**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** | 20210567 | Trần Đình Huy | DCCNTT12.10.2 |
| **2** | 20210617 | Trương Anh Thi | DCCNTT12.10.2 |
| **3** | 20210627 | Nguyễn Đình Nam | DCCNTT12.10.2 |

**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** | 20210567 | Trần Đình Huy | DCCNTT12.10.2 |
| **2** | 20210617 | Trương Anh Thi | DCCNTT12.10.2 |
| **3** | 20210627 | Nguyễn Đình Nam | DCCNTT12.10.2 |

|  |  |
| --- | --- |
|  |  |

**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 : Xây Dựng Hệ Thống Nhận Diện Biển Báo Giao Thông Bằng Cnn Và Keras**  **Tên học phần:**  **XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ:**  **XATGMT.03.K12.02.LH.C04.1\_LT.1\_TH** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)*  **Lương Thị Hồng Lan** | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Trần Đình Huy** | **Nguyễn Đình Nam** | **Trương Anh Thi** |
| --- | --- | --- | --- | --- | --- |
| 20210567 | 20210627 | 20210617 |
| **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 NÓI ĐẦU** 6](#_Toc184477729)

[**LỜI CẢM ƠN** 7](#_Toc184477730)

[**CHƯƠNG 1 : TỔNG QUAN** 8](#_Toc184477731)

[**1.1. Nhận dạng đối tượng** 8](#_Toc184477732)

1.1.1. [Nhận dạng đối tượng là gì ? 8](#_Toc184477733)

[1.1.2. Những vấn đề cần giải quyết trong bài toán 9](#_Toc184477734)

[**1.2**. **Các phương pháp áp dụng cho bài toán nhận dạng** 10](#_Toc184477735)

[1.2.1. Phương pháp dựa trên Nơ-ron 10](#_Toc184477736)

[1.2.2. Phương pháp Yolo 11](#_Toc184477737)

[**1.3. Ngôn ngữ lập trình và các thư viện** 12](#_Toc184477738)

[1.3.1. Python 12](#_Toc184477739)

[1.3.2. Giới thiệu về các thư viện và thuật toán 14](#_Toc184477740)

[**CHƯƠNG 2 : XÂY DỰNG HỆ THỐNG NHẬN DẠNG BÀI TOÁN** 22](#_Toc184477741)

[**2.1. Yêu cầu của bài toán nhận dạng biển báo giao thông:** 22](#_Toc184477742)

[2.1.1. Phân tích bài toán 22](#_Toc184477743)

[**2.2. Sơ đồ khối tổng quát** 24](#_Toc184477744)

[**CHƯƠNG 3 : THỰC NGHIỆM** 26](#_Toc184477745)

[**3.1. Dữ liệu** 26](#_Toc184477746)

[3.1.1. Tiền xử lý và phân chia dữ liệu 29](#_Toc184477747)

[**3.2. Các độ so sánh** 30](#_Toc184477748)

[**3.3. Kết quả thực nghiệm** 32](#_Toc184477749)

[3.3.1. Tạo giao diện người dùng 34](#_Toc184477750)

[**CHƯƠNG 4 : KẾT LUẬN** 39](#_Toc184477751)

[**TÀI LIỆU THAM KHẢO** 40](#_Toc184477752)

**LỜI NÓI ĐẦU**

Trong bối cảnh công nghệ ngày càng phát triển, các ứng dụng của Xử lý ảnh và Thị giác máy tính đang dần trở nên phổ biến trong nhiều lĩnh vực như an ninh, giám sát, y tế, giao thông và giải trí. Với mục tiêu cung cấp các giải pháp thông minh và tự động hóa, việc xây dựng hệ thống theo dõi đối tượng trong video đã và đang thu hút sự quan tâm lớn từ cộng đồng nghiên cứu cũng như ứng dụng thực tế.

Bài tập lớn này được thực hiện nhằm vận dụng các kiến thức lý thuyết và kỹ thuật đã học trong môn Xử lý ảnh và Thị giác máy tính để xây dựng một hệ thống nhận diện biển báo giao thông. Đề tài này không chỉ giúp sinh viên nắm vững các phương pháp cơ bản như xử lý khung hình, phát hiện đối tượng, và theo dõi chuyển động mà còn mở ra cơ hội tiếp cận với các thuật toán hiện đại như CNN và thư viện Keras.

Hệ thống được xây dựng sẽ có tính năng : phát hiện nhận diện các loại biển báo giao thông . Các công cụ và ngôn ngữ lập trình như Python, OpenCV, TensorFlow sẽ được sử dụng để triển khai hệ thống.

Với đề tài " Xây dựng hệ thống nhận diện biển báo giao thông bằng CNN và Keras", nhóm chúng em hy vọng không chỉ hoàn thành mục tiêu của bài tập lớn mà còn tích lũy được các kỹ năng và kinh nghiệm thực tế để áp dụng trong các dự án thực tế sau này.

**LỜI CẢM ƠN**

Trước tiên, nhóm chúng tôi xin gửi lời cảm ơn chân thành đến Bộ môn Xử lý ảnh và Thị giác máy tính, Khoa Công nghệ Thông tin, Trường Đại học Công Nghệ Đông Á, đã tạo điều kiện và cung cấp nền tảng kiến thức vững chắc để chúng tôi thực hiện bài tập lớn này.

Đặc biệt, chúng em xin bày tỏ lòng biết ơn sâu sắc tới cô Lương Thị Hồng Lan vì đã tận tình giảng dạy, định hướng, và hỗ trợ chúng tôi trong suốt quá trình học tập và triển khai đề tài. Sự hướng dẫn tận tình của thầycô không chỉ giúp nhóm hiểu sâu hơn về lý thuyết mà còn ứng dụng hiệu quả vào bài tập thực tế.

Chúng em cũng xin gửi lời cảm ơn đến các thành viên trong nhóm, vì sự phối hợp nhịp nhàng và tinh thần làm việc trách nhiệm đã góp phần giúp hoàn thành bài tập lớn này.

Mặc dù đã nỗ lực hết sức, nhưng bài tập lớn không thể tránh khỏi những thiếu sót. Chúng em mong nhận được sự góp ý từ thầy cô và các bạn để có thể hoàn thiện hơn trong các dự án tương lai.

Trân trọng cảm ơn!

# **CHƯƠNG 1 : TỔNG QUAN**

* 1. **. Nhận dạng đối tượng**
     1. **Nhận dạng đối tượng là gì ?**

Nhận dạng đối tượng (Object Recognition) là một kỹ thuật trong lĩnh vực thị giác máy tính (Computer Vision) và trí tuệ nhân tạo (AI), nhằm mục đích xác định và phân loại các đối tượng trong một hình ảnh hoặc video. Công nghệ này cho phép máy tính hiểu và xác định những gì đang hiện diện trong nội dung hình ảnh dựa trên các thuật toán học máy và mạng nơ-ron.



*Hình 1.1: Hình ảnh minh họa quá trình nhận diện hình ảnh biển báo giao thông*

* + 1. **Những vấn đề cần giải quyết trong bài toán**

Nhận dạng biển báo giao thông là một bài toán phức tạp trong lĩnh vực thị giác máy tính, và khi thực hiện, cần giải quyết những vấn đề sau :

**Đa dạng và thiết kế biển báo**

Biển báo giao thông có nhiều loại, kích thước, màu sắc và hình dạng khác nhau. Sự đa dạng này làm tăng độ khó trong việc nhận dạng chính xác. Cần sử dụng các mô hình mạnh như Convolutional Neural Networks (CNC) để học các đặc trưng đa dạng. Tăng cường dữ liệu bằng cách áp dụng các kỹ thuật như xoay, lật, di chuyển và thay đổi độ sáng hình ảnh.

**Điều kiện môi trường**

Hình ảnh biển báo có thể bị che khuất bởi cây cối, xe cộ hoặc các yếu tố khác, bên cạnh đó điều kiện thời tiết và thay đổi góc nhìn cũng ảnh hưởng đến độ rõ nét. Sử dụng kỹ thuật tăng cường độ tương phản hoặc giảm nhiễu, huấn luyện mô hình trên các tập dữ liệu có chứa nhiều trường hợp thực tế để tăng khả năng khái quát.

**Kích thước biển báo**

Biển báo giao thông trong hình ảnh có thể rất nhỏ so với toàn bộ khung hình, đặc biệt trong các bức chụp từ xa. Sử dụng các phương pháp phát hiện đối tượng như YOLO, SSD hoặc Faster R-CNC để định vị biển báo trước khi nhận dạng, tăng độ phân giải ảnh hoặc sử dụng công cụ phóng to vùng chứa biển báo

**Sự tương đồng giữa các biển báo**

Nhiều loại biển báo giao thông rất giống nhau nhưng lại mang ý nghĩa khác nhau, mô hình phải được huấn luyện với độ chính xác cao, tập trung vào các đặc điểm nhỏ, chẳng hạn như chữ số hoặc ký hiệu cụ thể, tăng độ phân giải của hình ảnh và chú trọng vào các vùng quan trọng trong ảnh.

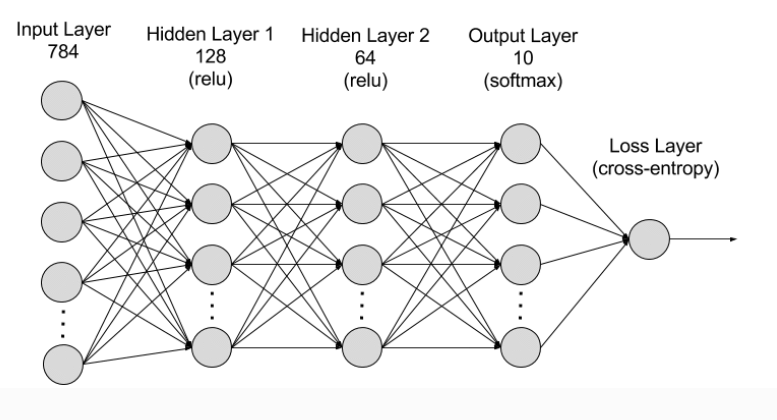
**Dữ liệu không cân bằng**

Một số biển báo xuất hiện thường xuyên hơn các loại khác, dẫn đến dữ liệu không đồng đều, làm ảnh hưởng đến hiệu suất nhận dạng của mô hình, sử dụng kỹ thuật lấy mẫu lại để cân bằng dữ liệu hoặc tăng cường dữ liệu cho các biển báo ít xuất hiện.

* 1. **.Các phương pháp áp dụng cho bài toán nhận dạng**

**1.2.1. Phương pháp dựa trên Nơ-ron**

Mạng nơ-ron là mô hình mô phỏng cách hoạt động của bộ não con người, với các nơ-ron (neurons) kết nối thành các lớp (layers).



*Hình 1.2 Kiến trúc mạng nơ ron*

**Các loại lớp chính:**

**Input Layer (Lớp đầu vào):** Nhận dữ liệu (ảnh biển báo giao thông) dưới dạng vector hoặc ma trận.

**Hidden Layers (Lớp ẩn):** Gồm nhiều tầng, thực hiện các phép tính và phát hiện đặc trưng.

Output Layer (Lớp đầu ra): Đưa ra dự đoán (ví dụ: loại biển báo giao thông).

**Mạng nơ-ron tích chập (CNN):** Một dạng ANN chuyên biệt cho dữ liệu hình ảnh, nổi bật nhờ khả năng tự động học các đặc trưng hình ảnh mà không cần trích xuất thủ công.

**Kiến trúc CNN**

CNN là phương pháp phổ biến nhất khi áp dụng mạng nơ-ron vào nhận diện biển báo giao thông. Các thành phần chính của CNN bao gồm:

**Tầng tích chập (Convolutional Layers):**

Phát hiện các đặc trưng cục bộ như cạnh, góc, hình dạng trong ảnh.

Sử dụng các kernel (ma trận con) để quét qua toàn bộ ảnh.

Kết quả là các "bản đồ đặc trưng" (feature maps).

**Tầng gộp (Pooling Layers):**

Giảm chiều dữ liệu và độ phức tạp tính toán.

Phổ biến nhất là MaxPooling (lấy giá trị lớn nhất trong vùng quét).

**Tầng dropout (Dropout Layer):**

Ngẫu nhiên tắt một số nơ-ron trong quá trình huấn luyện để giảm overfitting.

**Tầng kết nối đầy đủ (Fully Connected Layers):**

Kết nối toàn bộ các nơ-ron từ tầng trước.

Chuyển đặc trưng trích xuất thành dự đoán.

**Hàm kích hoạt (Activation Function):**

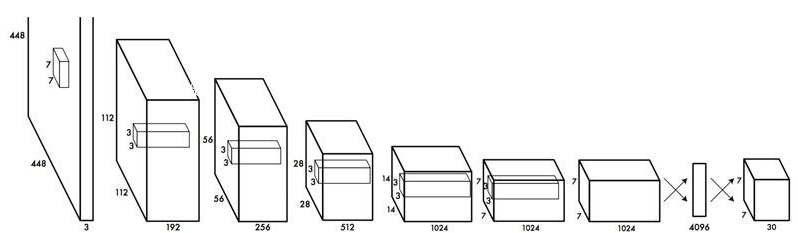
Tạo phi tuyến tính trong mô hình, thường dùng hàm ReLU (Rectified Linear Unit).

**Tầng Softmax:**

Biến đầu ra thành xác suất phân phối để phân loại nhiều lớp (multi-class classification).

**1.2.2. Phương pháp Yolo**

YOLO là một kỹ thuật phát hiện đối tượng theo thời gian thực được phát triển bởi Joseph Redmon và các cộng sự. Khác với các phương pháp truyền thống (như R-CNN hoặc Fast R-CNN), YOLO thực hiện phát hiện và phân loại đối tượng trong một lần quét (single pass).



*Hình 1.3 Kiến trúc YOLO và cơ chế hoạt động*

**Đặc điểm nổi bật:**

**Nhanh:** YOLO đạt tốc độ phát hiện đối tượng thời gian thực.

**Toàn diện:** Mô hình xử lý toàn bộ hình ảnh trong một lần, giảm thiểu các lỗi trùng lặp.

**Tổng quát hóa tốt:** YOLO hoạt động hiệu quả trên nhiều loại đối tượng.

**Kiến trúc YOLO**

YOLO sử dụng kiến trúc dựa trên CNN, nhưng có những đặc điểm riêng:

**Phân chia ảnh thành lưới (Grid):**

Chia ảnh đầu vào thành một lưới S×SS \times SS×S.

Mỗi ô trong lưới chịu trách nhiệm phát hiện đối tượng nằm ở tâm của ô.

**Dự đoán bounding boxes:**

Mỗi ô trong lưới dự đoán BBB hộp giới hạn (bounding boxes).

**Mỗi hộp bao gồm:**

x,yx, yx,y: Tọa độ tâm của đối tượng (được chuẩn hóa theo kích thước ảnh).

w,hw, hw,h: Chiều rộng và chiều cao của hộp giới hạn (cũng được chuẩn hóa).

pcp\_cpc​: Độ tin cậy (confidence) về sự tồn tại của đối tượng trong hộp.

**Phân loại đối tượng:**

Với mỗi ô, YOLO cũng dự đoán xác suất CCC cho từng lớp đối tượng (dựa trên Softmax).

**Hàm mất mát (Loss Function):**

**Bao gồm ba thành phần:**

Mất mát định vị (Localization Loss): Sai số giữa hộp dự đoán và hộp thật.

Mất mát độ tin cậy (Confidence Loss): Sai số của giá trị pcp\_cpc​.

Mất mát phân loại (Classification Loss): Sai số giữa lớp dự đoán và lớp thực tế.

* 1. **. Ngôn ngữ lập trình và các thư viện**

**1.3.1. Python**

Python là một **ngôn ngữ lập trình bậc cao, thông dịch và đa năng** (general-purpose programming language) được tạo ra bởi **Guido van Rossum** vào năm 1991. Python được thiết kế với triết lý đơn giản, dễ đọc và dễ học, nhờ cú pháp rõ ràng và gần gũi với ngôn ngữ tự nhiên.



*Hình 1.4. Ngôn ngữ lập trình Python*

**Đặc điểm nổi bật của Python**

**Dễ học và sử dụng:** Cú pháp đơn giản, dễ đọc, không yêu cầu nhiều ký tự phức tạp như các ngôn ngữ khác (ví dụ: không cần dấu chấm phẩy ;). Phù hợp cho người mới bắt đầu học lập trình.

**Đa nền tảng:** Python hoạt động trên nhiều hệ điều hành, bao gồm Windows, macOS và Linux.

**Ngôn ngữ thông dịch:** Mã Python được thực thi từng dòng bởi trình thông dịch (interpreter), giúp dễ dàng thử nghiệm và kiểm tra mã.

**Mạnh mẽ và linh hoạt:** Python hỗ trợ nhiều kiểu lập trình như lập trình hướng đối tượng (OOP), lập trình thủ tục và lập trình hàm (functional programming).

**Thư viện phong phú:** Python đi kèm với thư viện tiêu chuẩn lớn và một kho thư viện bên ngoài phong phú (PyPI), hỗ trợ các lĩnh vực như: Khoa học dữ liệu (NumPy, pandas, Matplotlib). Trí tuệ nhân tạo (TensorFlow, PyTorch). Phát triển web (Django, Flask). Tự động hóa (Selenium, Beautiful Soup).

**Ứng dụng:**

Python được sử dụng rộng rãi trong các lĩnh vực:

**Khoa học dữ liệu và trí tuệ nhân tạo:** Phân tích dữ liệu, xây dựng mô hình học máy.

**Phát triển web:** Xây dựng ứng dụng web với các framework như Django và Flask.

**Tự động hóa:** Viết các script để tự động hóa công việc lặp đi lặp lại.

**Xử lý hình ảnh và thị giác máy tính:** Phân tích và xử lý hình ảnh.

**Ứng dụng khoa học và kỹ thuật:** Hỗ trợ nghiên cứu và tính toán kỹ thuật.

* + 1. **Giới thiệu về các thư viện và thuật toán**

**a. TensorFlow** là một nền tảng mã nguồn mở mạnh mẽ dành cho **máy học(Machine Learning)** và **trí tuệ nhân tạo (AI)**, được phát triển bởi **Google Brain** vào năm 2015. TensorFlow hỗ trợ việc xây dựng, huấn luyện và triển khai các mô hình học máy trên nhiều loại thiết bị, từ máy tính cá nhân, máy chủ đến thiết bị di động và nhúng.

**Đặc điểm nổi bật của TensorFlow:**

**Mã nguồn mở:** TensorFlow được cung cấp miễn phí và có một cộng đồng lớn, giúp mở rộng và phát triển nhanh chóng.

**Linh hoạt và mở rộng:** Hỗ trợ nhiều cấp độ trừu tượng, từ xây dựng mô hình đơn giản bằng Keras đến tùy chỉnh chi tiết thông qua các API cấp thấp.

**Khả năng đa nền tảng:** TensorFlow có thể chạy trên CPU, GPU, TPU (Tensor Processing Unit) và hỗ trợ nhiều hệ điều hành như Windows, macOS và Linux.

**Hỗ trợ các công cụ và thư viện mạnh mẽ:** Cung cấp các công cụ như **TensorBoard** (trực quan hóa mô hình), **TensorFlow Lite** (triển khai trên thiết bị di động) và **TensorFlow.js** (triển khai trên trình duyệt web).

**Khả năng mở rộng cho sản xuất:** TensorFlow dễ dàng triển khai các mô hình học máy vào hệ thống sản xuất với khả năng hỗ trợ tại quy mô lớn.

**Ứng dụng của TensorFlow:**

**Học máy và học sâu:** TensorFlow được sử dụng để xây dựng các mô hình học máy (Machine Learning) và học sâu (Deep Learning), bao gồm:

Mạng nơ-ron nhân tạo (ANN).

Mạng nơ-ron tích chập (CNN) cho xử lý ảnh.

Mạng nơ-ron hồi tiếp (RNN) cho xử lý chuỗi thời gian.

**Xử lý ngôn ngữ tự nhiên (NLP):** Phát triển các ứng dụng như dịch ngôn ngữ, chatbot, và phân tích cảm xúc.

**Thị giác máy tính (Computer Vision):** Nhận diện khuôn mặt, phân loại hình ảnh, và phát hiện vật thể.

**Dữ liệu lớn và dự đoán:** Sử dụng TensorFlow để xây dựng các hệ thống dự đoán dựa trên dữ liệu lớn.

**Ứng dụng trong đời sống:** Triển khai trong các lĩnh vực như y tế, tài chính, thương mại điện tử, và tự động hóa.

**b. Numpy** (Numerical Python) là một thư viện mã nguồn mở mạnh mẽ dành cho **Python**, được sử dụng chủ yếu để xử lý các phép toán khoa học, tính toán số học, và thao tác với **mảng đa chiều**. NumPy được phát triển bởi **Travis Oliphant** vào năm 2005 là nền tảng của nhiều thư viện khác như SciPy, pandas, và Matplotlib.

**Đặc điểm nổi bật của NumPy:**

**Hỗ trợ mảng đa chiều:** NumPy cung cấp đối tượng ndarray để lưu trữ và thao tác với các mảng dữ liệu nhiều chiều một cách hiệu quả.

**Hiệu suất cao:** Các phép toán trong NumPy được viết bằng ngôn ngữ C, giúp tăng tốc độ tính toán so với việc sử dụng danh sách thông thường trong Python.

**Thư viện hàm toán học phong phú:** NumPy hỗ trợ các phép toán như cộng, trừ, nhân ma trận, biến đổi Fourier, đại số tuyến tính, và nhiều hàm thống kê.

**Khả năng thao tác dữ liệu mạnh mẽ:** Dễ dàng thêm, xóa, hoặc thay đổi kích thước của mảng, cũng như truy xuất dữ liệu theo các cách linh hoạt.

**Tương thích với các thư viện khác:** NumPy được sử dụng rộng rãi trong các thư viện khoa học dữ liệu như pandas, Matplotlib, SciPy, TensorFlow, v.v.

**Ứng dụng của NumPy:**

**Xử lý dữ liệu số:** NumPy giúp thao tác với dữ liệu số học nhanh chóng, từ các mảng đơn giản đến các ma trận phức tạp.

**Đại số tuyến tính:** Thực hiện các phép toán ma trận, giải hệ phương trình tuyến tính, và tìm giá trị riêng, vector riêng.

**Khoa học dữ liệu và học máy:** Là nền tảng cho các thư viện học máy như TensorFlow và PyTorch, giúp xử lý dữ liệu trước khi đưa vào các mô hình.

**Xử lý hình ảnh:** NumPy được sử dụng để biểu diễn hình ảnh dưới dạng mảng số, hỗ trợ các phép biến đổi như xoay, co giãn và làm mờ.

**Mô phỏng và phân tích khoa học:** Dùng để mô phỏng hiện tượng tự nhiên, xử lý tín hiệu và phân tích dữ liệu khoa học.

**c. Keras** là một thư viện mã nguồn mở được xây dựng trên nền tảng Python, chuyên dùng để xây dựng và huấn luyện các mô hình học sâu (deep learning). Thư viện này được thiết kế để dễ sử dụng, có khả năng mở rộng cao và hiệu suất tốt, phục vụ cho các nhà nghiên cứu cũng như các nhà phát triển ứng dụng.

**Đặc điểm nổi bật của Keras:**

**Dễ sử dụng:**

Giao diện thân thiện với người dùng, rõ ràng và dễ học.

Keras giúp người dùng tập trung vào xây dựng mô hình mà không bị phân tâm bởi các chi tiết kỹ thuật phức tạp.

**Linh hoạt :**

Hỗ trợ nhiều loại kiến trúc mạng như **MLP (Perceptron đa lớp)**, **CNN (Mạng nơ-ron tích chập)**, và **RNN (Mạng nơ-ron hồi quy)**.

Dễ dàng kết hợp các lớp tùy chỉnh hoặc sử dụng các hàm kích hoạt tự định nghĩa.

**Hỗ trợ nhiều backend :** Ban đầu, Keras hỗ trợ các backend như TensorFlow, Theano, và CNTK. Tuy nhiên, kể từ phiên bản 2.3 trở đi, Keras tập trung hoàn toàn vào TensorFlow và được tích hợp trong TensorFlow dưới tên tf.keras.

**Khả năng mở rộng :** Keras có thể mở rộng để tạo ra các mô hình phức tạp và tùy chỉnh, thích hợp cho cả nghiên cứu và phát triển công nghiệp.

**Ứng dụng của Keras :**

**Xử lí ảnh:** Phân loại, nhận dạng đối tượng, trích xuất đặc trưng.

**Xử lí ngôn ngữ tự nhiên (NLP):** Phân tích cảm xúc, dịch máy, chatbot.

**Dự đoán chuỗi thời gian:** Phân tích dữ liệu tài chính, dự báo thời tiết.

**Phát triển Al sáng tạo:** Tạo ảnh, video hoặc âm nhạc.

**d.Thuật toán CNN (Convolutional Neural Network)** : là một trong những phương pháp học sâu mạnh mẽ và quan trọng trong lĩnh vực thị giác máy tính. CNN được sử dụng rộng rãi để giải quyết các bài toán nhận diện hình ảnh, bao gồm phát hiện đối tượng, phân loại hình ảnh, và nhận diện khuôn mặt. Ý tưởng chính của CNN là sử dụng các lớp tích chập (convolutional layers) để tự động học và trích xuất các đặc trưng quan trọng từ dữ liệu hình ảnh hoặc video mà không cần sự can thiệp của con người. Các lớp tích chập giúp hệ thống học các đặc trưng không gian ở nhiều mức độ khác nhau, từ các đặc trưng cơ bản như cạnh, góc đến các đặc trưng phức tạp hơn như hình dáng, cấu trúc và các đối tượng trong hình ảnh.

**Ứng dụng của CNN:**

CNN (Mạng nơ-ron tích chập) có khả năng tự động học và trích xuất các đặc trưng từ dữ liệu không gian như hình ảnh và video. Dưới đây là các ứng dụng nổi bật của CNN trong các lĩnh vực khác nhau:

**Nhận diện và phân loại hình ảnh**

**Mô tả:** CNN được sử dụng để phân loại hình ảnh thành các nhóm cụ thể dựa trên các đặc điểm tự động học được.

**Ví dụ ứng dụng:**

Phân loại loài động vật, cây cối từ ảnh.

Nhận diện chữ viết tay (Handwritten Digit Recognition).

Xác định các loại biển báo giao thông.

**Ứng dụng thực tế:**

Google Photos, Instagram sử dụng CNN để tự động gắn thẻ (tag) ảnh.

**Phát hiện đối tượng (Object Detection)**

**Mô tả**: CNN phát hiện vị trí và nhận dạng các đối tượng trong ảnh hoặc video, thường được dùng trong giám sát hoặc xe tự hành.

**Ví dụ ứng dụng:**

Camera an ninh để phát hiện kẻ xâm nhập.

Phát hiện phương tiện giao thông trên đường.

Kiểm tra sản phẩm lỗi trong sản xuất.

Phương pháp: YOLO, SSD, Faster R-CNN.

**Nhận diện khuôn mặt**

**Mô tả:** CNN giúp phát hiện và nhận diện khuôn mặt trong ảnh hoặc video.

**Ví dụ ứng dụng:**

Mở khóa điện thoại bằng khuôn mặt.

Nhận diện danh tính trong các hệ thống bảo mật.

**Ứng dụng thực tế:**

DeepFace (Facebook), FaceNet (Google), hệ thống nhận diện khuôn mặt của Apple.

**Y tế và chăm sóc sức khỏe**

**Mô tả:** CNN được sử dụng để phân tích hình ảnh y khoa, hỗ trợ bác sĩ trong chẩn đoán.

**Ví dụ ứng dụng:**

Phát hiện ung thư từ ảnh X-quang, MRI, CT.

Phát hiện bệnh võng mạc tiểu đường từ ảnh mắt.

Phân tích tế bào bất thường trong xét nghiệm máu.

**Ứng dụng thực tế:**

Các hệ thống chẩn đoán hỗ trợ AI như Google Health.

**Thị giác máy tính trong xe tự hành**

**Mô tả:** CNN phân tích hình ảnh thu được từ camera trên xe để nhận diện vật cản, biển báo giao thông, vạch kẻ đường, người đi bộ.

**Ứng dụng thực tế:**

Xe tự hành của Tesla và Waymo.

**Phân tích video và giám sát**

**Mô tả:** Phân tích các khung hình trong video để phát hiện sự kiện hoặc hành vi đáng ngờ.

**Ví dụ ứng dụng:**

Phát hiện tai nạn trong video giám sát giao thông.

Phân tích hành vi khách hàng trong cửa hàng.

**Dịch thuật và nhận dạng ký tự**

**Mô tả:** CNN được áp dụng để nhận diện và trích xuất văn bản từ ảnh.

**Ví dụ ứng dụng:**

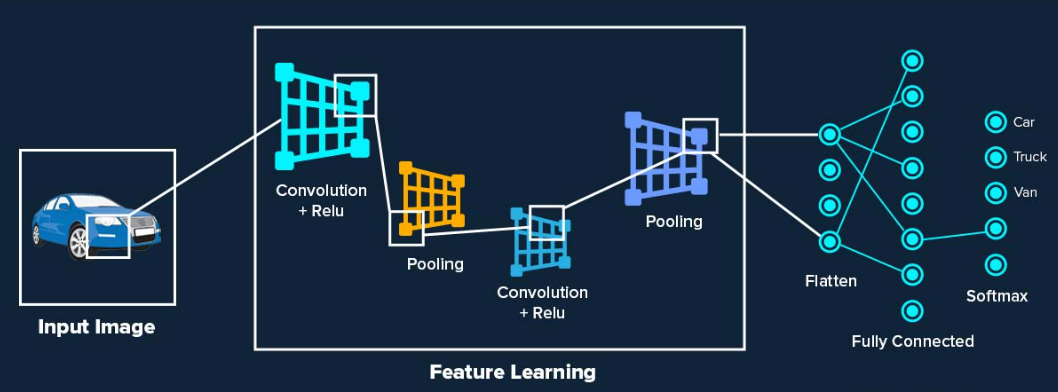
Nhận diện chữ viết tay (OCR - Optical Character Recognition).

Dịch biển báo giao thông qua hình ảnh (Google Translate Camera).

**Sơ đồ thuật toán và hình ảnh mô tả phương pháp**



*Hình 1.5 Thuật toán CNN*



*Hình 1.6 Hình ảnh minh họa thuật toán CNN*

**CHƯƠNG 2 : XÂY DỰNG HỆ THỐNG NHẬN DẠNG BÀI TOÁN**

* 1. **. Yêu cầu của bài toán nhận dạng biển báo giao thông:**

### 2.1.1. Phân tích bài toán

Bài toán nhận dạng biển báo giao thông là một bài toán trong lĩnh vực xử lý ảnh và thị giác máy tính, nhằm mục đích nhận diện và trích xuất biển số xe từ hình ảnh. Hệ thống sẽ nhận diện biển báo trong ảnh và chuyển đổi nó thành dạng văn bản để xử lý hoặc lưu trữ.

Mục tiêu chính của bài toán:

Phát hiện biển báo: Xác định và khoanh vùng biển báo trong ảnh.

Nhận diện các ký tự trên biển báo giao thông: Chuyển các ký tự trên biển số thành chuỗi văn bản có thể đọc được.

Lưu trữ và tra cứu thông tin: Lưu thông tin biển số vào cơ sở dữ liệu hoặc hiển thị kết quả trên giao diện người dùng.

Các bước chính trong hệ thống nhận dạng biển báo giao thông:

**Nhập ảnh đầu vào:** Ảnh chứa biển báo giao thông nhập từ ảnh tải lên.

**Phát hiện biển báo giao thông:** Xác định vị trí và vùng chứa biển báo trong ảnh.

**Nhận diện ký tự biển báo giao thông:** Trích xuất và nhận diện các ký tự từ biển báo.

**Xuất kết quả:** Hiển thị kết quả dưới dạng chuỗi ký tự biển số hoặc lưu vào cơ sở dữ liệu.

**Các yêu cầu cụ thể của bài toán**

Phát hiện biển báo giao thông trong ảnh

Mục tiêu chính ở đây là phát hiện và xác định vị trí của biển báo giao thông trong ảnh, bất kể các yếu tố như góc nghiêng, độ sáng, hay các yếu tố ngoại cảnh như bị che khuất một phần.

**Vấn đề cần giải quyết:**

Phát hiện biển báo giao thông chính xác trong các điều kiện ánh sáng khác nhau.

Xử lý các biển số bị méo hoặc góc nghiêng.

Biển số bị che khuất bởi các yếu tố ngoại cảnh.

**Nhận diện ký tự biển báo giao thông :**

Sau khi phát hiện được biển báo giao thông trong ảnh, hệ thống sẽ trích xuất các ký tự trên biển số và chuyển đổi chúng thành dạng văn bản.

**Vấn đề cần giải quyết:**

Nhận diện chính xác các ký tự chữ và số trong biển số.

Giải quyết vấn đề biển báo giao thông có chất lượng kém (nhòe, mờ, nhiễu).

Đảm bảo độ chính xác cao với các kiểu chữ và màu sắc biển báo giao thông khác nhau.

**Tốc độ xử lý**

Hệ thống cần xử lý ảnh nhanh chóng để có thể hoạt động trong môi trường thực tế

Yêu cầu về thời gian xử lý:

Thời gian xử lý một ảnh không quá 2 giây.

Khả năng nhận diện biển báo giao thông từ video hoặc luồng hình ảnh liên tục

**Độ chính xác cao**

Độ chính xác của hệ thống nhận diện phải đạt tỷ lệ cao, đặc biệt trong các điều kiện ánh sáng và chất lượng ảnh thay đổi.

**Yêu cầu về độ chính xác:**

Hệ thống cần đạt ít nhất 95% độ chính xác trong việc nhận diện biển báo giao thông với ảnh có chất lượng tốt.

Đảm bảo hệ thống vẫn hoạt động hiệu quả với ảnh bị nhiễu hoặc góc nghiêng.

**Dễ sử dụng và tích hợp**

Hệ thống cần có giao diện dễ sử dụng và có thể tích hợp với các hệ thống giám sát khác hoặc các hệ thống giám sát giao thông.

Yêu cầu về giao diện:

Giao diện người dùng (UI) dễ sử dụng và thân thiện.

**Input và Output**

**Input (Đầu vào):**

Ảnh biển báo giao thông: Hình ảnh chứa biển báo giao thông

Định dạng: .jpg, .png, .bmp.

Kích thước ảnh: tối thiểu 800x600 pixels, có thể lớn hơn nếu yêu cầu độ chi tiết cao.

Chất lượng: Hệ thống cần có khả năng xử lý các ảnh có độ phân giải thấp, bị mờ, hoặc nhiễu.

## **2.2. Sơ đồ khối tổng quát**

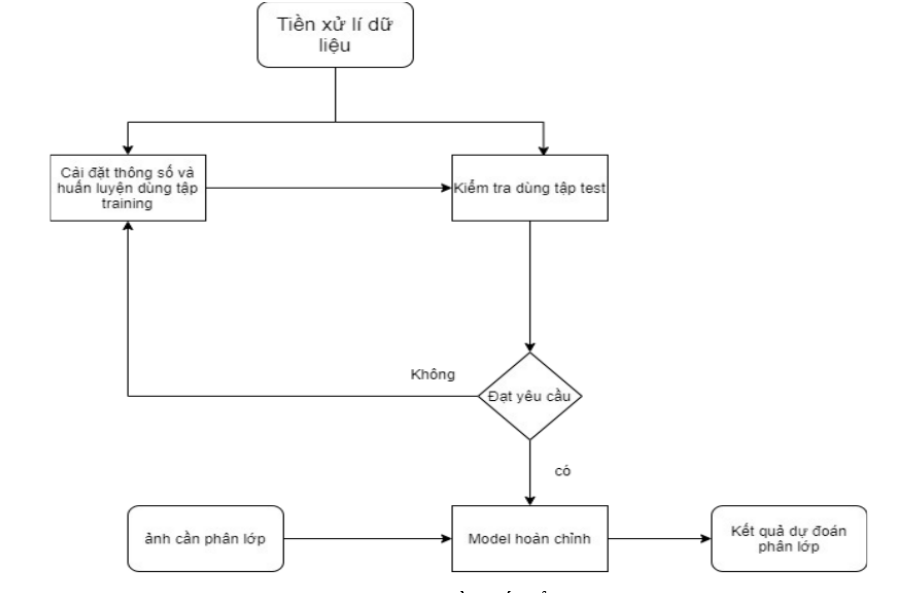
Bước 1: Tìm kiếm và thu thập datasets về các loại biển báo giao thông, tiền xử lý và

gán nhãn cho từng bức ảnh.

Bước 2: Phân chia dữ liệu thành tập train và tập test.

Bước 3: Huấn luyện mô hình.

Bước 4: Đánh giá mô hình.



*Hình 2.1 : Sơ đồ khối hệ thống*

**CHƯƠNG 3 : THỰC NGHIỆM**

**3.1. Dữ liệu**

**Nguồn gốc dữ liệu :** Chúng ta sẽ sử dụng tập dữ liệu công khai có sẵn tại kaggle. Tải xuống tập dữ liệu Traffic Signs Dataset.

Tập dữ liệu chứa hơn 50.000 hình ảnh về các biển báo giao thông khác nhau, nó được phân loại thành 43 lớp. Dữ liệu phân chia khá là không đồng đều, 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.

**Tập dữ liệu thường được chia làm 3 tập:**

**Tập training:** tập training thường có kích thước lớn nhất. Tập training thường được

dán nhãn và cho trước nhãn để tính toán loss function và cập nhập các tham số để mô hình

tăng độ chính xác.

**Tập validation (hay tập development):** Tập validation thường được chia ra từ tập

training. Tập validation là không được dùng để cập nhập các tham số của mô hình, mà nó

chỉ được dùng để đánh giá xem mô hình có cần điều chỉnh parameter lại hay không, có bị

high bias hay high variance hay không.

**Tập test:** tập test được dùng để đánh giá lại mô hình có hiệu quả hay không. Dữ liệu của tập test phải chưa từng xuất hiện trong tập training để có thể đánh giá mô hình ngoài thực tế.

Các tập training, test thường được chia với tỉ lệ 80/20/20 với tập dữ liệu không quá lớn với tập dữ liệu rất lớn (khoảng 1 triệu mẫu). Trong việc phân loại biển báo giao thông trong ảnh, tập datasets được sử dụng là các biển báo giao thông của Đức được công khai nguồn dữ liệu trên trang web Kaggle. Tại tập datasets này, chúng tôi sử dụng file Train.zip bao gồm 9030 mẫu biển báo của 43 class để tiến hành training và file Test.zip để tiến hành kiểm tra lại



*Hình 3.1: Tập dữ liệu được sử dụng để xây dựng mô hình CNN*

Trong đó, tập dữ liệu training mẫu biển báo giao thông của Đức với 43 class được gán nhãn như sau:

**BẢNG 3.1: NHÃN VÀ TÊN CÁC BIỂN BÁO TRONG BÀI TOÁN PHÂN LOẠI**

|  |  |  |  |
| --- | --- | --- | --- |
| Nhãn | Tên biển báo | Nhãn | Tên biển báo |
| 0 | Tốc độ giới hạn 20km/h | 22 | Cảnh báo đường nhấp nhô |
| 1 | Tốc độ giới hạn 30km/h | 23 | Cánh báo đường trơn trượt |
| 2 | Tốc độ giới hjan 50km/h | 24 | Đường bị hẹp bên phải |
| 3 | Tốc độ giới hạn 60km/h | 25 | Đường đang thi công |
| 4 | Tốc độ giới hạn 70km/h | 26 | Cảnh báo đèn có tín hiệu |
| 5 | Tốc độ giới hạn 80km/h | 27 | Cảnh báo có người đi bộ |
| 6 | Hết đoạn đường giới hạn 80km/h | 28 | Cảnh báo có học sinh đi qua |
| 7 | Tốc độ giới hạn 100km/h | 29 | Cảnh báo có xe đạp đi qua |
| 8 | Tốc độ giới hạn 120km/h | 30 | Cảnh báo tuyết lở |
| 9 | Cấm vượt | 31 | Cảnh báo thú hoang đi qua |
| 10 | Cấm xe trên 3,5 tấn vượt | 32 | Hết tất cả lệnh cấm |
| 11 | Giao nhau với dường không ưu tiên | 33 | Rẽ phải phía trước |
| 12 | Giao nhau với đường ưu tiên | 34 | Rẽ trái phái trước |
| 13 | Biển cảnh báo nguy hiểm | 35 | Đi thẳng |
| 14 | Biển dừng lại | 36 | Đi thẳng hoặc rẽ phải |
| 15 | Biển đường cấm | 37 | Đi thẳng hoặc rẽ trái |
| 16 | Cấm xe trên 3,5 tấn | 38 | Hướng phải đi vòng sang phải |
| 17 | Cấm đi ngược chiều | 39 | Hướng trái đi vòng sang trái |
| 18 | Biển nguy hiểm khác | 40 | Nơi giao nhau chạy theo vòng xuyến |
| 19 | Nguy hiểm khi rẽ trái | 41 | Hết đoạn đường cấm vượt |
| 20 | Nguy hiểm khi rẽ trái | 42 | Hết đoạn đường cấm vượt với xe trên 3,5 tấn |
| 21 | Chỗ ngoặt nguy hiểm |  |  |

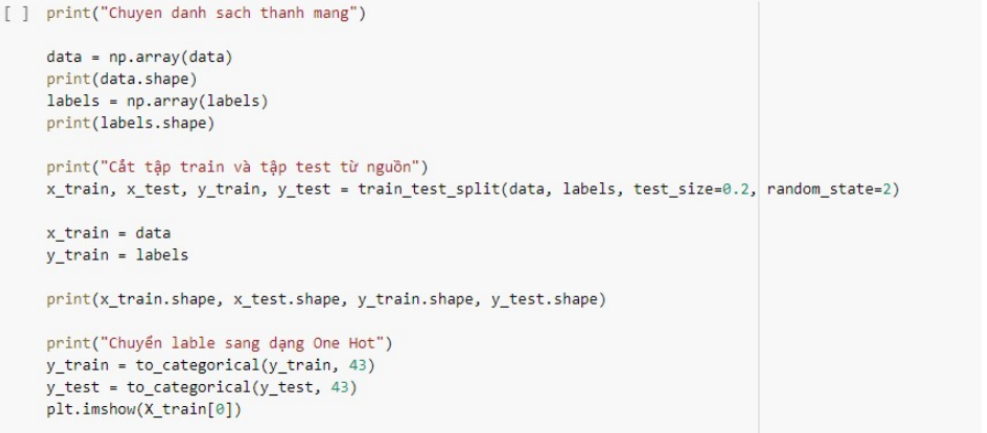
**3.1.1. Tiền xử lý và phân chia dữ liệu**

Đầu tiên, chúng ta phải truy suất đến tập tin lưu dữ liệu. Sau đó, chúng ta sẽ lưu lại dữ liệu, nhãn của từng bức ảnh và reshape lại tập datasets cho phù hợp với đầu vào của mạng nơtron tích chặp. Quá trình trên được viết như sau:



*Hình 3.2 Tiền xử lý tập dữ liệu*

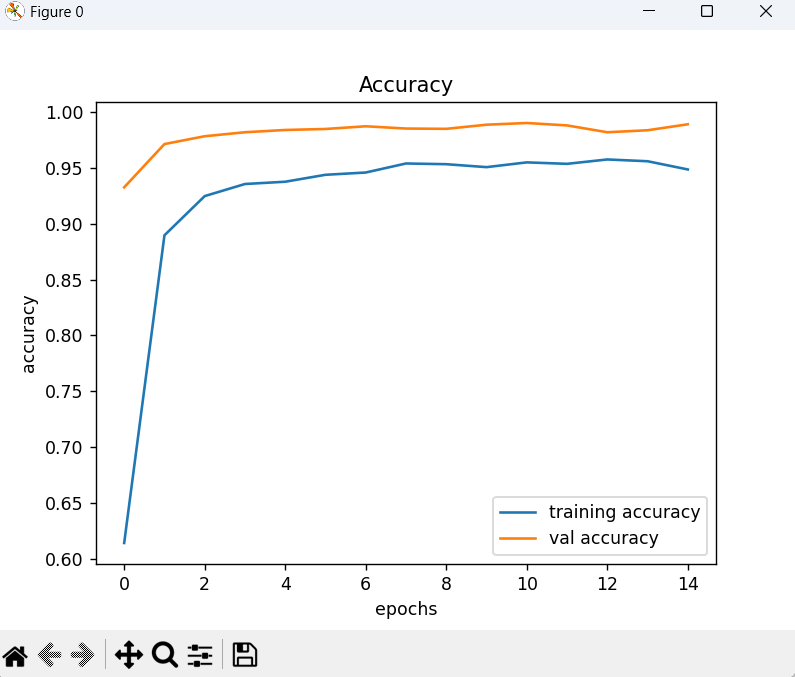
Tiếp theo, để dễ dàng cho việc tính toán của mạng nơtron tích chặp, chúng ta phải chuyển dữ liệu của mẫu biển báo này từ dạng danh sách sang mảng, sau đó tiến hành cắt tập train và tập test với tỷ lệ 8:2 và chuyển nhãn sang dạng one-hot cho phù hợp với mạng nơtron tích chặp. Quá trình trên được viết như sau:



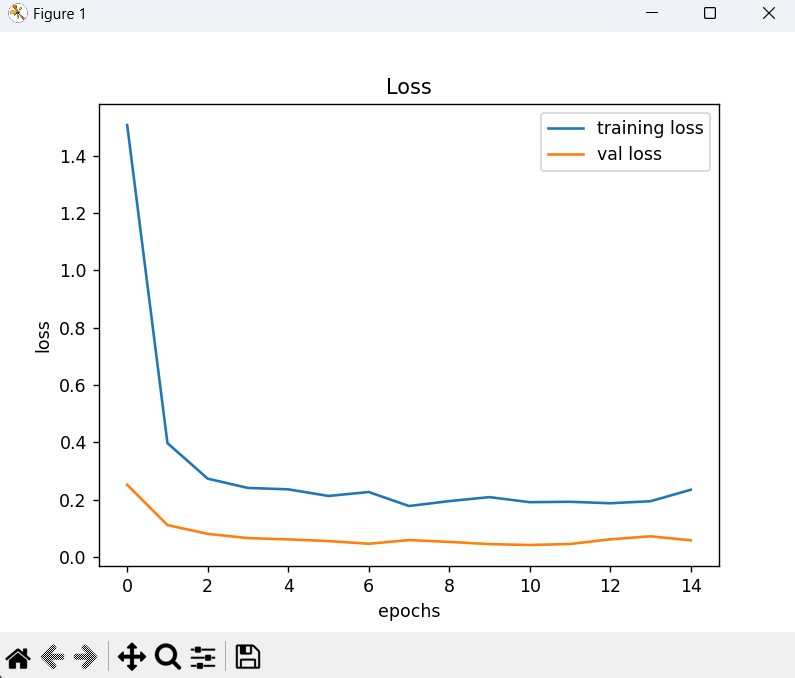
*Hình 3.3 Xử lý và phân chia tập train,test*

**3.2. Các độ so sánh**

Để đánh giá hiệu quả của mô hình, nghiên cứu sử dụng các độ đo sau:



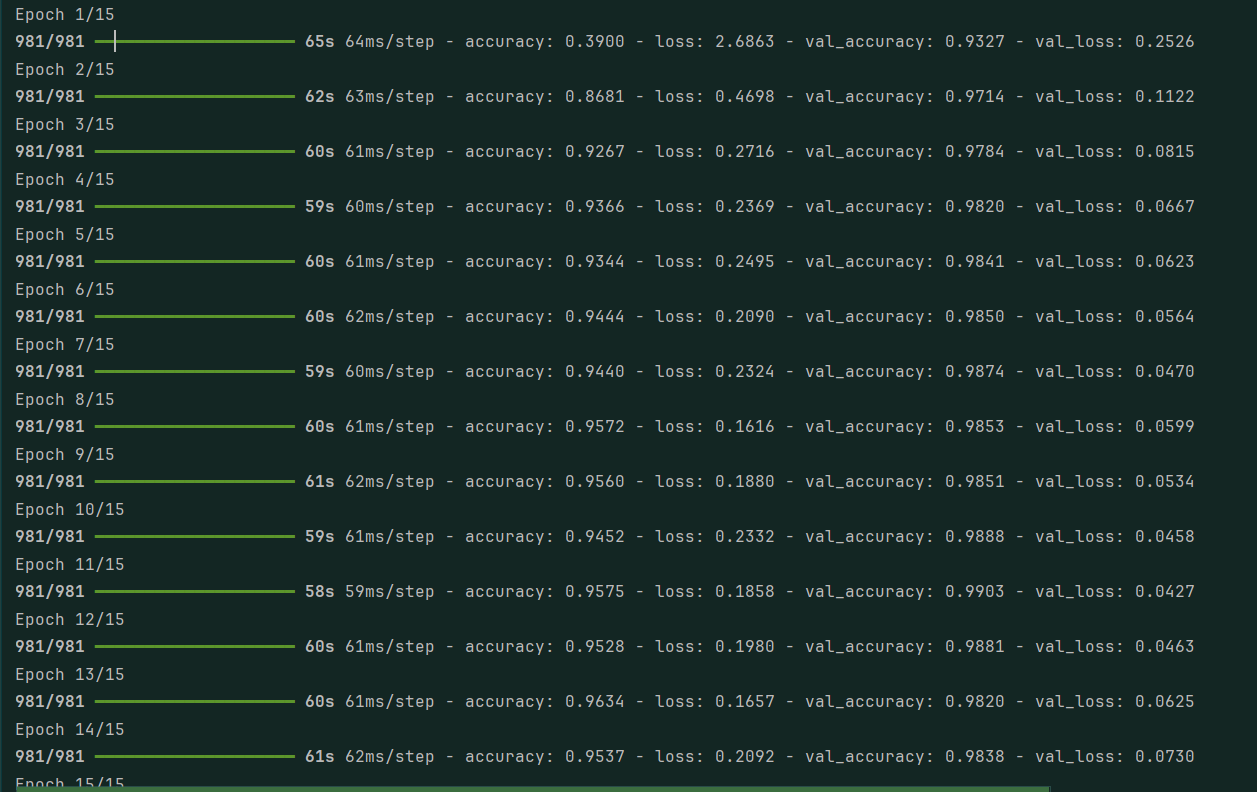
*Hình 3.4 : Gia trị chuẩn xác của mô hình huấn luyện*

****

*Hình 3.5 : Gia trị chưa chuẩn xác của mô hình huấn luyện*

Epochs : 15

Batch size : 32



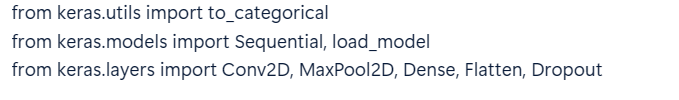
****

**3.3. Kết quả thực nghiệm**

Với đề tài này, nhóm chúng tôi huấn luyện mô hình trên thư viện Keras. Đây là một mã nguồn mở dành cho Neural Network. Keras là một framework cấp cao, cung cấp một số công cụ trừu tượng, 4 module được dược sử dụng chủ yếu trong Keras bao gồm : Keras models, Keras layers, Keras losses, Keras optimizers. Có 5 bước cơ bản để huấn luyện và kiểm tra mô hình dùng các hàm đã xây dựng sẵn của Keras.

**Bước 1: Khai báo các module cần thiết của thư viện Keras**

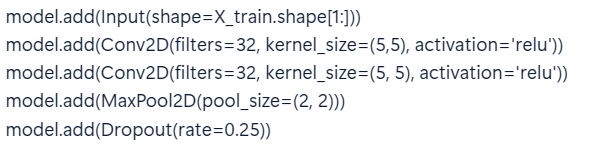
**VD**: Khai báo các thư viện cần thiết để thực hiện huấn luyện 3 kiến trúc bên trên



*Hình 3.6 : Khai báo thư viện Keras*

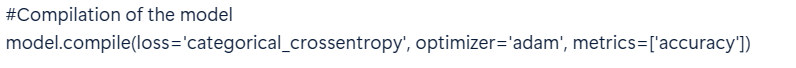
Bước 2: Tạo model bằng cách sử dụng các hàm trong thư viện Keras:

VD: Tạo một Convolutional Layer 2D, Maxpooling và Dropout

****

*Hình 3.7 : Tạo 1 model Keras*

Bước 3: Compile model bằng lệnh sau:



*Hình 3.8 : Compile model Keras*

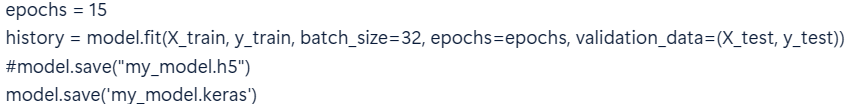
Trong đó:

Hàm optimizer có thể là ‘adam’, ‘sdg’,RMSprop, …

Hàm Loss dùng để tính toán chất lượng của model từ đó tìm cách giảm thiểu trong

quá trình đào tạo.

Bước 4: Train model và test model bằng lệnh sau:



*Hình 3.9 : Training model Keras*

Bước 5: Lưu model để sử dụng sau này

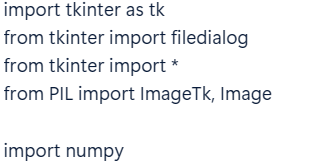


*Hình 3.10 : Lưu model Keras*

**3.3.1. Tạo giao diện người dùng**

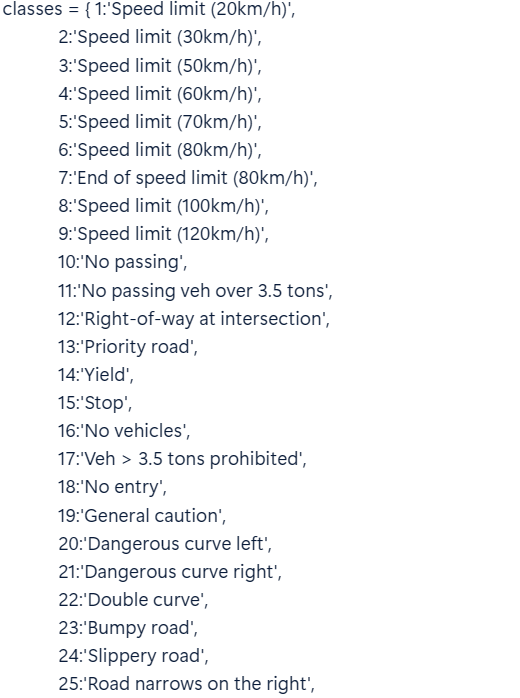
Sau quá trình training, chúng ta sẽ nhận được file lưu trọng số của model, lúc này chúng ta sẽ tiến hành xây dựng giao diện người dùng sử dụng thư viện Tkinter của Python và load model để nhận diện biển báo.

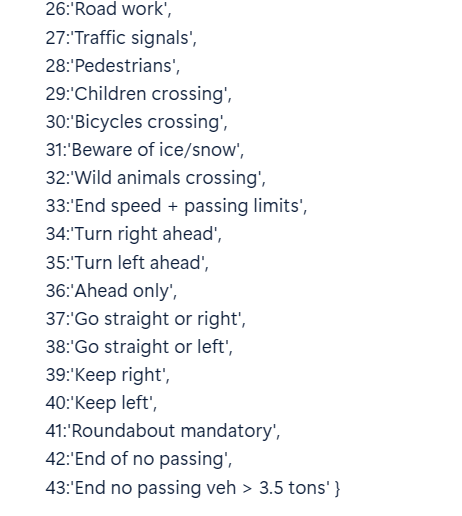
**Bước 1: Khai báo thư viện của Tkinter và Keras**

****

*Hình 3.11 : Khai báo thư viện Tkinter và Keras để tạo giao diện người dùng*

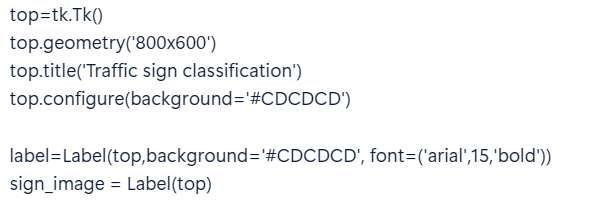
**Bước 2: Tạo mảng lưu lại nhãn của 43 lớp mẫu biển báo**

****

****

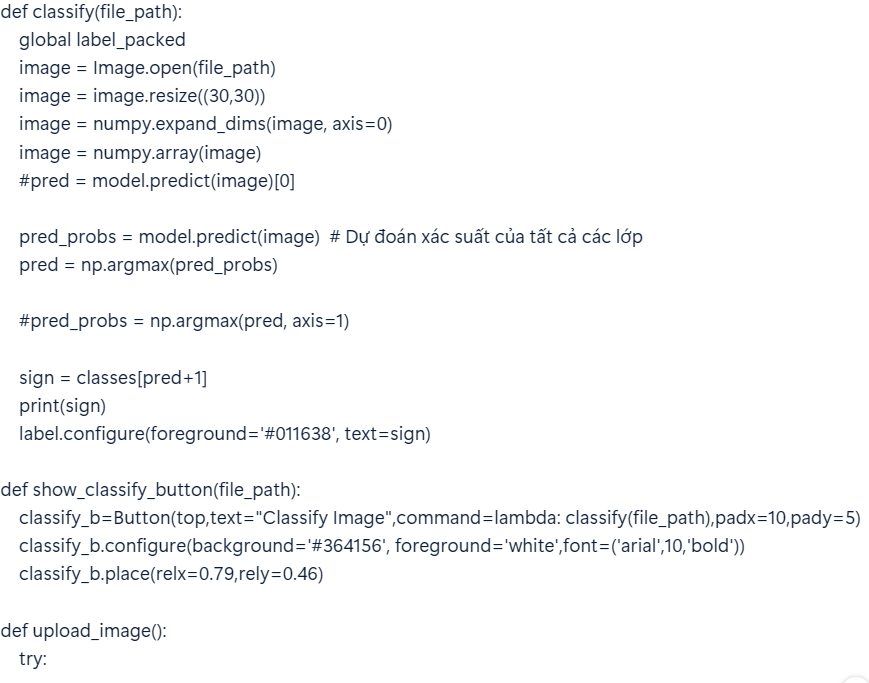
*Hình 3.12: Khai báo thư viện Tkinter và Keras để tạo giao diện người dùng*

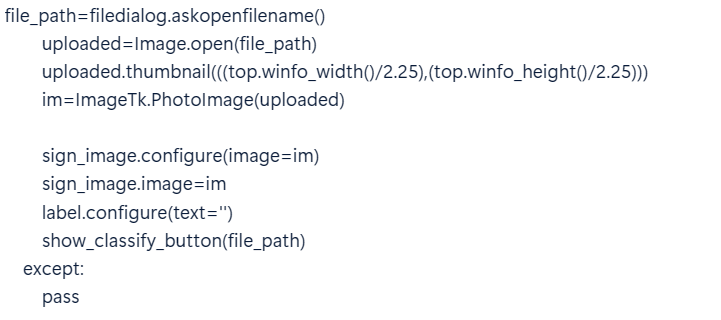
**Bước 3: Thiết kế giao diện**

****

*Hình 3.13: Tạo giao diện*

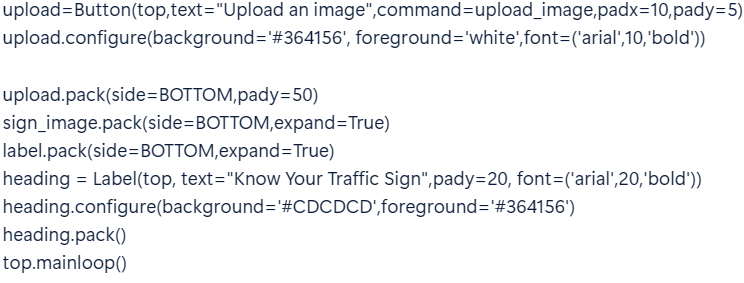
**Bước 4: Tạo các hàm để xử lý và đưa ra kết quả**

****

****

*Hình 3.14: Tạo hàm xử lý ảnh*

**Bước 5: Gọi hàm vừa khai báo và tiến hành classify ảnh**

****

*Hình 3.15: Tiến hành xử lý ảnh*

Kết quả thu được sau khi tạo mô hình và thiết kế giao diện:

****

*Hình 3.16 : Kết quả*

**CHƯƠNG 4 : KẾT LUẬN**

**Kết quả đạt được**

Hệ thống nhận diện biển báo giao thông bằng CNN đã đạt được những kết quả nổi bật. Cụ thể, hệ thống hoạt động với độ chính xác cao, có khả năng nhận diện chính xác các loại biển báo giao thông trong nhiều điều kiện môi trường khác nhau như ánh sáng, góc nhìn và thời tiết. Với tốc độ xử lý nhanh, hệ thống đảm bảo khả năng hoạt động gần thời gian thực, phù hợp cho các ứng dụng thực tế. Nhờ sử dụng các công cụ như Keras và TensorFlow, hệ thống có cấu trúc gọn nhẹ, dễ dàng huấn luyện và triển khai trên nhiều nền tảng phần cứng khác nhau. Dữ liệu đầu vào và kết quả nhận diện được lưu trữ hiệu quả, hỗ trợ tốt cho việc phân tích và phát triển thêm các tính năng liên quan đến hệ thống giao thông thông minh.

**Hướng phát triển**

Hệ thống cần được nâng cấp để nhận diện đa dạng hơn các loại biển báo giao thông từ nhiều quốc gia với độ chính xác cao hơn. Tích hợp các mô hình học sâu tiên tiến như EfficientNet hoặc MobileNetV2 để tối ưu hóa hiệu suất và giảm tài nguyên xử lý. Việc kết nối với các thiết bị IoT sẽ giúp gửi cảnh báo và thông tin biển báo theo thời gian thực qua các kênh như ứng dụng di động hoặc Email. Bổ sung tính năng nhận diện biển báo bị che khuất hoặc hư hỏng bằng cách kết hợp các kỹ thuật xử lý ảnh nâng cao.

Ngoài ra, hệ thống nên được tối ưu hóa để hoạt động tốt trên các thiết bị có phần cứng hạn chế như Raspberry Pi, phù hợp cho các ứng dụng gắn trên xe tự hành. Giao diện quản lý và hiển thị kết quả cũng cần được xây dựng trực quan, thân thiện để người dùng dễ dàng quản lý và giám sát. Cuối cùng, hệ thống có thể mở rộng ứng dụng vào các lĩnh vực như đào tạo lái xe, phân tích giao thông, hoặc hỗ trợ xây dựng hệ thống giao thông thông minh, góp phần nâng cao an toàn và hiệu quả trong vận hành.

**TÀI LIỆU THAM KHẢO**

"Deep Learning" - Ian Goodfellow, Yoshua Bengio, Aaron Courville

Một cuốn sách nền tảng về học sâu, cung cấp kiến thức lý thuyết và ứng dụng liên quan đến các mạng nơ-ron tích chập (CNN).

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" - Aurélien Géron

Sách hướng dẫn chi tiết cách sử dụng Keras và TensorFlow để xây dựng các mô hình học sâu, bao gồm nhận diện hình ảnh.

"Convolutional Neural Networks for Visual Recognition" - Stanford University CS231n

Một khóa học phổ biến với tài liệu giảng dạy về CNN, có nhiều ví dụ về nhận diện đối tượng và hình ảnh.

Bộ dữ liệu [GTSRB - German Traffic Sign Recognition Benchmark | Kaggle](https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign)