục (ví dụ: dự đoán giá trị cổ phiếu, doanh thu bán hàng).
3. **Phân cụm**:
   * Nhóm các đối tượng tương tự nhau (ví dụ: phân cụm khách hàng dựa trên hành vi mua sắm).
4. **Giảm chiều dữ liệu**:
   * Rút gọn số lượng đặc trưng mà vẫn giữ lại thông tin quan trọng (ví dụ: PCA - Principal Component Analysis).
5. **Tối ưu mô hình**:
   * Tuning các siêu tham số (hyperparameters) của mô hình để đạt hiệu suất tốt nhất.

### Matplotlib

**Matplotlib** là một thư viện được sử dụng để vẽ đồ thị, trực quan hóa dữ liệu. Nó cung cấp khả năng tạo các biểu đồ từ đơn giản đến phức tạp như biểu đồ đường, cột, phân tán, heatmap, 3D,...

* **Tính năng chính**
* **Hỗ trợ nhiều loại biểu đồ**:
  + Biểu đồ đường (**Line plots**).
  + Biểu đồ cột (**Bar plots**).
  + Biểu đồ tán xạ (**Scatter plots**).
  + Biểu đồ hình tròn (**Pie charts**).
  + Biểu đồ phân bố (**Histograms**).
  + Biểu đồ hộp (**Box plots**).
  + Đồ thị 3D.
* **Tùy chỉnh linh hoạt**:
  + Thêm nhãn, tiêu đề, chú thích.
  + Điều chỉnh trục, kích thước, màu sắc, kiểu đường.
* **Khả năng tích hợp tốt**:
  + Làm việc với NumPy, Pandas và các thư viện khác như seaborn để mở rộng tính năng.
* **Ứng dụng thực tế**

1. **Trực quan hóa dữ liệu trong phân tích**:
   * Hiển thị xu hướng (trend), phân phối (distribution), mối quan hệ (relationships).
2. **Phân tích thống kê**:
   * Biểu diễn phân bố dữ liệu, boxplot để kiểm tra outliers.
3. **Báo cáo và trình bày**:
   * Tạo đồ thị chuyên nghiệp cho các báo cáo và trình chiếu.
4. **Kết hợp với Seaborn**:
   * Tạo đồ thị đẹp mắt và giàu thông tin hơn với seaborn dựa trên matplotlib.

### Pil

**Pillow** là một thư viện Python mạnh mẽ để xử lý và thao tác hình ảnh. Nó là một phiên bản cải tiến và duy trì của **PIL (Python Imaging Library)**, cung cấp nhiều tính năng để xử lý hình ảnh như mở, chỉnh sửa, và lưu hình ảnh ở nhiều định dạng khác nhau.

* **Tính năng chính**

1. **Hỗ trợ nhiều định dạng ảnh**:
   * JPEG, PNG, BMP, GIF, TIFF, và nhiều định dạng khác.
2. **Xử lý hình ảnh**:
   * Cắt (cropping), thay đổi kích thước (resizing), xoay (rotating).
   * Chuyển đổi định dạng ảnh.
   * Thêm bộ lọc (filters) như làm mờ, sắc nét.
3. **Chỉnh sửa hình ảnh**:
   * Vẽ hình, chèn văn bản.
   * Điều chỉnh độ sáng, độ tương phản, màu sắc.
4. **Xử lý ảnh cấp độ pixel**:
   * Truy cập và chỉnh sửa từng pixel của ảnh.
5. **Tích hợp với NumPy**:
   * Chuyển đổi ảnh thành mảng NumPy để xử lý nhanh hơn.

* **Ứng dụng thực tế**

1. **Xử lý ảnh cơ bản**:
   * Cắt, thay đổi kích thước, xoay, chuyển đổi định dạng ảnh.
2. **Tự động hóa chỉnh sửa ảnh**:
   * Batch processing cho một số lượng lớn ảnh.
3. **Xử lý ảnh trong AI**:
   * Tiền xử lý ảnh trước khi đưa vào mô hình machine learning.
4. **Tạo watermark**:
   * Chèn watermark hoặc văn bản để bảo vệ bản quyền.
5. **Phát triển ứng dụng đồ họa**:
   * Dùng Pillow để tạo giao diện đồ họa hoặc xử lý ảnh đầu vào.

### Numpy

**NumPy** là một thư viện Python mạnh mẽ và phổ biến trong lĩnh vực khoa học dữ liệu và học máy. Nó cung cấp các công cụ để làm việc với mảng (arrays) và các phép toán đại số tuyến tính, xử lý ma trận, cũng như các phép toán toán học cấp cao một cách hiệu quả.

* **Tính năng chính của NumPy**

1. **Mảng n-dimensional (ndarray)**:
   * Cung cấp cấu trúc dữ liệu mảng đa chiều hiệu quả hơn so với danh sách Python thông thường.
   * Hỗ trợ các phép toán trên mảng nhanh chóng và tối ưu.
2. **Hàm toán học hiệu suất cao**:
   * Hỗ trợ các phép toán đại số tuyến tính (matrix multiplication, dot product, inverse matrix...).
   * Tích hợp các hàm toán học: sin, cos, exp, log, sqrt,...
3. **Xử lý mảng lớn**:
   * Thao tác với dữ liệu lớn một cách nhanh chóng nhờ cơ chế tính toán vector hóa (vectorization).
4. **Hỗ trợ random sampling**:
   * Sinh ngẫu nhiên các giá trị, mảng ngẫu nhiên từ các phân phối xác suất khác nhau.
5. **Tích hợp với các thư viện khác**:
   * NumPy là nền tảng cho nhiều thư viện khoa học và học máy khác như Pandas, Scikit-learn, TensorFlow, PyTorch,...

* **Ứng dụng thực tế**

1. **Xử lý dữ liệu lớn**:
   * NumPy tối ưu các phép toán mảng và ma trận để xử lý dữ liệu nhanh hơn so với Python thuần.
2. **Tiền xử lý dữ liệu**:
   * Chuẩn hóa, chuẩn bị dữ liệu trước khi đưa vào mô hình học máy.
3. **Phân tích thống kê**:
   * Tính các giá trị thống kê như trung bình, phương sai, độ lệch chuẩn.
4. **Học máy và AI**:
   * NumPy được dùng để xử lý đầu vào/đầu ra trong các mô hình học máy và deep learning.
5. **Xử lý hình ảnh**:
   * NumPy được sử dụng để chuyển đổi và xử lý ảnh dưới dạng ma trận pixel.

### Os

Thư viện **os** là một thư viện tiêu chuẩn trong Python, cung cấp giao diện để tương tác với hệ điều hành. Thông qua os, bạn có thể thực hiện các tác vụ như quản lý file, thư mục, môi trường hệ thống và nhiều chức năng liên quan đến hệ điều hành.

* **Tính năng chính của os**

1. **Quản lý file và thư mục**:
   * Tạo, đổi tên, xóa file hoặc thư mục.
   * Truy cập thông tin về file (kích thước, thời gian tạo, ...).
2. **Thao tác với đường dẫn**:
   * Ghép, tách, chuẩn hóa đường dẫn.
   * Xác định file hay thư mục.
3. **Tương tác với hệ thống**:
   * Thực thi lệnh hệ thống.
   * Lấy thông tin về môi trường.
4. **Thao tác với biến môi trường**:
   * Lấy hoặc thiết lập biến môi trường.

* **Ứng dụng thực tế**

1. **Quản lý file/thư mục tự động**:
   * Tạo, xóa, hoặc quản lý file theo lô.
2. **Phân tích hệ thống**:
   * Liệt kê file log hoặc thông tin file trong thư mục lớn.
3. **Tích hợp với biến môi trường**:
   * Sử dụng các biến môi trường để tùy chỉnh cấu hình ứng dụng.
4. **Thực thi lệnh shell**:
   * Chạy các lệnh hệ thống từ trong Python.
5. **Kiểm tra và xử lý lỗi hệ thống**:
   * Phát hiện và khắc phục các vấn đề liên quan đến file hoặc thư mục.

### Tkinter

**Tkinter** là thư viện giao diện đồ họa (GUI) tiêu chuẩn được tích hợp sẵn trong Python. Nó cung cấp các công cụ để xây dựng ứng dụng giao diện người dùng một cách nhanh chóng và dễ dàng, với các widget cơ bản như nút bấm, hộp nhập liệu, menu, danh sách, v.v.

* **Tính năng chính của Tkinter**

1. **Tích hợp sẵn**:
   * Không cần cài đặt thêm thư viện, Tkinter được đi kèm với Python (trừ một số phiên bản cần cấu hình trên Linux).
2. **Cung cấp widget phong phú**:
   * Button, Label, Entry, Text, Frame, Menu, Canvas, Scrollbar, v.v.
3. **Dễ sử dụng**:
   * Thích hợp cho người mới bắt đầu học lập trình GUI.
4. **Hỗ trợ tùy chỉnh**:
   * Tùy chỉnh giao diện, màu sắc, kích thước, font chữ, hành vi.

* **Ứng dụng thực tế**

1. **Ứng dụng nhỏ gọn**:
   * Tạo công cụ quản lý công việc, máy tính đơn giản, hoặc trình đọc file.
2. **Trò chơi 2D đơn giản**:
   * Sử dụng Canvas để xây dựng các trò chơi như ping-pong, bắn bóng.
3. **Ứng dụng tương tác**:
   * Giao diện để tương tác với cơ sở dữ liệu hoặc API.
4. **Công cụ học tập**:
   * Tạo flashcard, từ điển, hoặc ứng dụng trực quan hóa.

* **Ưu và nhược điểm của Tkinter**
* **Ưu điểm**
* **Dễ sử dụng**: Thích hợp cho người mới học lập trình GUI.
* **Đa nền tảng**: Chạy được trên Windows, macOS, Linux.
* **Được tích hợp sẵn**: Không cần cài đặt thêm.
* **Nhược điểm**
* **Giới hạn giao diện**: Giao diện cơ bản, không đẹp hoặc hiện đại như các thư viện khác (PyQt, Kivy).
* **Khả năng mở rộng hạn chế**: Không phù hợp cho các ứng dụng phức tạp hoặc giao diện đòi hỏi tính chuyên nghiệp.

## Xây dựng hệ thống nhận diện biển báo giao thông

### Vấn đề bài toán

Trong môi trường giao thông hiện đại, biển báo giao thông đóng vai trò quan trọng trong việc hướng dẫn và đảm bảo an toàn cho người tham gia giao thông. Tuy nhiên, với số lượng biển báo đa dạng và tình trạng giao thông phức tạp, việc tự động hóa nhận diện biển báo trở thành yếu tố cần thiết trong các hệ thống xe tự hành, cảnh báo giao thông, hoặc hỗ trợ người lái.

* **Mục tiêu:**
* Xây dựng một mô hình học sâu (deep learning) dựa trên mạng nơ-ron tích chập (CNN) để nhận diện các biển báo giao thông với độ chính xác cao.
* Sử dụng thư viện Keras để huấn luyện và triển khai mô hình, với khả năng phân loại đúng loại biển báo từ hình ảnh đầu vào.
* **Dữ liệu:**
* **Nguồn dữ liệu:** Sử dụng bộ dữ liệu công khai có sẵn German Traffic Sign Recognition. Tập dữ liệu chứa hơn 50.000 hình ảnh về các biển báo giao thông khác nhau, được phân loại thành 43 lớp. Dữ liệu phân chia không đều, với một số lớp có rất nhiều hình ảnh trong khi một số lớp lại có rất ít. Tập dữ liệu có một thư mục train chứa các hình ảnh bên trong mỗi lớp và một thư mục test chứa các hình ảnh dùng để thử nghiệm mô hình.

A screenshot of a graph

Description automatically generated

Hình 2.1: Bộ dữ liệu công khai German Traffic Sign Recognition

* **Đặc điểm dữ liệu:**
* Gồm các hình ảnh biển báo giao thông thuộc nhiều lớp (categories) khác nhau.
* Hình ảnh đa dạng về kích thước, góc chụp, điều kiện ánh sáng và chất lượng.
* **Yêu cầu bài toán:**
* Nhận diện các loại biển báo giao thông phổ biến như biển cấm, biển chỉ dẫn, biển nguy hiểm...
* Hệ thống cần xử lý được các yếu tố gây nhiễu trong hình ảnh như:
  + Độ mờ, nhiễu, hoặc chất lượng ảnh kém.
  + Góc nhìn thay đổi.
  + Các điều kiện môi trường (ánh sáng yếu, trời mưa...).

### Ý tưởng thực hiện

* **Phân tích bài toán:**
* **Input:** 1 ảnh kích thước 30x30.
* **Output:** Tên biển báo.
* **Dataset:** German Traffic Sign Recognition (hơn 50.000 ảnh và 43 lớp).
* **Tiền xử lý ảnh:** Resize ảnh, scale dữ liệu,...
* **Xây dựng và huấn luyện 1 mô hình CNN để phân loại ảnh.**
* **Các bước thực hiện:**

**A diagram of a diagram

Description automatically generated**

Hình 2.2: Các bước thực hiện bài toán

**1. Import dataset và xử lý hình ảnh**

* **Mục tiêu:** Chuẩn bị dữ liệu để sử dụng cho mô hình học sâu.
* **Thực hiện:**
  + Tải dữ liệu hình ảnh từ các thư mục được tổ chức theo lớp (labels).
  + Xử lý hình ảnh bao gồm:
    - Thay đổi kích thước tất cả hình ảnh về cùng kích thước (30x30).
    - Chuyển hình ảnh thành mảng NumPy để làm đầu vào cho mô hình.
  + Chuyển đổi nhãn (label) thành dạng one-hot encoding để mô hình dễ dàng hiểu được bài toán phân loại nhiều lớp.

**2. Xây dựng mô hình CNN**

* **Mục tiêu:** Thiết kế một mạng nơ-ron tích chập (CNN) để phân loại hình ảnh.
* **Thực hiện:**
* Sử dụng các lớp Conv2D (tích chập) để trích xuất đặc trưng không gian từ hình ảnh.
* Dùng MaxPooling2D để giảm kích thước không gian đặc trưng, tăng hiệu quả tính toán.
* Áp dụng Dropout để giảm hiện tượng overfitting.
* Thêm các lớp Dense (kết nối đầy đủ) ở cuối để thực hiện phân loại hình ảnh dựa trên đặc trưng đã trích xuất.
* Sử dụng hàm Softmax ở lớp đầu ra để mô hình trả về xác suất cho từng lớp.

**3. Training và xác thực**

* **Mục tiêu:** Huấn luyện mô hình trên tập dữ liệu huấn luyện và đánh giá trên tập kiểm tra.
* **Thực hiện:**
  + Huấn luyện mô hình qua nhiều epoch, sử dụng categorical\_crossentropy làm hàm mất mát và adam làm thuật toán tối ưu.
  + Theo dõi độ chính xác (accuracy) và mất mát (loss) trong cả quá trình huấn luyện và xác thực.
  + Lưu mô hình sau khi huấn luyện để sử dụng sau này.

**4. Kiểm tra mô hình với bộ dữ liệu test set**

* **Mục tiêu:** Đánh giá hiệu năng của mô hình trên tập dữ liệu kiểm tra độc lập.
* **Thực hiện:**
  + Chuẩn bị dữ liệu kiểm tra (Test Set) từ file Test.csv.
  + Sử dụng mô hình đã huấn luyện để dự đoán nhãn của các hình ảnh trong tập kiểm tra.
  + So sánh kết quả dự đoán với nhãn thực tế và tính độ chính xác (accuracy).

### Xây dựng hệ thống

* **Import dataset, thực hiện tiền xử lý ảnh:**

Thư mục train có chứa 43 thư mục con, mỗi thư mục đại diện cho một lớp khác nhau được đánh số thứ tự từ 0 đến 42. Sử dụng module os, ta sẽ lặp qua tất cả các lớp để kết nối các hình ảnh với nhãn tương ứng của chúng. Sử dụng thư viện PIL để chuyển hình ảnh thành một mảng.

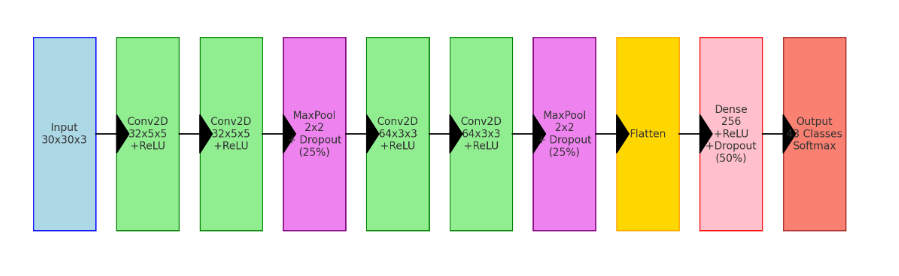
|  |
| --- |
| data = []  labels = []  classes = 43  cur\_path = os.getcwd()  #Retrieving the images and their labels  for i in range(classes):      path = os.path.join(cur\_path,'train',str(i))      images = os.listdir(path)      for a in images:          try:              image = Image.open(path + '\\'+ a)              image = image.resize((30,30))              image = np.array(image)              data.append(image)              labels.append(i)          except:              print("Error loading image")  #Converting lists into numpy arrays  data = np.array(data)  labels = np.array(labels) |

Sau đó, ta sẽ lưu tất cả các hình ảnh và nhãn của chúng vào các lists data và labels. Chúng ta cần chuyển các lists thành các numpy array để làm đầu vào cho model. Kích thước của dữ liệu là (39209, 30, 30, 3), có nghĩa là có 39209 hình ảnh, mỗi ảnh có kích thước 30x30 pixel, số 3 ở cuối thể hiện dữ liệu là hình ảnh màu (giá trị RGB). Sử dụng phương thức train\_test\_split() của sklearn để phân chia bộ dữ liệu thành train set và test set. Sử dụng phương thức to\_categorical của keras.utils để xử lý các nhãn của y\_train và y\_test bằng one-hot encoding.

|  |
| --- |
| print(data.shape, labels.shape)  #Splitting training and testing dataset  X\_train, X\_test, y\_train, y\_test = train\_test\_split(data, labels, test\_size=0.2, random\_state=43)  print(X\_train.shape, X\_test.shape, y\_train.shape, y\_test.shape)  #Converting the labels into one hot encoding  y\_train = to\_categorical(y\_train, 43)  y\_test = to\_categorical(y\_test, 43) |

* **Xây dựng mô hình CNN:**

Để phân loại các hình ảnh thành các danh mục tương ứng, chúng ta sẽ xây dựng mô hình CNN (Convolutional Neural Network). CNN là mô hình tốt nhất cho mục đích phân loại hình ảnh.



* + 1. **Input Layer (30x30x3)**
* Vai trò: Đây là nơi đầu vào hình ảnh được nhận.
* Thông số: Mỗi hình ảnh có kích thước 30x30 pixel và 3 kênh màu (RGB).
  + 1. **Convolutional Layers (Conv2D)**
* Conv2D 1 (32 filters, kernel size 5x5, activation ReLU)
  + Vai trò: Trích xuất các đặc trưng từ hình ảnh bằng cách áp dụng 32 bộ lọc kích thước 5x5.
  + ReLU: Loại bỏ giá trị âm trong các đặc trưng, giúp tăng tính phi tuyến.
* Conv2D 2 (32 filters, kernel size 5x5, activation ReLU)
  + Vai trò: Tiếp tục trích xuất thêm các đặc trưng sâu hơn với các bộ lọc tương tự.
    1. **Max Pooling Layer + Dropout**
* Max Pooling (2x2): Giảm kích thước của các đặc trưng (down-sampling) bằng cách lấy giá trị lớn nhất trong vùng 2x2.
* Dropout (25%): Giảm hiện tượng overfitting bằng cách loại bỏ ngẫu nhiên 25% các kết nối trong lớp.
  + 1. **Convolutional Layers (Conv2D)**
       - Conv2D 3 (64 filters, kernel size 3x3, activation ReLU)
  + Vai trò: Trích xuất các đặc trưng sâu hơn với số lượng bộ lọc lớn hơn (64) và kích thước nhỏ hơn (3x3).
* Conv2D 4 (64 filters, kernel size 3x3, activation ReLU)
  + Vai trò: Tiếp tục trích xuất thêm các đặc trưng phức tạp hơn.
    1. **Max Pooling Layer + Dropout**
* Max Pooling (2x2): Tiếp tục giảm kích thước đặc trưng.
* Dropout (25%): Tương tự như trước, giảm hiện tượng overfitting.
  + 1. **Flatten Layer**
* Chuyển đổi dữ liệu từ định dạng đa chiều (ma trận đặc trưng) thành một vector một chiều, để chuẩn bị cho các lớp fully connected (Dense).
  + 1. **Dense Layers(256 units, activation ReLU)**
  + Vai trò: Xây dựng mối quan hệ phi tuyến giữa các đặc trưng đã được trích xuất.
  + Dropout (50%): Loại bỏ ngẫu nhiên 50% các kết nối để giảm overfitting.
    1. **Output Layer (43 units, activation Softmax)**
  + Vai trò: Dự đoán xác suất của từng lớp (43 lớp biển báo).
  + Softmax: Biến đổi đầu ra thành xác suất, trong đó tổng xác suất bằng 1.
* **Kiến trúc mô hình CNN của chúng ta như sau:**
* **2 Conv2D layers** (filter=32, kernel\_size=(5,5), activation="relu")
* **MaxPool2D layer** (pool\_size=(2,2))
* **Dropout layer** (rate=0.25)
* **2 Conv2D layers** (filter=64, kernel\_size=(3,3), activation="relu")
* **MaxPool2D layer** (pool\_size=(2,2))
* **Dropout layer** (rate=0.25)
* **Flatten layer** để nén các lớp vào 1 chiều
* **Dense Fully connected layer** (256 nodes, activation="relu")
* **Dropout layer** (rate=0.5)
* **Dense layer** (43 nodes, activation="softmax")

|  |
| --- |
| #Building the model  model = Sequential()  model.add(Conv2D(filters=32, kernel\_size=(5,5), activation='relu', input\_shape=X\_train.shape[1:]))  model.add(Conv2D(filters=32, kernel\_size=(5,5), activation='relu'))  model.add(MaxPool2D(pool\_size=(2, 2)))  model.add(Dropout(rate=0.25))  model.add(Conv2D(filters=64, kernel\_size=(3, 3), activation='relu'))  model.add(Conv2D(filters=64, kernel\_size=(3, 3), activation='relu'))  model.add(MaxPool2D(pool\_size=(2, 2)))  model.add(Dropout(rate=0.25))  model.add(Flatten())  model.add(Dense(256, activation='relu'))  model.add(Dropout(rate=0.5))  model.add(Dense(43, activation='softmax'))  #Compilation of the model  model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy']) |

* **Training và xác thực:**

Sau khi xây dựng kiến trúc mô hình, chúng ta sẽ huấn luyện mô hình bằng cách sử dụng model.fit(). Chúng ta sẽ thử với các giá trị batch\_size là 32 và 64, và 15 epochs. Cuối cùng, chúng ta sẽ lưu mô hình đã đào tạo bằng cách sử dụng hàm model.save() của Keras.

|  |
| --- |
| epochs = 15  history = model.fit(X\_train, y\_train, batch\_size=64, epochs=epochs, validation\_data=(X\_test, y\_test))  model.save("my\_model.h5") |

Sử dụng matplotlib để vẽ biểu đồ tương quan giữa độ chính xác và sự mất mát.

|  |
| --- |
| #plotting graphs for accuracy  plt.figure(0)  plt.plot(history.history['accuracy'], label='training accuracy')  plt.plot(history.history['val\_accuracy'], label='val accuracy')  plt.title('Accuracy')  plt.xlabel('epochs')  plt.ylabel('accuracy')  plt.legend()  plt.show()  plt.figure(1)  plt.plot(history.history['loss'], label='training loss')  plt.plot(history.history['val\_loss'], label='val loss')  plt.title('Loss')  plt.xlabel('epochs')  plt.ylabel('loss')  plt.legend()  plt.show() |

* **Kiểm tra mô hình với bộ dữ liệu test set:**

Tập dữ liệu của chúng ta có chứa một thư mục test và một tệp test.csv. Chúng ta cần phải trích xuất hình ảnh và nhãn tương ứng bằng cách sử dụng pandas. Sau đó, để dự đoán mô hình, chúng ta phải thay đổi kích thước hình ảnh thành 30 × 30 pixel và tạo một mảng numpy chứa tất cả dữ liệu hình ảnh. Sử dụng precision\_score của sklearn.metrics để đánh giá các nhãn của bộ test.

|  |
| --- |
| #testing accuracy on test dataset  from sklearn.metrics import accuracy\_score  y\_test = pd.read\_csv('Test.csv')  labels = y\_test["ClassId"].values  imgs = y\_test["Path"].values  data=[]  for img in imgs:      image = Image.open(img)      image = image.resize((30,30))      data.append(np.array(image))  X\_test = np.array(data)  pred = np.argmax(model.predict(X\_test), axis=-1)  #Accuracy with the test data  from sklearn.metrics import accuracy\_score  print(accuracy\_score(labels, pred)) |

# CHƯƠNG III: THỰC NGHIỆM



## Dữ liệu

* **Dữ liệu**
* Dữ liệu được sử dụng trong bài toán là bộ dữ liệu Traffic Sign Recognition. Sử dụng thư mục “train” để thực hiện huấn luyện gồm 43 lớp (0 đến 42)
* **Tiền xử lý dữ liệu**

1. Đọc ảnh từ thư mục

* Sử dụng thư viện PIL để mở ảnh từ các thư mục.
* Đọc tất cả ảnh từ thư mục train tương ứng với từng lớp (0 đến 42).

1. Resize ảnh

* Tất cả ảnh được chuyển đổi về kích thước chuẩn **30x30 pixel** để đảm bảo đồng nhất kích thước khi đưa vào mô hình.

1. Chuyển đổi ảnh thành mảng numpy

* Ảnh sau khi được resize sẽ được chuyển đổi thành mảng numpy với 3 kênh màu (RGB).

1. Gán nhãn

* Ảnh từ mỗi thư mục được gắn nhãn tương ứng với tên thư mục (lớp từ 0 đến 42).
* Nhãn này được lưu vào danh sách labels.
* **Chia dữ liệu train-test**

Sử dụng train\_test\_split từ thư viện sklearn để chia dữ liệu thành hai tập:

* Tập huấn luyện (X\_train, y\_train):
  + 80% dữ liệu ban đầu, được dùng để huấn luyện mô hình.
* Tập kiểm tra (X\_test, y\_test):
  + 20% dữ liệu ban đầu, được dùng để đánh giá mô hình sau khi huấn luyện.
* One-hot encoding cho nhãn:
  + Nhãn được chuyển đổi sang định dạng one-hot encoding để phù hợp với mô hình phân loại đa lớp.
  + Sử dụng hàm **to\_categorical** từ Keras

## Độ đánh giá

* **Thực hiện tính toán Accuracy (độ chính xác) và Loss ( độ mất mát)**
* **Mục đích:**
* Đánh giá mô hình trên tập dữ liệu kiểm tra (validation set) sau mỗi epoch huấn luyện.
* Giúp xác định xem mô hình có bị overfitting hoặc underfitting không:
  + Overfitting: Mô hình hoạt động tốt trên dữ liệu huấn luyện nhưng kém trên dữ liệu kiểm tra.
  + Underfitting: Mô hình không học đủ từ dữ liệu huấn luyện.
* **Cách tính Accuracy và Loss**

Accuracy

A black text on a white background

Description automatically generated

Loss

A black and white text

Description automatically generated

* **So sánh kết quả giữa batch size 32 và 64 khi thực hiện training.**

|  |  |  |
| --- | --- | --- |
|  | **Batch size 32** | **Batch size 64** |
| **Huấn luyện** | **A screen shot of a computer  Description automatically generated** |  |
| **Mô hình accuracy** | **A graph of a line  Description automatically generated with medium confidence** |  |
| **Mô hình loss** |  |  |

Bảng 3.1: bảng so sánh kết quả **giữa batch size 32 và 64 khi thực hiện training**

**=> Có thể thấy mô hình hoạt động tốt hơn với batch size 64.**

* **Kết quả khi kiểm tra mô hình với test set là 95%.**

**A green line on a black background

Description automatically generated**

Hình 3.1: **Kết quả khi kiểm tra mô hình với test set**

## Kết quả thực nghiệm

Xây dựng 1 giao diện người dùng cho phép người dùng tải ảnh và thực hiện nhận dạng biển báo. Dưới đây là 1 số hình ảnh thực nghiệm nhận dạng biển báo:

A screenshot of a computer

Description automatically generated

Hình 3.2: Hình ảnh thực nghiệm nhận diện biển báo giới hạn 30km/h

A screenshot of a computer

Description automatically generated

Hình 3.3: Hình ảnh thực nghiệm nhận diện biển báo chỉ dẫn đi bên phải

A screenshot of a computer

Description automatically generated

Hình 3.4: Hình ảnh thực nghiệm nhận diện biển báo cấm vượt

A screenshot of a computer

Description automatically generated

Hình 3.5: Hình ảnh thực nghiệm nhận diện biển báo cấm xe cộ

A screenshot of a computer

Description automatically generated

Hình 3.6: Hình ảnh thực nghiệm nhận diện biển báo giới hạn tốc độ 80km/h

A screenshot of a computer

Description automatically generated

Hình 3.7: Hình ảnh thực nghiệm nhận diện biển báo đường ưu tiên

# DANH MỤC THAM KHẢO

1. Sách Deep Learing cơ bản (Tái bản lần 2) – Nguyễn Thanh Tuấn
2. Sách Machine Learning cơ bản – Vũ Hữu Tiệp
3. Trang web viblo.asia/newest [ Tham khảo ngày 5-11-2024 ]
4. Trang web cloudfly.vn[ Tham khảo ngày 5-11-2024 ]
5. Trang web tailieubkhn.com [ Tham khảo ngày 5-11-2024 ]
6. Trang web studocu.com [ Tham khảo ngày 16-11-2024 ]
7. Trang web thuvientvc.files.wordpress.com [ Tham khảo ngày 26-11-2024 ]
8. Trang web tailieu.vn [ Tham khảo ngày 1-12-2004 ]