BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG


**NIÊN LUẬN CƠ SỞ NGÀNH**
**NGÀNH KHOA HỌC MÁY TÍNH**

**Đề tài**

# NHẬN DIỆN BIỂN BÁO GIAO THÔNG BẰNG MÁY HỌC VÉC-TƠ HỖ TRỢ VÀ MẠNG NƠ-RON TÍCH CHẬP

**Sinh viên thực hiện:**
**Lê Tuấn Đạt**
**Mã số: B2113328**
**Khóa: 47**

Cần Thơ, 05/2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG


**NIÊN LUẬN CƠ SỞ NGÀNH**
**NGÀNH KHOA HỌC MÁY TÍNH**

**Đề tài**

# NHẬN DIỆN BIỂN BÁO GIAO THÔNG BẰNG MÁY HỌC VÉC-TƠ HỖ TRỢ VÀ MẠNG NƠ-RON TÍCH CHẬP

**Giảng viên hướng dẫn:**
**TS. Lưu Tiến Đạo**

**Sinh viên thực hiện:**
**Lê Tuấn Đạt**
**Mã số: B2113328**
**Khóa: 47**

Cần Thơ, 05/2024

# NHẬN XÉT CỦA GIẢNG VIÊN

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# LỜI CẢM ƠN

Để có được bài niên luận này, em xin được bày tỏ lòng biết ơn chân thành và sâu sắc đến Thầy Lưu Tiến Đạo – người đã trực tiếp tận tình hướng dẫn, giúp đỡ em. Trong suốt quá trình thực hiện niên luận, nhờ những sự chỉ bảo và hướng dẫn quý giá đó mà bài niên luận này được hoàn thành một cách tốt nhất.

Em cũng xin gửi lời cám ơn chân thành đến các Thầy Cô Giảng viên Đại học Cần Thơ, đặc biệt là các Thầy Cô ở Trường CNTT & TT, những người đã truyền đạt những kiến thức quý báu trong thời gian qua.

Em cũng xin chân thành cảm ơn bạn bè cùng với gia đình đã luôn động viên, khích lệ và tạo điều kiện giúp đỡ trong suốt quá trình thực hiện để em có thể hoàn thành bài niên luận một cách tốt nhất.

Tuy có nhiều cố gắng trong quá trình thực hiện niên luận, nhưng không thể tránh khỏi những sai sót. Em rất mong nhận được sự đóng góp ý kiến quý báu của quý Thầy Cô và các bạn để bài niên luận hoàn thiện hơn.

*Cần Thơ, ngày 13 tháng 05 năm 2024*

***Người viết***

*(Ký và ghi họ tên)*

# MỤC LỤC

# DANH MỤC HÌNH

# DANH MỤC BẢNG

# DANH MỤC CÁC TỪ VIẾT TẮT

| STT | Từ viết tắt | Diễn giải |
|---|---|---|
| 1 | CNN | Convolutional Neural Networks |
| 2 | GTSRB | German Traffic Sign Regconition Benchmark |
| 3 | SVM | Support Vector Machines |
| 4 | ReLU | Rectified Linear Unit |

# TÓM TẮT

Trong thời đại ngày nay, việc nhận diện biển báo giao thông đóng vai trò quan trọng trong việc tham gia giao thông của tất cả mọi người. Với sự phát triển của trí tuệ nhân tạo ngày nay, các hãng xe lớn dần áp dụng các ứng dụng của trí tuệ nhân tạo vào các sản phẩm của họ. Các hãng xe lớn như Tesla, BMW, và Mercedes-Benz đã và đang đầu tư mạnh mẽ vào việc nghiên cứu và phát triển các hệ thống nhận diện biển báo giao thông và xe tự lái.

"Nhận diện biển báo giao thông bằng máy học véc-tơ hỗ trợ và mạng nơ-ron tích chập" là một ứng dụng web cho phép nhận diện nhanh chóng và chính xác các loại biển báo giao thông. Trang web sẽ cung cấp thông tin các loại biển báo sau khi đã qua xử lý bằng các mô hình học máy phân loại. Trang web được viết bằng thư viện Flask dựa trên ngôn ngữ lập trình Python.

Tóm lại, ứng dụng "nhận diện biển báo giao thông bằng máy học véc-tơ hỗ trợ và mạng nơ-ron tích chập" có thể được xem là một công cụ hữu ích và thông minh để giúp giảm thiểu tình trạng tai nạn giao thông khi người tham gia đã biết được về các loại biển báo.

# ABSTRACT

Nowadays, recognizing traffic signs plays an important role in everyone's traffic participation. With the development of artificial intelligence today, major car manufacturers are gradually applying AI applications to their products. Major car manufacturers like Tesla, BMW, and Mercedes-Benz have been and are heavily investing in the research and development of traffic sign recognition systems and self-driving cars.

“Traffic sign recognition using support vector machine and convolutional neural networks” is a web application that allows quick and accurate recognition of various types of traffic signs. The website will provide information about types of signs after being processed by classification models. The website is written by using the Flask library based on the Python programming language.

In summary, the application "traffic sign recognition using support vector machine and convolutional neural networks" can be considered a useful and smart tool to help reduce traffic accidents when participants already know about the types of signs.

# PHẦN I: GIỚI THIỆU

## 1. Đặt vấn đề

Trong thời đại của sự tiên tiến vượt bậc trong lĩnh vực trí tuệ nhân tạo (AI), việc tích hợp công nghệ này vào ngành công nghiệp ô tô không chỉ là một xu hướng mà còn là một yếu tố quyết định trong việc tạo ra những dòng sản phẩm có khả năng tự lái và an toàn hơn. Các hãng xe lớn như Tesla, BMW, Audi, và Mercedes-Benz đều đang đẩy mạnh nghiên cứu và phát triển các công nghệ AI để áp dụng vào các dòng xe của họ. Các hệ thống trí tuệ nhân tạo không chỉ giúp cải thiện trải nghiệm lái xe mà còn tăng cường tính an toàn cho cả người lái và người đi đường.

Một trong những ứng dụng tiêu biểu của trí tuệ nhân tạo trong ô tô là hệ thống nhận diện biển báo giao thông. Công nghệ này cho phép phương tiện tự động nhận biết các biển báo trên đường và thích ứng với chúng một cách linh hoạt và chính xác. Tính năng này không chỉ giúp lái xe dễ dàng nhận diện thông tin về tốc độ giới hạn và hướng đi, mà còn giúp nâng cao tính an toàn bằng cách cảnh báo về các biển báo cấm hay biển báo nguy hiểm.

Trong bối cảnh này, việc nghiên cứu và phát triển các phương pháp hiệu quả cho việc nhận diện biển báo giao thông là cực kỳ quan trọng. Một trong những phương pháp phổ biến và hiệu quả đã được áp dụng là kết hợp giữa máy học vector hỗ trợ (SVM) và mạng nơ-ron Tích chập (CNN). SVM được sử dụng để phân loại các đối tượng trong không gian đa chiều một cách hiệu quả, trong khi CNN nổi bật trong việc xử lý và trích xuất đặc trưng từ hình ảnh. Kết hợp giữa hai phương pháp này giúp tăng cường khả năng nhận diện và phân loại biển báo giao thông, từ đó cải thiện hiệu suất của hệ thống nhận diện trên các xe tự lái.

Nghiên cứu này nhằm mục đích đóng góp vào việc phát triển công nghệ lái xe tự động và nâng cao độ an toàn trên các con đường hiện đại. Bằng cách áp dụng SVM và CNN vào việc nhận diện biển báo giao thông, hy vọng rằng chúng ta có thể xây dựng những hệ thống thông minh hơn, giúp giảm thiểu rủi ro tai nạn giao thông và tạo ra một môi trường lái xe an toàn và tiện lợi hơn cho mọi người.

## 2. Lịch sử giải quyết vấn đề

Lịch sử của việc giải quyết vấn đề "nhận diện biển báo giao thông" là một hành trình từ những phương pháp truyền thống đến các phương pháp trí tuệ nhân tạo và học sâu. Cụ thể:

- Phương pháp truyền thống: Trước khi trí tuệ nhân tạo (AI) trở nên phổ biến, việc nhận diện biển báo giao thông thường dựa vào phân loại màu sắc, kích thước, hình dạng và các đặc trưng cụ thể khác của biển báo. Các phương pháp này thường dựa vào các giải thuật xử lý ảnh và phân loại tín hiệu để nhận diện.

- Sử dụng các mô hình máy học: Đối với các hệ thống nhận diện biển báo giao thông tiên tiến hơn, các kỹ thuật học máy đã được áp dụng như máy học véc-tơ hỗ trợ, rừng ngẫu nhiên và gọm cụm, gom nhóm để tối ưu hóa quá trình nhận diện.

- Sự phát triển của trí tuệ nhân tạo và học sâu: Sự phát triển của học sâu đã mang đến làn gió mới cho vấn đề này. Các mô hình học sâu như mạng nơ-ron tích chập, mạng nơ-ron hồi quy đã trở thành tiêu chuẩn trong việc giải quyết vấn đề nhận diện biển báo giao thông.

## 3. Mục tiêu đề tài

Mục tiêu của đề tài là huấn luyện được các mô hình CNN và SVM để nhận diện được các loại biển báo giao thông với độ chính xác cao, kể cả điều kiện thời tiết và ánh sáng có biến đổi.

## 4. Đối tượng và phạm vi nghiên cứu

- ✓ Đối tượng nghiên cứu tập trung vào 43 loại biển báo giao thông đường bộ được lấy từ tập dữ liệu chứa các biển báo giao thông tại Đức (GTSRB).
- ✓ Phạm vi nghiên cứu bao gồm thu thập, xử lý tập dữ liệu, huấn luyện mô hình và tích hợp các mô hình vào giao diện với Flask.

## 5. Phương pháp nghiên cứu

Tập dữ liệu nghiên cứu được lấy từ tập dữ liệu các biển báo giao thông tại Đức (GTSRB) bao gồm 43 lớp đối tượng, với 39.209 hình ảnh dành cho huấn luyện và 12.630 hình ảnh dành cho thực nghiệm.

Sử dụng tập dữ liệu vừa thu thập để huấn luyện các mô hình CNN và SVM.

Sử dụng tập dữ liệu kiểm tra để đánh giá hiệu suất các mô hình.

Xây dựng giao diện bằng Flask để hỗ trợ nhận diện các biển báo giao thông.

## 6. Kết quả đạt được

- Xây dựng được tập dữ liệu biển báo giao thông với 43 lớp đối tượng, bao gồm 51.839 hình ảnh.

- Huấn luyện mô hình CNN và SVM nhận dạng các biển báo giao thông dựa vào tập dữ liệu đã thu thập.
- Xây dựng được giao diện nhận dạng biển báo giao thông và tích hợp các mô hình vào giao diện.

**7. Bố cục niên luận:**

**Phần I: Giới thiệu**

Giới thiệu tổng quát về đề tài.

**Phần II: Nội dung**

**Chương 1:** Mô tả bài toán

**Chương 2**: Thiết kế và cài đặt

**Chương 3:** Kiểm thử và đánh giá

**Phần III: Kết luận**

Trình bày kết quả đạt được và hướng phát triển hệ thống.

# PHẦN II: NỘI DUNG

## CHƯƠNG 1: MÔ TẢ BÀI TOÁN

### 1. Mô tả chi tiết bài toán

Bài toán "Nhận diện biển báo giao thông bằng máy học véc-tơ hỗ trợ và mạng nơ-ron tích chập" nhằm xây dựng một hệ thống có khả năng nhận diện các biển báo giao thông từ hình ảnh.

Để giải quyết bài toán, đầu tiên cần xây dựng các mô hình nhận dạng biển báo giao thông bằng mạng nơ-ron tích chập (CNN) và máy học véc-tơ hỗ trợ (SVM). Hai mô hình này sẽ được huấn luyện dựa trên tập dữ liệu bao gồm 43 loại biển báo giao thông. Mục tiêu là nhận dạng được các loại biển báo.

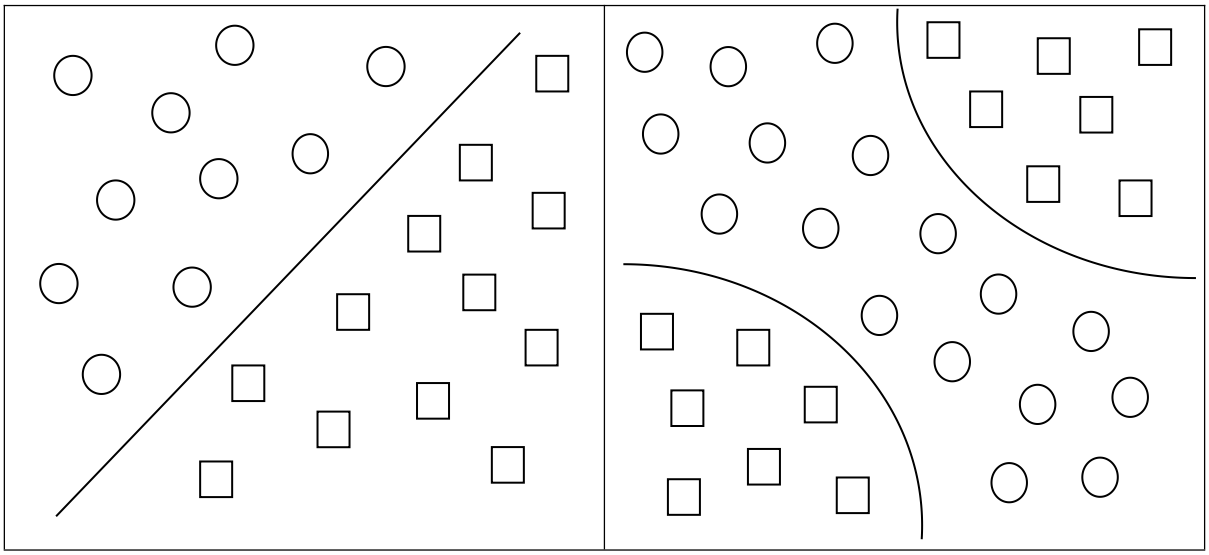
Sau khi đã huấn luyện xong hai mô hình, giao diện sẽ được xây dựng trên nền tảng Streamlit để nhận diện các hình ảnh đầu vào từ các mô hình đã được tích hợp từ trước. Kết quả cuối cùng sẽ là thông tin của loại biển báo cần nhận diện.

### 2. Vấn đề và giải pháp liên quan đến bài toán

#### 2.1. Mô hình "Máy học véc-tơ hỗ trợ"

Máy học véc-tơ hỗ trợ (SVM) được Vapnik nghiên cứu từ những năm 1965, mãi đến những năm 1990 thì giải thuật mới chính thức được phát triển mạnh, trở thành công cụ hữu hiệu và phổ biến của lĩnh vực máy học, nhận dạng và khai mỏ dữ liệu. SVM được áp dụng thành công trong rất nhiều ứng dụng như nhận dạng mặt người, phân loại văn bản, phân loại bệnh ung thư, v.v… Bằng việc kết hợp với phương pháp hàm nhân, máy học véc-tơ hỗ trợ hỗ trợ cung cấp các mô hình hiệu quả chính xác cho các vấn đề phân lớp và hồi quy tuyến tính và phi tuyến (xem hình 1) trong thực tế. Giải thuật máy học SVM nhận đầu vào là một hàm nhân sẽ tạo ra một mô hình mới mà không cần đến bất kỳ thay đổi nào từ chương trình. Giải thuật học dẫn đến giải bài toán quy hoạch toàn phương, luôn có kết quả tối ưu toàn cục.

Là thuật toán học có giám sát được sử dụng cho cả phân loại và hồi quy, nhưng nó phù hợp nhất để phân loại. Mục đích chính của thuật toán là tìm một siêu phẳng tối ưu trong không gian n chiều có thể phân tách các điểm dữ liệu giống nhau về một phía trong các lớp khác nhau trong không gian đặc trưng. Siêu phẳng cố gắng sao cho khoảng cách giữa các điểm gần nhất của các lớp khác nhác nhau phải lớn nhất có thể. Kích thước của siêu phẳng phụ thuộc vào số lượng đặc trưng. Nếu số lượng đặc trưng đầu vào là hai thì siêu phẳng chỉ là một đường thẳng. Nếu số lượng đặc trưng đầu vào là ba thì siêu phẳng sẽ trở thành mặt phẳng 2-D.



*Hình 1 - Dạng mô hình máy học véc-tơ hỗ trợ*

Các thuật ngữ quan trọng trong giải thuật SVM:

- Siêu phẳng: Siêu phẳng là ranh giới quyết định được sử dụng để phân tách các điểm dữ liệu của các lớp khác nhau trong một không gian đặc trưng. Trong trường hợp phân loại tuyến tính, nó sẽ là một phương trình tuyến tính, tức là $wx + b = 0$, trong đó, w đại diện cho véc-tơ pháp tuyến của siêu phẳng, tức là hướng vuông góc với siêu phẳng, còn b là độ lệch hoặc khoảng cách của siêu phẳng tính từ gốc dọc theo w.

- Các véc-tơ hỗ trợ: Các véc-tơ hỗ trợ là các điểm dữ liệu gần nhất với siêu phẳng, đóng vai trò quan trọng trong việc quyết định siêu phẳng và lề.

- Lề: Là khoảng cách giữa véc-tơ hỗ trợ và siêu phẳng. Mục tiêu chính của thuật toán là tối đa hóa lề. Biên độ rộng hơn thấy hiệu suất phân loại tốt hơn.

- Nhân: Hàm nhân là hàm toán học, được sử dụng để ánh xạ các điểm dữ liệu đầu vào ban đầu cho các không gian đặc trưng nhiều chiều, do đó, có thể dễ dàng tìm ra siêu phẳng ngay cả khi các điểm dữ liệu không thể phân tách tuyến tính trong đầu vào ban đầu không gian. Một số hàm nhân phổ biến là hàm tuyến tính, đa thức, hàm cơ sở xuyên tâm (RBF) và sigmoid.

- Lề cứng: Siêu phẳng lề tối đa hoặc siêu phẳng lề cứng là một siêu phẳng giúp phân tách chính xác các điểm dữ liệu của các danh mục khác nhau mà không có bất kỳ phân loại sai nào.

- Lề mềm: Khi dữ liệu không thể phân tách hoàn hảo hoặc chứ các ngoại lệ, SVM cho phép kỹ thuật lề mềm. Mỗi điểm dữ liệu có một phụ biến phụ được

đưa vào bởi công thức SVM biên mềm, giúp giảm nhẹ yêu cầu về biên nghiêm ngặt và cho phép phân loại sai hoặc vi phạm nhất định. Nó phát hiện ra sự thỏa hiệp giữa việc tăng lề và giảm vi phạm.

- C: Tối đa hóa lề và độ phân loại sai được cân bằng bởi tham số C trong SVM. Hậu quả của việc đi quá lề hoặc phân loại sai điểm dữ liệu sẽ được quyết định bởi C. C càng lớn thì hậu quả càng lớn, dẫn đến biên độ nhỏ hơn và có thể ít phân loại sai hơn.

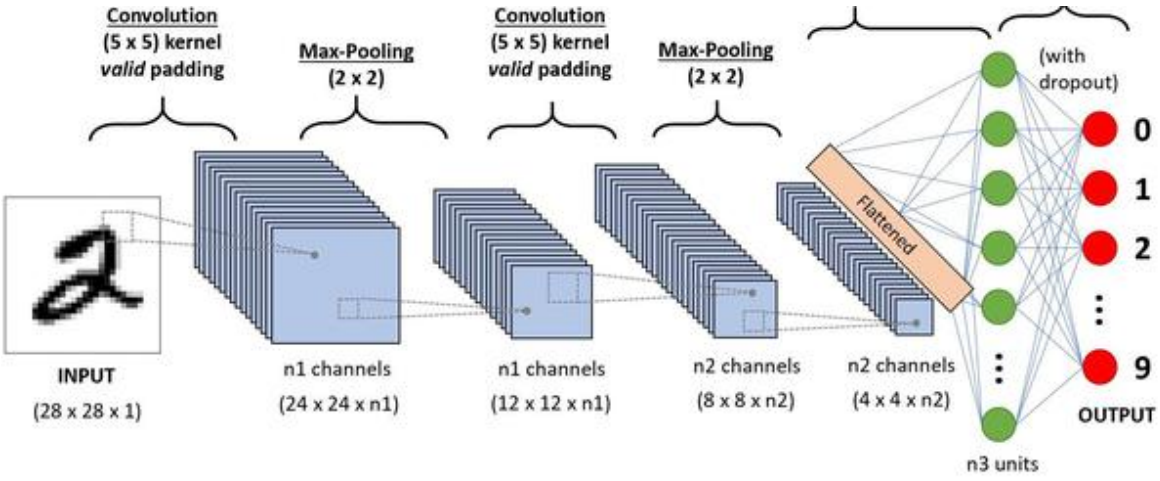
- Mất lề: Một hàm mất mát điển hình trong SVM là mất lề. Nó thể hiện cho việc phân loại không chính xác hoặc vi phạm lề. Hàm mục tiêu trong SVM thường được hình thành bằng việc kết hợp nó với thuật toán chính quy hóa.

## 2.2. Mô hình "Mạng nơ-ron tích chập"

Mạng nơ-ron tích chập (CNN) là một mạng nơ-ron nhân tạo được thiết kế đặc biệt cho việc xử lý dữ liệu không gian như hình ảnh và video. Mạng nơ-ron tích chập là phiên bản mở rộng của mạng nơ-ron nhân tạo (ANN) chủ yếu được sử dụng để trích xuất tính năng từ bộ dữ liệu ma trận giống như lưới.

Lịch sử phát triển của mạng nơ-ron tích chập như sau: Năm 1979, dựa trên để xuất về cấu trúc phân cấp của các tế bào, Fukushima đề xuất Neocognitron dùng để nhận diện ký tự Nhật viết tay. Có thể xem Neocognitron là mạng CNN đầu tiên. Tiếp theo đó, năm 1989, Yann Lecun đề xuất CNN có thể được huấn luyện bằng thuật toán lan truyền ngược. CNN vượt xa các mô hình khác tại kỳ thi ILSVRC (ImageNet Large Scale Recognition Challenge). Các mô hình CNN chiến thắng tại kỳ thi tương tự sau đó là: AlexNet (2012), ZFNet (2013), GoogleNet và VGG (2014) và ResNet (2015).

Mạng CNN hoạt động bằng cách áp dụng một loạt các lớp tích chập và gộp cho hình ảnh hoặc video đầu vào. Các lớp tích chập trích xuất các đặc trưng từ đầu vào bằng cách trượt một bộ lọc nhỏ trên hình ảnh hoặc video và tính toán tích số chấm giữa bộ lọc và đầu vào. Sau đó, các lớp gộp lại lấy mẫu đầu ra của các lớp tích chập để giảm tính chiều của dữ liệu và làm cho dữ liệu hiệu quả hơn về mặt tính toán. Việc sử dụng nhiều lớp tích chập trong CNN cho phép mạng tìm hiểu các tính năng ngày càng phức tạp từ hình ảnh hoặc video đầu vào. Các lớp tích chập đầu tiên tìm hiểu các tính năng đơn giản, chẳng hạn như các cạnh và góc. Các lớp tích chập sâu hơn tìm hiểu các tính năng phức tạp hơn, chẳng hạn như hình dạng và đối tượng.

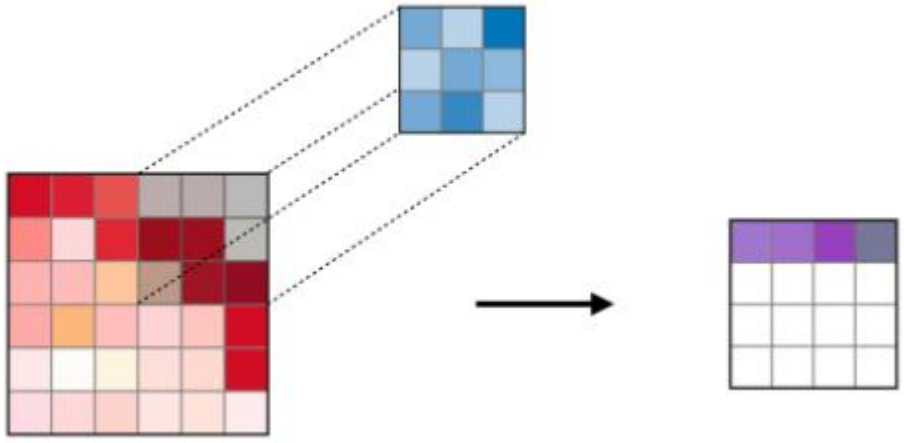


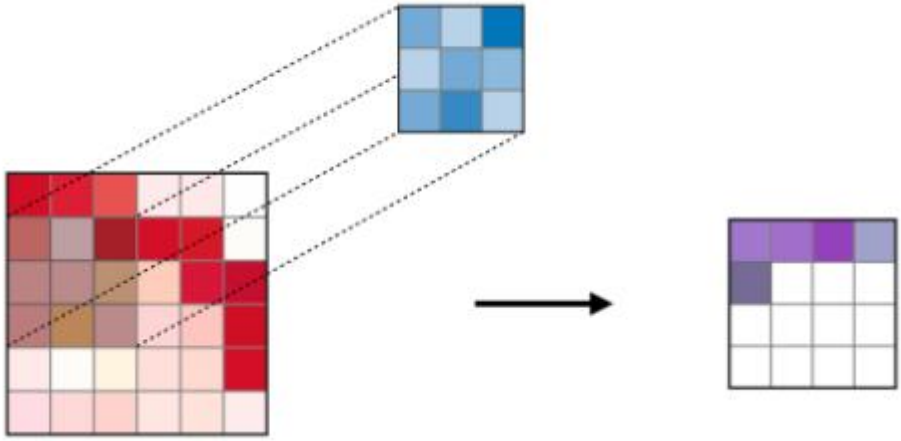

*Hình 2 - Một ví dụ về kiến trúc mạng CNN*

### 2.2.1. Tầng tích chập

Tầng tích chập thực hiện việc áp dụng các bộ lọc lên dữ liệu đầu vào để trích xuất các đặc trưng cụ thể. Mỗi bộ lọc sẽ quét qua đầu vào và tính toán tổng trọng số của các pixel tương ứng trong khu vực quét. Kết quả của mỗi phép tích chập là một ma trận.

Ví dụ, ma trận đầu vào có kích thước 6x6 khi đi qua bộ lọc 3x3 sẽ cho ra kết quả là một ma trận có kích thước 4x4 với bước nhảy là 1. Cụ thể, kích thước đầu ra được tính như sau:



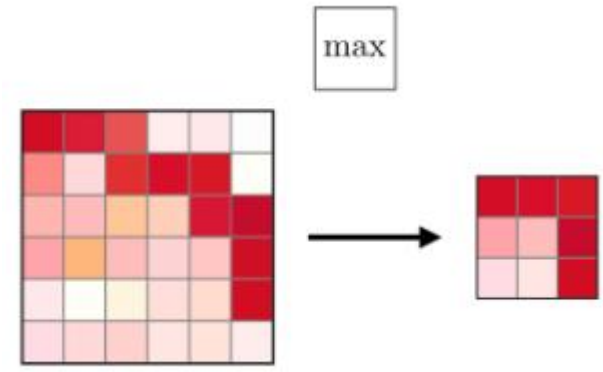


*Hình 3 - Phép tích chập dựa vào ma trận đầu vào với bộ lọc 3x3*

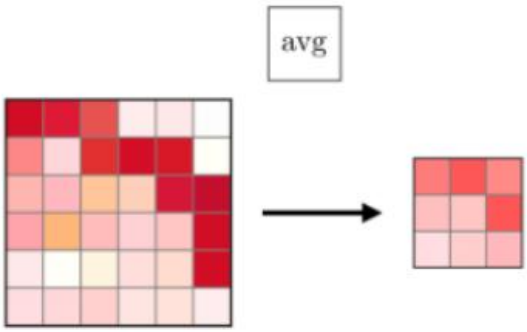
### 2.2.2. Tầng gộp

Tầng gộp trong mạng nơ-ron tích chập là một phần quan trọng giúp giảm kích thước của feature maps và giảm độ phức tạp của mạng, đồng thời giữ lại các đặc trưng quan trọng của dữ liệu.

Tầng gộp thường được sử dụng sau tầng tích chập để giảm kích thước của feature maps bằng cách chọn giá trị đại diện từ các vùng nhỏ của feature map. Có hai phương pháp gộp phổ biến là Max Pooling và Average Pooling. Trong Max Pooling, giá trị lớn nhất trong mỗi vùng được chọn làm giá trị đại diện để giữ lại các đặc trưng nổi bật và giảm nhiễu, trong khi đó, Average Pooling sẽ tính giá trị trung bình của các gái trị trong mỗi vùng được chọn làm giá trị đại diện nhằm giúp giảm độ nhạy cảm đối với nhiễu và giảm sự biến động giữa các đặc trưng. Thông thường, Max Pooling được sử dụng nhiều hơn so Average Pooling.




*Hình 4 - Tầng gộp áp dụng Max Pooling*




*Hình 5 - Tầng gộp áp dụng Average Pooling*

Bước nhảy ở tầng gộp cũng tương tự như tầng tích chập, nó sẽ quy định khoảng cách giữa các vùng gộp trong feature map. Bước nhảy lớn hơn sẽ giảm kích thước của feature map sau gộp. Ngoài ra, tầng gộp cũng có thể mở rộng đầu vào để duy trì kích thước của các feature map.

### 2.2.3. Tầng kết nối đầy đủ

Tầng kết nối đầy đủ là một thành phần quan trọng của mạng nơ-ron tích chập, thường được sử dụng ở cuối của mạng để chuyển đổi các đặc trưng được trích xuất từ các tầng trước đó thành đầu ra dự đoán thông qua các kết nối đầy đủ giữa nơ-ron và quá trình học từ dữ liệu huấn luyện.

Tầng kết nối đầy đủ bao gồm một số lượng lớn các nơ-ron, mỗi nơ-ron kết nối với tất cả các nơ-ron trước đó. Các nơ-ron này được kích hoạt bằng các hàm kích hoạt phi tuyến tính như hàm ReLU, sigmoid hoặc tanh.

Mỗi kết nối giữa các nơ-ron trong tầng kết nối đầy đủ có một trọng số tương ứng, biểu thị mức độ quan trọng của đặc trưng đó đối với đầu ra dự đoán. Ngoài ra, mỗi nơ-ron còn có một bias, một giá trị được thêm vào đầu vào trước khi áp dụng hàm kích hoạt, giúp tăng tính linh hoạt của mô hình.

Kết quả đầu ra của mô hình được tính toán từ đầu ra của tầng kết nối đầy đủ, thường thông qua một hàm kích hoạt cuối cùng như Softmax (trong phân loại nhiều lớp) hoặc Sigmoid (trong phân loại hai lớp).

## 2.3. Các hàm kích hoạt

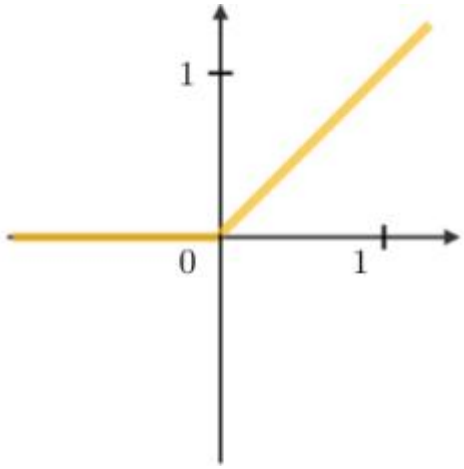
### 2.3.1. Hàm ReLU

Hàm ReLU là một hàm kích hoạt phi tuyến tính phổ biến, thường được sử dụng trong các tầng ẩn của mạng nơ-ron, bao gồm cả mạng nơ-ron tích chập. Hàm ReLU có công thức toán học như sau:

$$f(x) = max(0, x)$$

Trong đó:

- x: Là đầu vào của hàm ReLU.
- f(x): Là giá trị đầu ra của hàm ReLU.




*Hình 6 - Đồ thị hàm ReLU*

Hàm ReLU chỉ đơn giản trả về giá trị của đầu vào nếu đầu vào là dương, và trả về 0 nếu đầu vào là âm. Điều này có nghĩa là hàm ReLU sẽ giữ lại tất cả các giá trị dương không thay đổi và chặn tất cả các giá trị âm thành 0.

Ưu điểm của hàm ReLU bao gồm tính toán đơn giản, tính toán nhanh chóng và khả năng chống lại hiện tượng biến mất gradient trong quá trình huấn luyện. Đặc biệt, hàm ReLU thường được ưa chuộng trong các mạng nơ-ron sâu vì khả năng hỗ trợ huấn luyện hiệu quả.

### 2.3.2. Hàm Softmax

Hàm Softmax là một hàm kích hoạt thường được sử dụng trong tầng đầu ra của các mạng nơ-ron, đặc biệt là với bài toán phân loại nhiều lớp. Hàm này chuyển đổi giá trị đầu ra của mạng thành một phân phối xác suất trên các lớp khác nhau.

Công thức toán học của hàm Softmax cho một véc-tơ đầu vào x là:

$$Softmax(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}$$

Trong đó:

- $x_i$: Là phần tử thứ *i* của véc-tơ đầu vào x.
- N: Là số lượng lớp.
- $Softmax(x)_i$: Là giá trị của phần tử thứ *i* của véc-tơ đầu ra sau khi áp dụng hàm Softmax.

Hàm Softmax thực hiện hai bước chính: Tính tổng của tất cả các phần tử trong véc-tơ đầu vào x, sau đó, chia từng phần tử của véc-tơ cho tổng này, biến đổi chúng thành các giá trị xác suất. Điều này có nghĩa là mỗi phần tử của véc-tơ đầu ra

sau khi áp dụng hàm Softmax sẽ nằm trong khoảng [0, 1] và tổng của tất cả các phần tử sẽ bằng 1, tạo thành một phân phối xác suất. Đây là lí do hàm Softmax được lựa chọn trong các bài toán phân loại nhiều lớp, nơi mô hình cần dự đoán xác suất của mỗi lớp.

## 2.4. Các chỉ số đánh giá

### 2.4.1. Ma trận nhầm lẫn (Confusion matrix)

Ma trận nhầm lẫn là một bảng có kích thước n x n, trong đó, n là số lượng lớp trong bài toán phân loại. Mỗi hàng của ma trận đại diện cho lớp thực tế, và mỗi cột đại diện cho lớp được dự đoán bởi mô hình.

| | | Mô hình dự đoán | |
|---|---|---|---|
| Thực tế | | Positive | Negative |
| | Positive | True Positive<br>(Dự đoán Đúng là Positive) | False Negative<br>(Dự đoán Sai là Negative) |
| | Negative | False Positive<br>(Dự đoán sai là Positive) | True Negative<br>(Dự đoán Đúng là Negative) |

*Bảng 1 - Ma trận nhầm lẫn*

Ma trận nhầm lẫn bao gồm các thành phần sau:

- True Positive (TP): Số lượng các trường hợp mô hình dự đoán đúng là tích cực (Positive) so với thực tế.
- False Positive (FP): Số lượng các trường hợp mô hình dự đoán sai là tích cực (Positive), nghĩa là thực tế là Negative nhưng mô hình dự đoán là Positive.
- True Negative (TN): Số lượng các trường hợp mô hình dự đoán đúng là tiêu cực (Negative) so với thực tế.
- False Negative (FN): Số lượng các trường hợp mô hình dự đoán sai là tiêu cực (Negative), nghĩa là thực tế là Positive nhưng mô hình dự đoán là Negative.

Ma trận nhầm lẫn cung cấp cái nhìn tổng quan về hiệu suất mô hình phân loại trên từng lớp riêng biệt, giúp phát hiện các vấn đề như dự đoán sai lớp nào nhiều nhất, hoặc dự đoán đúng lớp nào ít nhất.

### 2.4.2. Độ chính xác (Precision)

Độ chính xác: Là tỉ lệ số lượng dự đoán đúng tích cực (True Positive) trên tổng số lượng dự đoán là tích cực. Độ chính xác được tính bằng công thức:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

### 2.4.3. Tính nhớ - Tỷ lệ bao phủ (Recall)

Tính nhớ - Tỷ lệ bao phủ là tỉ lệ dự đoán đúng tích cực (True Positive) trên tổng số lượng thực tế là tích cực. Tỷ lệ bao phủ được tính bằng công thức:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

### 2.4.4. F1-score

F1-score là một chỉ số kết hợp giữa độ chính xác và tỷ lệ bao phủ, thường được sử dụng để đánh giá hiệu suất của mô hình phân loại.

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

### 2.4.5. Độ chính xác toàn cục (Accuracy)

Độ chính xác toàn cục là tỉ lệ số lượng dự đoán đúng trên tổng số lượng dự đoán. Accuracy được tính bằng công thức sau:

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Negative}$$

## 2.5. Xây dựng giao diện với Flask

Flask là một thư viện cung cấp những tính năng cơ bản cần thiết cho việc phát triển web được viết bằng Python, được thiết kế để giúp các nhà phát triển xây dựng các ứng dụng web một cách dễ dàng và nhanh chóng. Flask được phát triển bởi Armin Ronacher, và nó nổi tiếng với sự đơn giản, linh hoạt và khả năng mở rộng.

Các đặc điểm chính của Flask:

- Flask cung cấp những tính năng cơ bản cần thiết cho phát triển web nhưng không bao gồm hững thành phần không cần thiết. Điều này giúp cho Flask nhẹ và linh hoạt, cho phép các nhà phát triển chỉ thêm những gì họ thực sự cần thông qua các tiện ích mở rộng.

- Flask sử dụng hệ thống routing để xác định các URL khác nhau sẽ kích hoạt những hàm xử lý tương ứng. Điều này được thực hiện bằng cách sử dụng decorators trong Python.

- Flask sử dụng Jinja2 làm công cụ template mặc định, cho phép nhúng Python vào HTML và quản lý các phần giao diện của ứng dụng web.

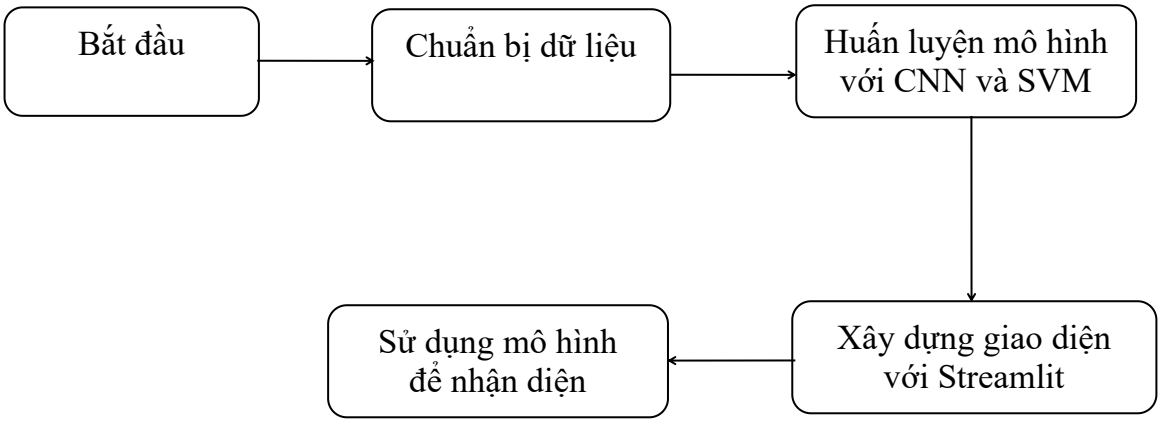
- Flask có một API đơn giản và dễ sử dụng, điều này làm cho nó phù hợp với cả người mới học và các nhà phát triển có kinh nghiệm.

Tóm lại, Flask là một lựa chọn mạnh mẽ và linh hoạt cho việc phát triển các ứng dụng web trong Python, từ các dự án nhỏ đến các hệ thống phức tạp.

# CHƯƠNG 2: THIẾT KẾ VÀ CÀI ĐẶT

## 1. Thiết kế hệ thống

### 1.1. Quá trình thực hiện

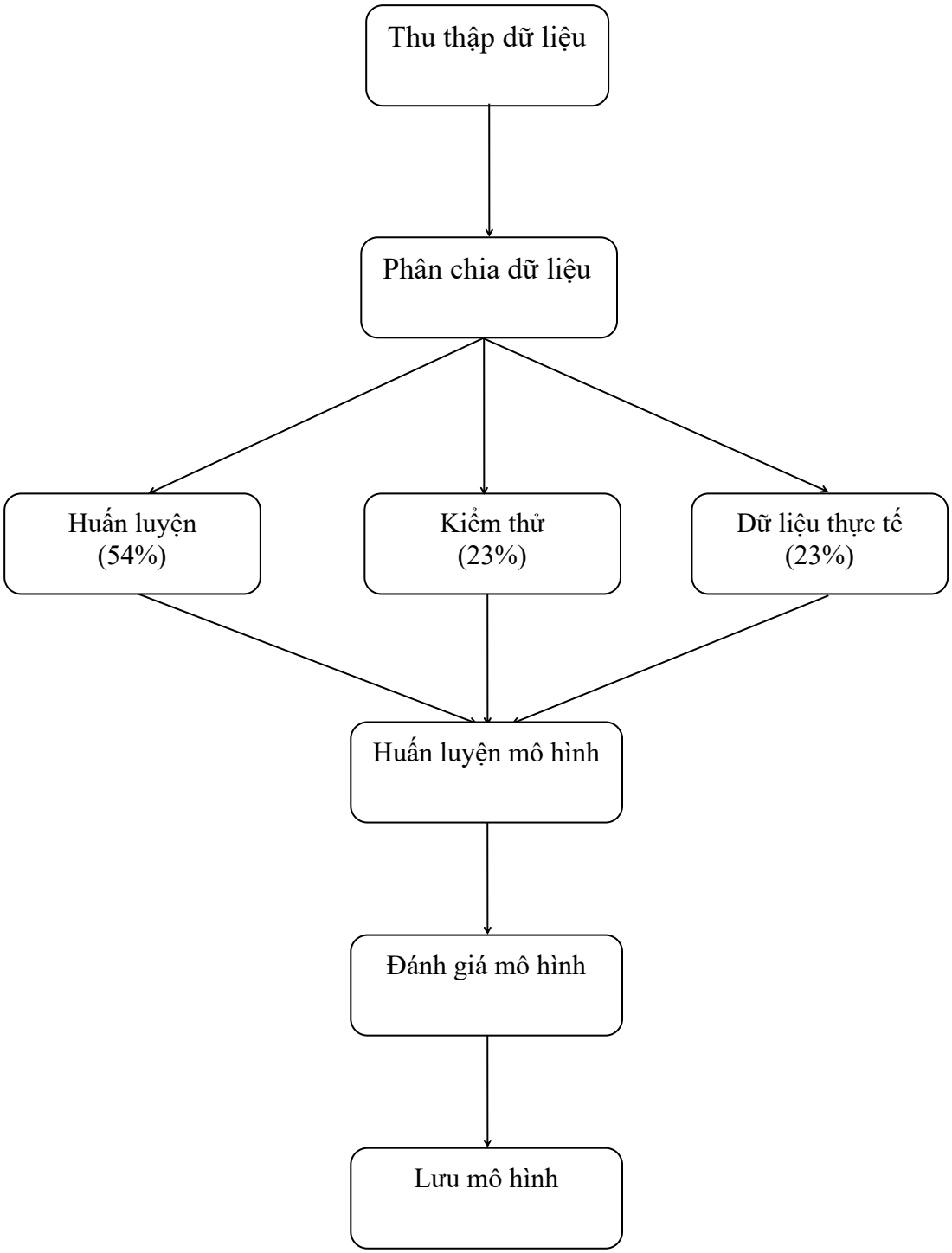



*Hình 7 - Sơ đồ quá trình thực hiện đề tài*

Quá trình thực hiện được miêu tả như hình. Đầu tiên, điều tất yếu của các mô hình nhận diện là tập dữ liệu. Sau khi có tập dữ liệu, thực hiện huấn luyện các mô hình CNN, SVM và lưu mô hình để sử dụng cho hệ thống. Tiếp theo, đem mô hình cho nhận diện với tập dữ liệu kiểm tra và đánh giá hiệu suất của mô hình. Cuối cùng, sử dụng Flask để xây dựng giao diện demo và tích hợp các mô hình vào trang web.

### 1.2. Quá trình huấn luyện mô hình:

Quá trình huấn luyện bắt đầu từ việc thu thập dữ liệu đến khi lưu được lưu hình. Dữ liệu là 43 loại biển báo giao thông ở Đức với các góc độ khác nhau và trong các điều kiện ánh sáng và thời tiết khác nhau. Sau khi đã có dữ liệu, dữ liệu sẽ được chia làm ba phần: Huấn luyện mô hình (53%), kiểm thử mô hình (23%) và dữ liệu thực tế để đánh giá hiệu suất mô hình (24%). Dữ liệu huấn luyện sẽ được đưa vào mô hình phân lớp SVM và mô hình học sâu CNN. Quá trình này bao gồm việc truyền dữ liệu vào mô hình, điều chỉnh các trọng số của mô hình và lặp lại đến khi mô hình đạt được hiệu suất mong muốn. Sau khi huấn luyện, mô hình sẽ được đưa qua tập dữ liệu kiểm thử để đánh giá mô hình đã học tốt trên dữ liệu huấn luyện hay không. Cuối cùng, khi đã kiểm thử mô hình, các mô hình được lưu lại và được sử dụng để phân loại các dữ liệu thực tế.




*Hình 8 - Quá trình huấn luyện mô hình*

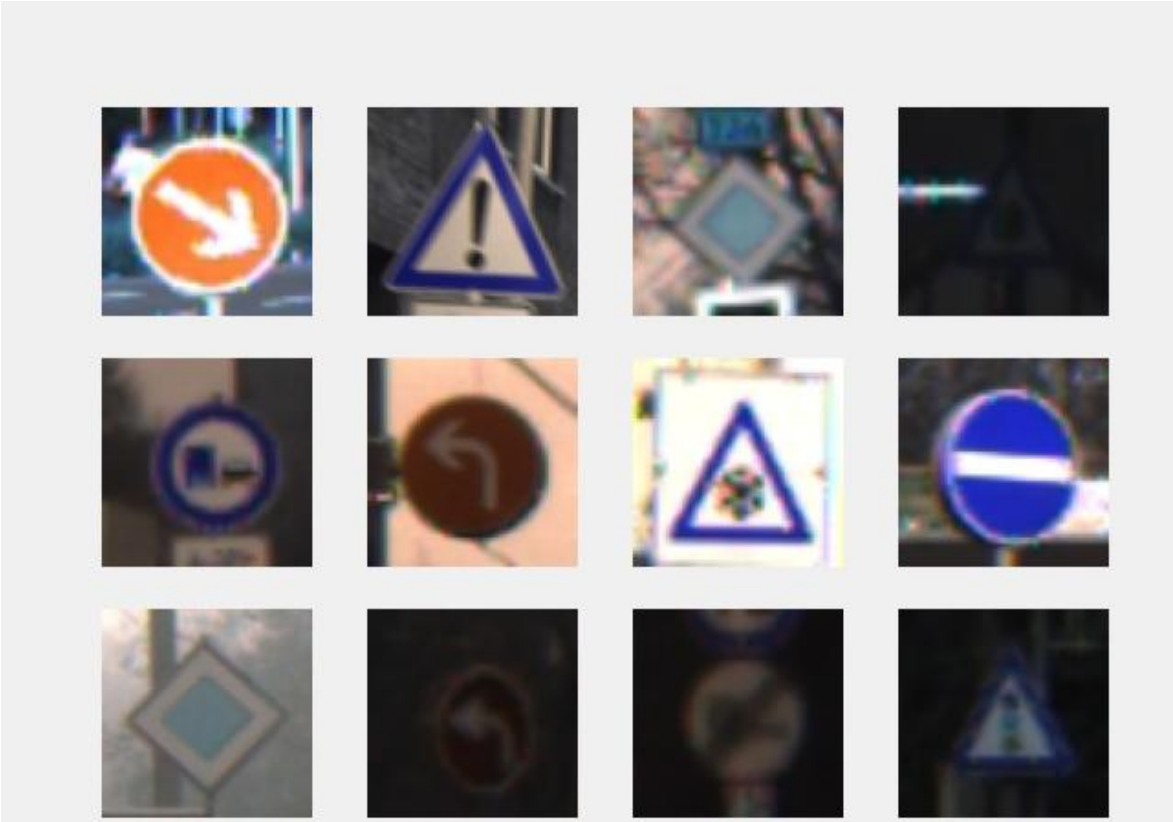
## 2. Cài đặt hệ thống

Tập dữ liệu bao gồm 43 loại biển báo giao thông ở Đức, cụ thể:

| Nhãn | Tên loại biển báo | Nhãn | Tên loại biển báo |
|---|---|---|---|
| 0 | Speed limit (20km/h) | 11 | Righ-of-way at intersection |
| 1 | Speed limit (30km/h) | 12 | Priority road |
| 2 | Speed limit (50km/h) | 13 | Yield |
| 3 | Speed limit (60km/h) | 14 | Stop |
| 4 | Speed limit (70km/h) | 15 | No vehicles |
| 5 | Speed limit (80km/h) | 16 | Veh > 3.5 tons prohibited |
| 6 | End of speed limit (80km/h) | 17 | No entry |
| 7 | Speed limit (100km/h) | 18 | General caution |
| 8 | Speed limit (120km/h) | 19 | Dangerous curve left |
| 9 | No passing | 20 | Dangerous curve right |
| 10 | No passing veh over 3.5 tons | 21 | Double curve |
| | | | |
| 22 | Bumpy road | 31 | Wild animals crossing |
| 23 | Slippery road | 32 | End speed + passing limits |
| 24 | Road narrows on the right | 33 | Turn right ahead |
| 25 | Road work | 34 | Turn left ahead |
| 26 | Traffic signals | 35 | Ahead only |
| 27 | Pedestrians | 36 | Go straight or right |
| 28 | Children crossing | 37 | Go straight or left |

| | | | |
|---|---|---|---|
| 29 | Bicycles crossing | 38 | Keep right |
| 30 | Beware of ice/snow | 39 | Keep left |
| 40 | Roundabout mandatory | | |
| 41 | End of no passing | | |
| 42 | End no passing veh > 3.5 tons | | |

*Bảng 2 - Các loại biển báo giao thông tại Đức*



*Hình 9 - Một số loại biển báo trong dữ liệu huấn luyện*

Như vậy, các mô hình chính được sử dụng trong hệ thống là các mô hình phân loại, cụ thể ở bài toán này là mô hình CNN và SVM. Bước đầu trong cài đặt hệ thống là xây dựng và huấn luyện mô hình.

## 2.1. Huấn luyện mô hình với CNN và SVM

### 2.1.1. Thu thập dữ liệu

Như đã đề cập trước đó, dữ liệu được lấy từ GTRSB với tổng cộng 51.839 hình ảnh bao gồm 43 loại biển báo. Số lượng hình ảnh của từng loại biển báo được thống kê như sau:

| Nhãn | Số lượng | Nhãn | Số lượng | Nhãn | Số lượng | Nhãn | Số lượng | Nhãn | Số lượng |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 270 | 9 | 1950 | 18 | 1590 | 27 | 300 | 36 | 510 |
| 1 | 2940 | 10 | 2670 | 19 | 270 | 28 | 690 | 37 | 270 |
| 2 | 3000 | 11 | 1740 | 20 | 450 | 29 | 360 | 38 | 2760 |
| 3 | 1860 | 12 | 2790 | 21 | 420 | 30 | 600 | 39 | 390 |
| 4 | 2640 | 13 | 2880 | 22 | 510 | 31 | 1050 | 40 | 450 |
| 5 | 2490 | 14 | 1050 | 23 | 660 | 32 | 300 | 41 | 300 |
| 6 | 570 | 15 | 840 | 24 | 360 | 33 | 899 | 42 | 330 |
| 7 | 1890 | 16 | 570 | 25 | 1980 | 34 | 540 | | |
| 8 | 1860 | 17 | 1470 | 26 | 780 | 35 | 1590 | | |

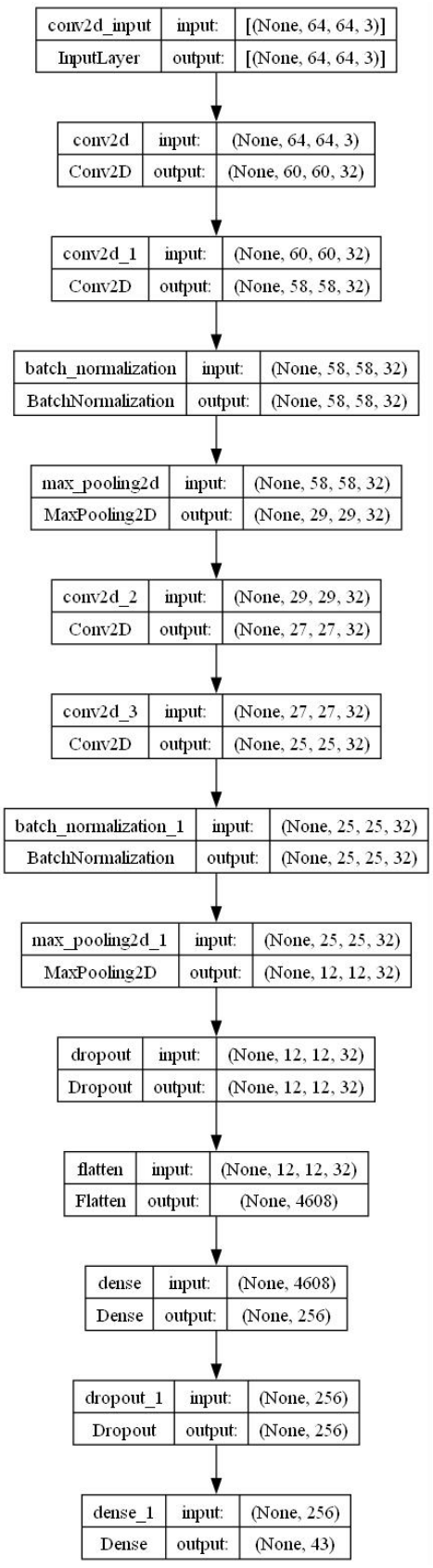
*Bảng 3 - Số lượng từng lớp đối tượng*

### 2.1.2. Huấn luyện mô hình

#### 2.1.2.1. Mô hình CNN

Mô hình CNN được xây dựng dựa trên thư viện Keras được tích hợp vào TensorFlow, một trong những thư viện mạnh mẽ và phổ biến nhất trong lĩnh vực học sâu. Bằng Keras, mô hình có thể được xây dựng nhanh chóng và dễ dàng thích ứng với nhu cầu cụ thể của bài toán, từ việc xây dựng các lớp tích chập để trích xuất đặc trưng từ dữ liệu ảnh đến việc tạo ra các kết nối đầy đủ để thực hiện phân loại cuối cùng. Đồng thời, việc tích hợp vào TensorFlow có thể tận dụng tối đa các tính năng mạnh mẹ của nó như tối ưu hóa hiệu suất, đào tào trên nhiều thiết bị và triển khai trên nhiều nền tảng khác nhau.

Mô hình CNN cho bài toán "Nhận diện biển báo gia thông" được mô tả như hình bên dưới:




*Hình 10 - Kiến trúc mạng cho bài toán nhận diện biển báo giao thông*

Đầu tiên, mô hình được khởi tạo bằng cách sử dụng lớp Sequential(), cho phép chồng các lớp nơ-ron một cách tuần tự.

Tiếp theo, mô hình bao gồm một chuỗi các lớp tích chập 2D (Conv2D), là các lớp quan trọng để học các đặc trưng từ hình ảnh. Trong trường hợp này, có tổng cộng ba lớp tích chập. Mỗi lớp tích chập sử dụng hàm kích hoạt ReLU (activation='relu') để tăng tính phi tuyến tính và đồng thời giữ các giá trị dương. Điều này giúp mô hình học được các đặc trưng phức tạp từ dữ liệu hình ảnh.

Sau mỗi lớp tích chập, lớp BatchNormalization() được áp dụng để chuẩn hóa các đặc trưng đầu ra, giúp cải thiện sự ổn định và tốc độ hội tụ của mô hình.

Tiếp theo là các lớp MaxPooling2D, được sử dụng để giảm kích thước của đầu ra từ các lớp tích chập trước đó. Điều này giúp giảm độ phức tạp của mô hình và làm giảm thời gian tính toán.

Sau khi áp dụng các lớp tích chập và lớp gộp, một lớp Dropout được thêm vào để ngẫu nhiên loại bỏ một phần tỷ lệ các nơ-ron trong quá trình đào tạo. Điều này giúp tránh hiện tượng overfitting bằng cách ngăn chặn sự phụ thuộc quá mức vào một số nơ-ron cụ thể.

Sau khi các lớp tích chập, gộp và dropout, đầu ra được làm phẳng hóa thành một vector 1 chiều bằng cách sử dụng lớp Flatten(), để chuẩn bị cho việc đưa vào các lớp kết nối đầy đủ.

Các lớp kết nối đầy đủ (Dense) được sử dụng để kết hợp các đặc trưng đã học từ dữ liệu và áp dụng chúng vào việc phân loại hình ảnh. Các lớp này sử dụng hàm kích hoạt ReLU.

Sau lớp kết nối đầy đủ cuối cùng, một lớp kết nối đầy đủ khác được thêm vào với số lượng đơn vị bằng số lượng lớp trong tập dữ liệu, và hàm kích hoạt softmax được sử dụng để tính toán xác suất cho mỗi lớp.

Sau khi xây dựng mô hình, tối ưu hóa Adam được sử dụng để điều chỉnh trọng số của mạng nơ-ron trong quá trình đào tạo. Hàm mất mát được chọn là categorical_crossentropy, phù hợp với bài toán phân loại nhiều lớp.

Tiếp theo là quá trình đào tạo mô hình sử dụng dữ liệu huấn luyện và dữ liệu kiểm thử. Dữ liệu được chia thành các batch có kích thước 32 và được huấn luyện qua nhiều lần lặp.

#### 2.1.2.2. Mô hình SVM

Mô hình SVM được xây dựng trong thư viện scikit-learn, một thư viện phổ biến dành cho máy học và khai mỏ dữ liệu. Scikit-learn cung cấp một công cụ mạnh mẽ để xử lý và phân tích dữ liệu, cũng như triển khai các thuật toán tiêu biểu.

Đối với bài toán này, khi đầu vào là hình ảnh, cần trích xuất đặc trưng từ hình ảnh để mô hình SVM có thể học được. Đến đây, có rất nhiều cách trích xuất đặc trưng tốt, nhưng trích xuất đặc trưng bằng mạng CNN là phương pháp được chọn do CNN có khả năng học các đặc trưng cấp cao từ dữ liệu hình ảnh thông qua quá trình lan truyền ngược (back propagation) trong quá trình huấn luyện. CNN đã được chứng minh là hiệu quả đối với các bài toán nhận dạng và phân loại đối tượng.

Để trích xuất đặc trưng từ CNN đã huấn luyện trước đó, tạo một mô hình mới với đầu vào là một phần của mô hình ban đầu. Phần này bao gồm tất cả các lớp từ đầu đến lớp cuối cùng, kết quả của lớp này là đầu ra được sử dụng làm đặc trưng đầu vào cho mô hình SVM.

Sử dụng mô hình mới tạo để trích xuất các đặc trưng từ dữ liệu huấn luyện và dữ liệu thực tế. Các đặc trưng này lưu lần lượt là *feat_train* và *feat_test*.

Sau đó, khởi tạo mô hình SVM với các tham số như nhân là "rbf", tham số C là 1000 và gamma là 0.001. Nhân "rbf" là một trong những nhân phổ biến nhất được sử dụng trong SVM. Nó cho phép phân loại phi tuyến tính bằng cách ánh xạ dữ liệu vào không gian cao chiều với các hàm Gaussian. Tham số C điều chỉnh độ linh hoạt của biên quyết định trong SVM. Khi C càng lớn, mô hình sẽ cố gắng phân loại mọi điểm dữ liệu tối ưu nhất có thể. Tham số gamma quyết định độ ảnh hưởng của một điểm dữ liệu đến hình dạng của siêu mặt phẳng. Khi gamma thấp, mô hình được xem xét các điểm xa hơn, ảnh hưởng đến hình dạng của biên quyết định một cách rộng hơn.

Cuối cùng, sau khi huấn luyện, mô hình sẽ được thực hiện với dữ liệu *feat_test* và độ chính xác của mô hình được tính bằng chỉ số accuracy.

# CHƯƠNG 3: KIỂM THỬ VÀ ĐÁNH GIÁ

## 1. Môi trường thực nghiệm

Để xây dựng, kiểm tra hệ thống và mô hình huấn luyện, máy tính (laptop) có cấu hình và các thư viện được cài đặt như sau:

### 1.1. Cấu hình máy

- 11th Gen Intel(R) Core(TM) i5-1155G7 @ 2.50GHz  2.50 GHz.
- Hệ điều hành: Windows 11, 64 bit.

### 1.2. Các thư viện đã sử dụng

| Tên thư viện | Phiên bản |
|---|---|
| numpy | 1.26.3 |
| pandas | 2.2.0 |
| matplotlib | 3.8.2 |
| seaborn | 0.13.1 |
| tensorflow | 2.15.0 |
| scikit-learn | 1.3.2 |
| joblib | 1.3.2 |
| streamlit | 1.34.0 |

*Bảng 4 - Thư viện sử dụng*

## 2. Kết quả mô hình

### 2.1. Mô hình CNN

Sau khi huấn luyện mô hình với 20 lần lặp, mô hình sẽ được kiểm tra với dữ liệu thực tế và thu được bảng báo cáo sau:

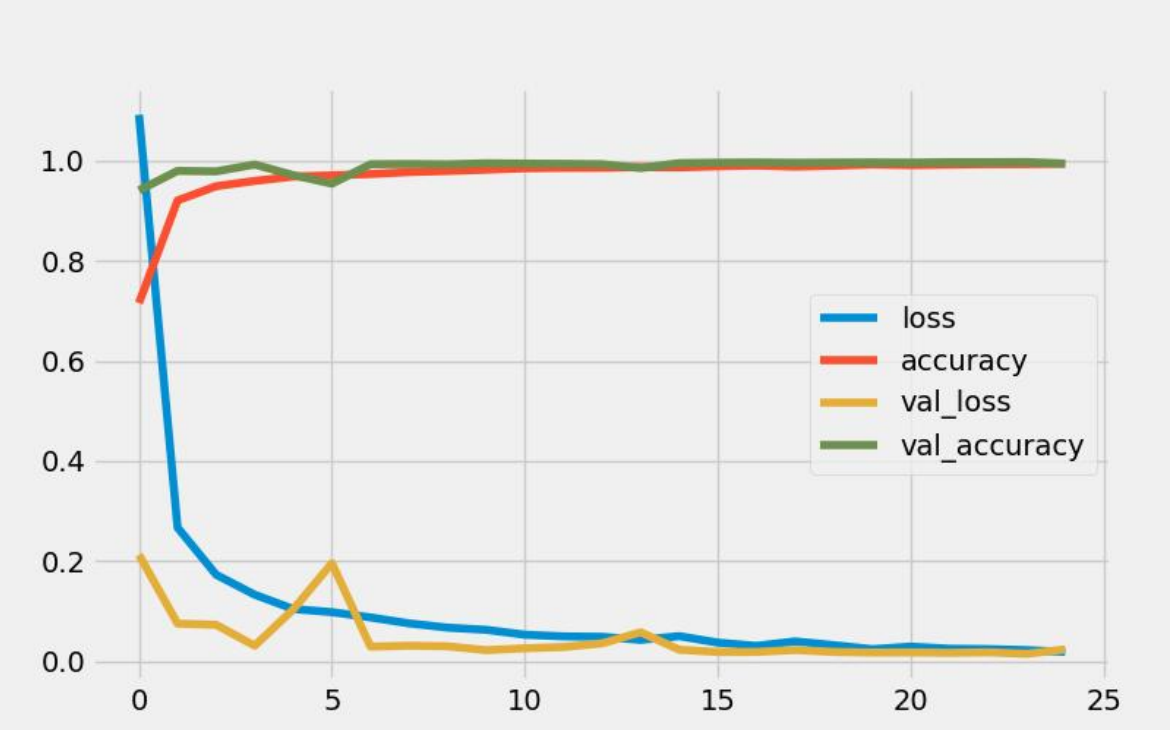
| | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Precision | 1.00 | 0.98 | 0.99 | 0.97 | 1.00 | 0.93 | 1.00 | 0.96 | 0.98 | 0.97 | 1.00 |
| Recall | 0.97 | 0.99 | 0.99 | 0.95 | 0.98 | 0.99 | 0.85 | 0.97 | 0.98 | 1.00 | 0.99 |
| F1-score | 0.98 | 0.99 | 0.99 | 0.96 | 0.99 | 0.96 | 0.92 | 0.97 | 0.98 | 0.98 | 1.00 |
| | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** |
| Precision | 0.98 | 1.00 | 0.98 | 1.00 | 0.95 | 1.00 | 1.00 | 0.99 | 0.98 | 0.91 | 0.97 |
| Recall | 0.99 | 0.97 | 1.00 | 1.00 | 1.00 | 0.99 | 1.00 | 0.97 | 1.00 | 1.00 | 1.00 |
| F1-score | 0.98 | 0.99 | 0.99 | 1.00 | 0.97 | 1.00 | 1.00 | 0.98 | 0.99 | 0.95 | 0.98 |

| | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Precision | 0.83 | 0.93 | 0.99 | 0.99 | 0.96 | 0.97 | 0.97 | 0.97 | 0.97 | 0.99 | 0.94 |
| Recall | 0.99 | 0.99 | 0.97 | 0.97 | 0.85 | 1.00 | 0.97 | 0.98 | 0.85 | 0.99 | 1.00 |
| F1-score | 0.9 | 0.96 | 0.98 | 0.98 | 0.90 | 0.98 | 0.98 | 0.97 | 0.90 | 0.99 | 0.97 |
| | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** | **41** | **42** | |
| Precision | 0.99 | 0.99 | 1.00 | 0.99 | 0.98 | 0.99 | 0.95 | 0.99 | 0.78 | 0.95 | |
| Recall | 0.99 | 0.99 | 0.99 | 0.97 | 1.00 | 0.99 | 0.99 | 0.89 | 0.77 | 0.86 | |
| F1-score | 0.99 | 0.99 | 1.00 | 0.98 | 0.99 | 0.99 | 0.97 | 0.93 | 0.77 | 0.90 | |

| | | |
|---|---|---|
| Accuracy | | 0.98 |
| Macro avg | Precision | 0.97 |
| | Recall | 0.97 |
| | F1-score | 0.97 |
| Weighted avg | Precision | 0.98 |
| | Recall | 0.98 |
| | F1-score | 0.98 |

*Bảng 5 - Bảng báo cáo mô hình CNN*

Dựa vào báo cáo đánh giá hiệu suất của mô hình phân loại CNN trên tập dữ liệu đã gắn nhãn, có thể nhận xét rằng mô hình đã đạt được hiệu suất tốt trên hầu hết các lớp. Độ chính xác trung bình của mô hình đạt khoảng 98%, chỉ ra khả năng dự đoán chính xác cao trên hầu hết các lớp. Các chỉ số Precision, Recall và F1-score đều ở mức cao, cho thấy mô hình có khả năng dự đoán chính xác và phát hiện đúng các trường hợp tích cực và tiêu cực. Tuy nhiên, có một số lớp có precision và recall thấp hơn, cần được xem xét kỹ lưỡng hơn để cải thiện hiệu suất tổng thể của mô hình. Số lượng mẫu trong mỗi lớp cũng đa dạng, từ những lớp có ít mẫu đến những lớp có nhiều mẫu, điều này có thể ảnh hưởng đến hiệu suất của mô hình trên các lớp có ít dữ liệu. Tổng quan, mô hình CNN đã đạt được hiệu suất tốt trong việc phân loại các lớp dữ liệu.




*Hình 11 - Kết quả giữa tập huấn luyện và tập kiểm thử*

Dựa vào biểu đồ, có thể thấy rằng khi số lần lặp càng gần về cuối thì độ chính xác của tập huấn luyện càng cao (gần chạm đến 1.0), theo đó, độ chính xác của tập kiểm thử cũng tăng dần theo độ chính xác của tập dữ liệu. Điều này cho biết rằng mô hình đã học tốt các đặc trưng cần thiết của dữ liệu huấn luyện và có khả năng tổng quát hóa tốt trên dữ liệu kiểm thử. Việc độ chính xác của tập kiểm thử tăng dần và tiến đến độ chính xác tập huấn luyện là một dấu hiệu tích cực, cho thấy rằng mô hình không bị "học vẹt". Tổng quan, mô hình sau khi huấn luyện đã có thể học tốt và sẵn sàng kiểm chứng với dữ liệu thực tế.

## 2.2. Mô hình SVM:

| | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Precision | 1.00 | 0.98 | 0.97 | 0.97 | 0.99 | 0.96 | 1.00 | 0.98 | 0.98 | 0.98 | 0.99 |
| Recall | 0.92 | 0.98 | 0.99 | 0.98 | 0.98 | 0.96 | 0.86 | 0.96 | 0.96 | 0.99 | 0.99 |
| F1-score | 0.96 | 0.98 | 0.98 | 0.98 | 0.98 | 0.96 | 0.92 | 0.97 | 0.97 | 0.99 | 0.99 |
| | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** |
| Precision | 0.92 | 1.00 | 0.99 | 1.00 | 0.98 | 1.00 | 1.00 | 0.98 | 0.98 | 0.84 | 1.00 |
| Recall | 0.99 | 0.99 | 1.00 | 1.00 | 1.00 | 0.99 | 1.00 | 0.92 | 1.00 | 0.96 | 0.83 |
| F1-score | 0.95 | 1.00 | 0.99 | 1.00 | 0.99 | 1.00 | 0.99 | 1.00 | 0.99 | 0.90 | 0.91 |
| | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** |
| Precision | 0.97 | 0.89 | 0.93 | 0.97 | 0.98 | 0.88 | 0.98 | 0.99 | 0.96 | 0.98 | 0.90 |
| Recall | 0.95 | 1.00 | 0.98 | 0.95 | 0.88 | 0.97 | 0.99 | 0.99 | 0.84 | 0.99 | 1.00 |
| F1-score | 0.96 | 0.94 | 0.95 | 0.96 | 0.93 | 0.92 | 0.99 | 0.99 | 0.90 | 0.99 | 0.94 |
| | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** | **41** | **42** | |
| Precision | 1.00 | 0.99 | 0.99 | 0.99 | 0.97 | 1.00 | 0.93 | 0.92 | 0.86 | 0.98 | |
| Recall | 0.99 | 0.99 | 1.00 | 1.00 | 1.00 | 0.99 | 1.00 | 0.96 | 0.90 | 0.90 | |
| F1-score | 0.99 | 0.99 | 1.00 | 1.00 | 0.98 | 1.00 | 0.96 | 0.94 | 0.88 | 0.94 | |

| | | |
|---|---|---|
| Accuracy | | 0.98 |
| Macro avg | Precision | 0.97 |
| | Recall | 0.97 |
| | F1-score | 0.96 |
| Weighted avg | Precision | 0.98 |
| | Recall | 0.98 |
| | F1-score | 0.98 |

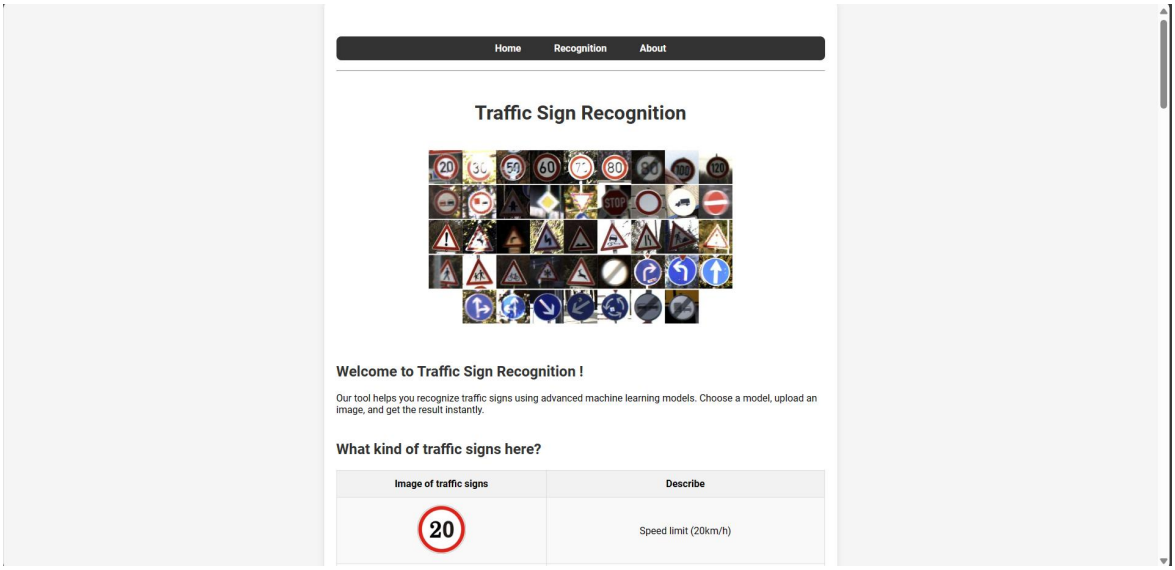
*Bảng 6 - Bảng báo cáo mô hình SVM*

Dựa vào báo cáo đánh giá hiệu suất của mô hình phân loại SVM trên tập dữ liệu đã gắn nhãn, có thể nhận xét rằng mô hình đã đạt được hiệu suất tốt trên hầu hết các lớp. Độ chính xác trung bình của mô hình đạt khoảng 98%, chỉ ra khả năng dự đoán chính xác cao trên hầu hết các lớp. Các chỉ số Precision, Recall và F1-score

đều ở mức cao, cho thấy mô hình có khả năng dự đoán chính xác và phát hiện đúng các trường hợp. Tổng quan, mô hình SVM đã đạt được hiệu suất tốt trong việc phân loại các lớp dữ liệu.

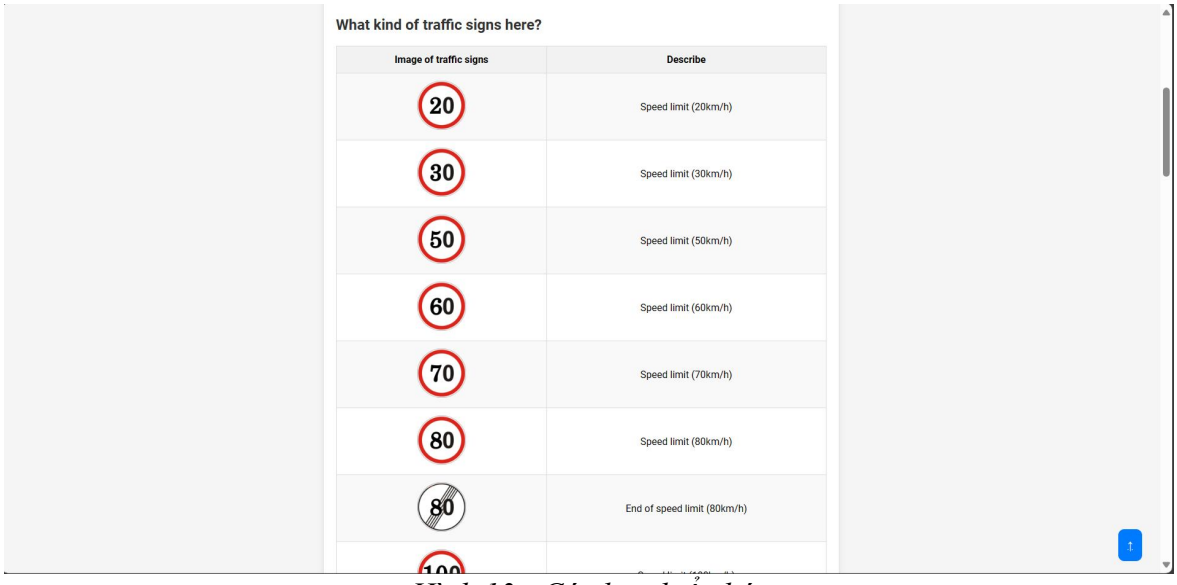
## 3. Giao diện hệ thống

Giao diện là một phần không thể thiếu đối với hệ thống, dù hệ thống có ít tính năng thì vẫn có giao diện. Giao diện nhằm giúp cho người dùng tương tác với hệ thống thuận tiện và dễ dàng hơn. Trong đề tài này, hệ thống sử dụng thư viện Flask để xây dựng giao diện.

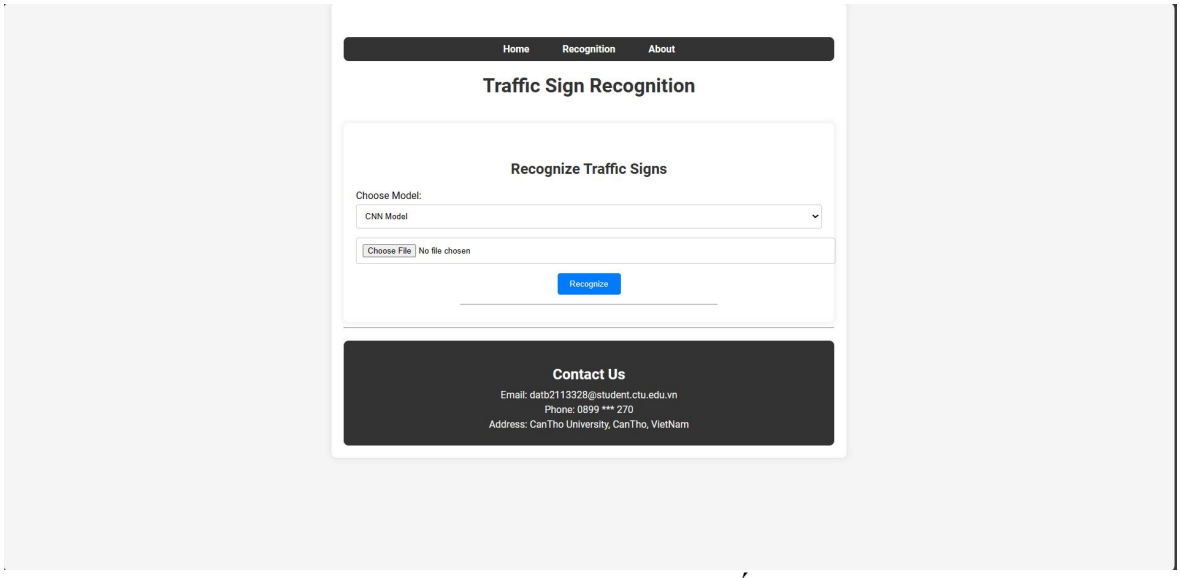



*Hình 12 - Giao diện hệ thống*

Hình 12 mô tả giao diện hệ thống "Nhận diện biển báo giao thông" gồm có các trang như trang chủ (Home) để giới thiệu sơ lược về trang web, trang nhận diện (Regconition) để nhận diện các hình ảnh đầu vào và trang công dụng của trang web (About).

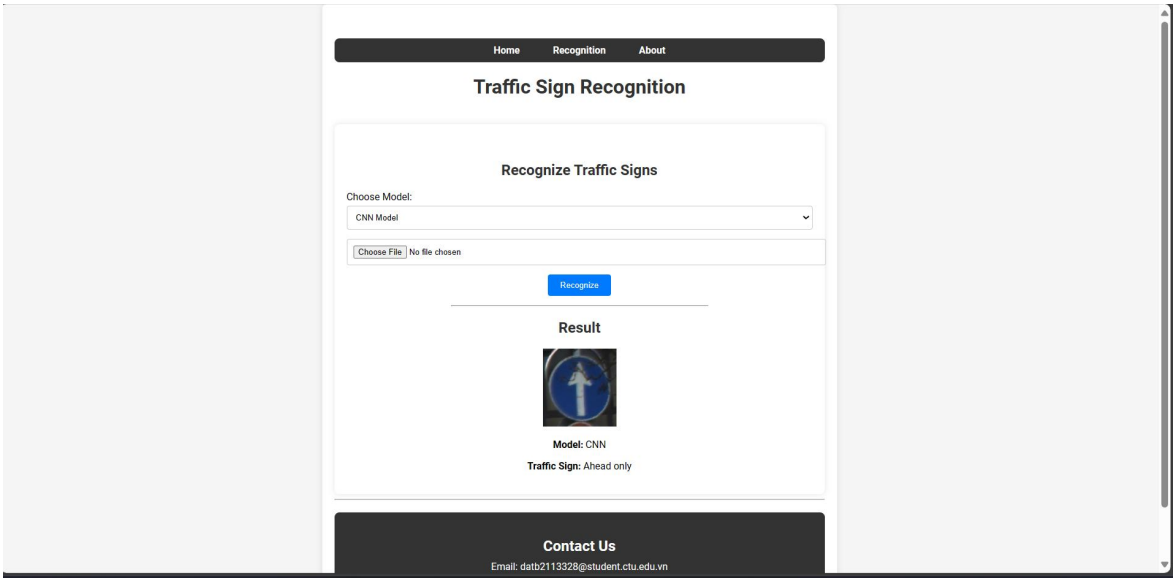



*Hình 13 - Các loại biển báo*

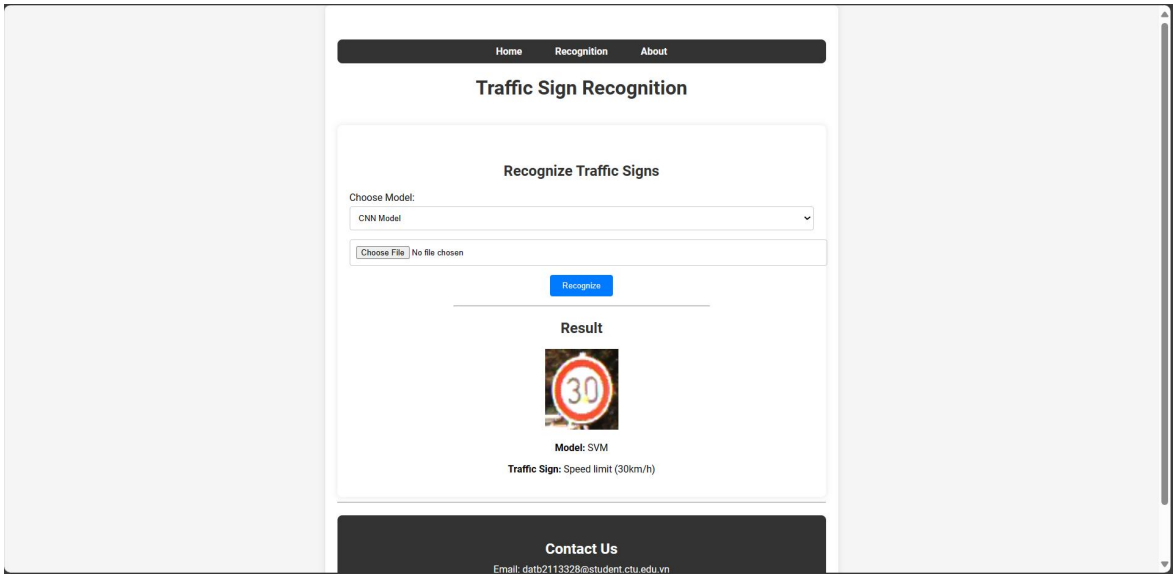



*Hình 14 - Giao diện hệ thống*


*Hình 15 - Giao diện khi chọn ảnh đầu vào*




*Hình 16 - Giao diện khi dự đoán ảnh với CNN*




*Hình 17 - Giao diện khi dự đoán ảnh với SVM*

# PHẦN KẾT LUẬN

## 1. Kết quả đạt được

- ✓ Xây dựng được hệ thống "Nhận diện biển báo giao thông" với các chức năng đơn giản.
- ✓ Nhận diện được các loại biển báo và hiệu suất các mô hình khá cao.
- ✓ Xây dựng được giao diện người dùng trên nền tảng ứng dụng.

## 2. Hướng phát triển

- ✓ Cải thiện độ chính xác của mô hình cao hơn nữa.
- ✓ Có thể xem lại tất cả các dự đoán trước của mô hình trên giao diện.
- ✓ Xây dựng tập dữ liệu lớn hơn với nhiều loại biển báo giao thông hơn.
- ✓ Phát triển được ứng dụng lên nền tảng Android.

# TÀI LIỆU THAM KHẢO

[1] Bùi Chí Thông, "Xây dựng ứng dụng phát hiện một số côn trùng gây hại trên cây ăn quả ở đồng bằng sông Cửu Long với YOLOv8", Luận văn tốt nghiệp ngành Khoa học máy tính, Đại học Cần Thơ, Cần Thơ, 2023.

[2] Nguyễn Hữu Thành, "Nhận dạng các loại rau củ quả phổ biến ở Việt Nam", Niên luận cơ sở ngành Khoa học máy tính, Đại học Cần Thơ, Cần Thơ, 2023.

[3] Đỗ Thanh Nghị, Phạm Nguyên Khang - Giáo trình "Nguyên lý máy học", Cần Thơ, 2012