HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH

****KHOA CÔNG NGHỆ THÔNG TIN 2

**BÁO CÁO CUỐI KÌ**

**IOT VÀ ỨNG DỤNG**

**Đề tài: Nghiên cứu machine learning model pipelines (KEYLOGGER DETECTION - CNN)**

**Giảng viên hướng dẫn *:* *GV. Đàm Minh Lịnh***

**Nhóm sinh viên thực hiện *:* *Nhóm 14***

**Thành viên *: Văn Chí Khanh – N20DCCN032***

***Nguyễn Tiến Ngọc – N20DCCN046***

**Lớp *: D20CQCNPM01-N***

*Hồ Chí Minh ngày 05 tháng 01 năm 2024*

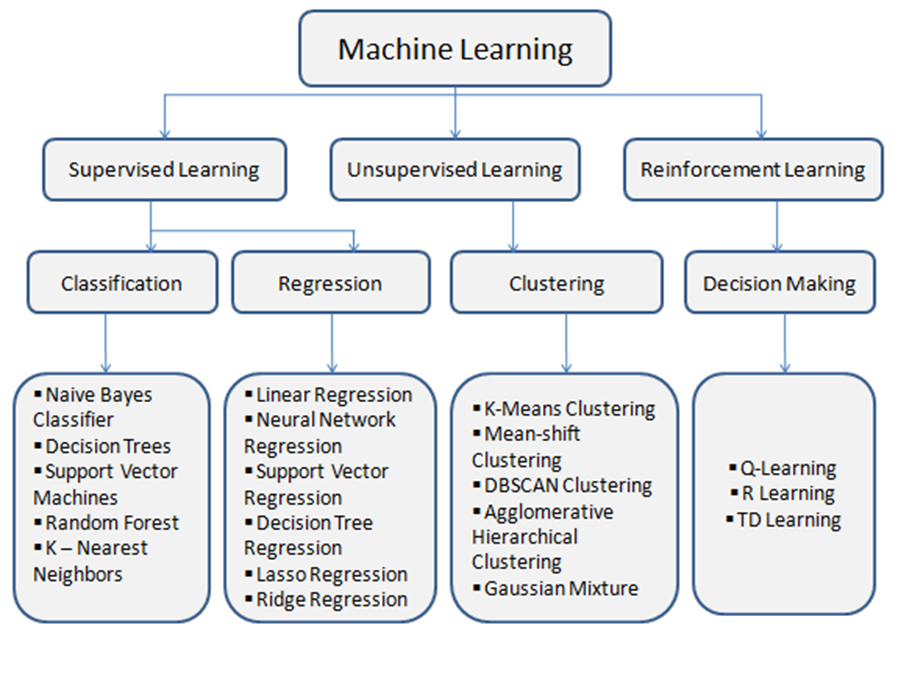
**LỜI CẢM ƠN**

Để hoàn thành tốt đề tài và bài báo cáo này, Em xin gửi lời cảm ơn chân thành đến giảng viên, thầy Đàm Minh Lịnh, trong quá trình học tập và tìm hiểu bộ môn Iot và Ứng dụng, em đã nhận được sự quan tâm giúp đỡ, hướng dẫn tận tình, tâm huyết của thầy. Thầy đã giúp em tích luỹ thêm nhiều kiến thức để có cái nhìn sâu sắc hơn về Iot và ứng dụng thực tiễn, người đã trực tiếp hỗ trợ em trong suốt quá trình học tập và làm đề tài. Em cảm ơn thầy về những bài giảng, những lời khuyên từ kinh nghiệm thực tiễn của mình để định hướng cho em đi đúng với yêu cầu của đề tài đã chọn.

Đề tài và bài báo cáo được nhóm em thực hiện trong khoảng thời gian ngắn, với những kiến thức còn hạn chế cùng nhiều hạn chế khác về mặt kĩ thuật và kinh nghiệm trong việc thực hiện một dự án IOT. Do đó, trong quá trình làm nên đề tài có những thiếu sót là điều không thể tránh khỏi nên em rất mong nhận được những ý kiến đóng góp quý báu của thầy để kiến thức của em được hoàn thiện hơn và em có thể làm tốt hơn nữa trong những lần sau.

Xin trân trọng cảm ơn!

1. **TỔNG QUAN VỀ MACHINE LEARNING (HỌC MÁY)**
2. **Khái niệm Machine Learning**
   * Machine learning – ML (Học máy) là một nhánh của trí tuệ nhân tạo (AI) và khoa học máy tính, tập trung vào việc sử dụng dữ liệu và thuật toán để bắt chước hành động của con người, dần dần cải thiện độ chính xác của nó.
   * Machine learning còn là một thành phần quan trọng của lĩnh vực khoa học dữ liệu đang phát triển. Thông qua việc sử dụng các phương pháp thống kê, các thuật toán được đào tạo để đưa ra các phân loại hoặc dự đoán và khám phá những thông tin chi tiết từ chính các dự án khai thác dữ liệu.
   * Bài toán của machine learning thường được chia làm hai loại là dự đoán (prediction) và phân loại (classification). Các bài toán dự đoán thường là giá nhà, giá xe, v.v, còn các bài toán phân loại thường là nhận diện chữ viết tay, đồ vật, v.v.
3. **Các phương pháp Machine Learning**

* Học máy có hai dạng là: Học máy có giám sát và học máy không có giám sát.

**2.1. Supervised Learning**

* Supervised learning còn được gọi là máy học có giám sát, được định nghĩa bằng cách sử dụng các tập dữ liệu được gắn nhãn để huấn luyện các thuật toán phân loại dữ liệu hoặc dự đoán kết quả một cách chính xác.
* Khi dữ liệu đầu vào được đưa vào mô hình, mô hình sẽ điều chỉnh trọng lượng của nó cho đến khi nó được lắp một cách thích hợp. Điều này xảy ra như một phần của quá trình xác nhận chéo để đảm bảo rằng mô hình tránh trang bị quá nhiều hoặc trang bị thiếu thông tin.
* Supervised machine learning giúp các tổ chức giải quyết nhiều vấn đề trong thế giới thực trên quy mô lớn, chẳng hạn như phân loại thư rác trong một thư mục riêng biệt từ hộp thư đến của bạn.
* Một số phương pháp được sử dụng trong Supervised machine learning bao gồm: logistic regression, neural networks, linear regression, naive bayes, random forest, và support vector machine (SVM).

**2.2.** **Unsupervised Learning**

* Unsupervised machine learning là phương pháp sử dụng các thuật toán máy học để phân tích và phân cụm các tập dữ liệu không được gắn nhãn.
* Không cần sự can thiệp của con người, các thuật toán này có thể phát hiện ra các mẫu hoặc nhóm dữ liệu ẩn. Khả năng phát hiện ra những điểm tương đồng và khác biệt trong thông tin của phương pháp này khiến nó trở nên lý tưởng cho việc phân tích dữ liệu khám phá, chiến lược bán chéo (cross-sell), phân khúc khách hàng cũng như nhận dạng hình ảnh và mẫu.
* Unsupervised machine learning cũng được sử dụng để giảm số lượng các tính năng trong một mô hình thông qua quá trình giảm kích thước. Phân tích thành phần chính (PCA) và phân tích giá trị đơn lẻ (SVD) là hai cách tiếp cận phổ biến cho việc này.
* Các thuật toán khác được sử dụng trong học tập không giám sát bao gồm: k-means clustering, neural networks, và probabilistic clustering methods.

**3. Các thuật toán phổ biến của Machine Learning**

* Neural networks: Mô phỏng cách thức hoạt động của bộ não con người, với một số lượng khổng lồ các nút xử lý được liên kết. Neural networks là thuật toán được dùng trong việc nhận dạng các mẫu và đóng một vai trò quan trọng trong các ứng dụng bao gồm dịch ngôn ngữ tự nhiên, nhận dạng hình ảnh, nhận dạng giọng nói và tạo hình ảnh.
* Linear regression: Thuật toán này được sử dụng để dự đoán các giá trị số, dựa trên mối quan hệ tuyến tính giữa các giá trị khác nhau.
* Logistic regression: Thuật toán giúp đưa ra dự đoán cho các biến phản hồi phân loại, chẳng hạn như câu trả lời “có/không” cho các câu hỏi. Nó có thể được sử dụng cho các ứng dụng như phân loại thư rác và kiểm soát chất lượng trên dây chuyền sản xuất.
* Clustering: Các thuật toán phân cụm có thể xác định các mẫu trong dữ liệu để nó có thể được nhóm lại. Máy tính có thể giúp các nhà khoa học dữ liệu bằng cách xác định sự khác biệt giữa các mục dữ liệu mà con người đã bỏ qua.
* Decision trees: Là thuật toán được sử dụng để dự đoán giá trị số (hồi quy) và phân loại dữ liệu. Decision trees sử dụng một chuỗi phân nhánh của các quyết định được liên kết có thể được biểu diễn bằng sơ đồ cây. Một trong những ưu điểm của decision trees là chúng dễ xác thực và kiểm tra, không giống thuật toán Neural networks.
* Random forests: Trong một khu rừng ngẫu nhiên, thuật toán máy học dự đoán một giá trị hoặc danh mục bằng cách kết hợp các kết quả từ một số cây quyết định.

**II. Nghiên cứu cơ sở lý thuyết**

**1. Giới thiệu về Keylogger Detection**

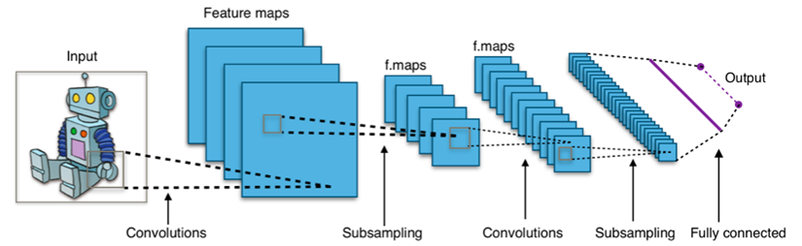
* Keylogger là một loại phần mềm gián điệp đặc biệt nguy hiểm, có thể ghi lại và đánh cắp các lần nhấn phím liên tiếp (và hơn thế nữa) mà người dùng nhập trên thiết bị. Thuật ngữ keylogger hay “trình ghi thao tác bàn phím” rất dễ hiểu: Phần mềm ghi lại những gì bạn gõ trên bàn phím. Tuy nhiên, keylogger cũng có thể cho phép tội phạm mạng nghe lén bạn, theo dõi bạn trên camera hệ thống hoặc nghe qua micrô trên điện thoại thông minh của bạn.
* Keylogger là một công cụ phổ biến của các tập đoàn, được bộ phận CNTT sử dụng để khắc phục sự cố kỹ thuật trên hệ thống và mạng của họ—hoặc để lén lút theo dõi nhân viên. Điều tương tự cũng xảy ra với các bậc cha mẹ muốn theo dõi hoạt động của con cái họ. Những người phối ngẫu đáng ngờ là một thị trường khác cho những kẻ theo dõi thao tác bàn phím.
* Mối lo ngại về keylogger là khi có những kẻ độc hại đứng đằng sau chúng. Và họ chắc chắn không sở hữu thiết bị mà họ lây nhiễm. Bạn không biết họ đã xâm nhập vào máy tính của bạn; và tùy thuộc vào loại keylogger, nó có thể đánh cắp bất kỳ mật khẩu nào bạn đã nhập, chụp ảnh màn hình định kỳ, ghi lại các trang web bạn xem, lấy trộm email đã gửi của bạn và bất kỳ phiên nhắn tin tức thời nào cũng như thông tin tài chính nhạy cảm. (chẳng hạn như số thẻ tín dụng, mã PIN và tài khoản ngân hàng), sau đó gửi tất cả dữ liệu đó qua mạng tới máy tính hoặc máy chủ web từ xa. Ở đó, người vận hành chương trình ghi nhật ký có thể lấy lại tất cả, có khả năng gửi nó cho bên thứ ba vì mục đích tội phạm.

**2. Machine Learning Pipeline**

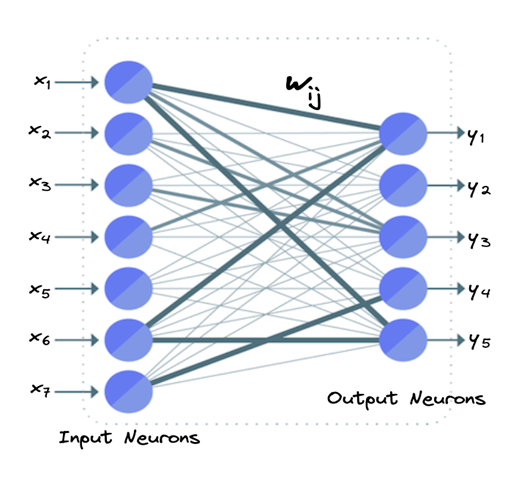
* Machine Learning Pipeline là một chuỗi các bước mô hình hóa và xử lý dữ liệu được kết nối với nhau được thiết kế để tự động hóa, tiêu chuẩn hóa và hợp lý hóa quy trình xây dựng, đào tạo, đánh giá và triển khai các mô hình học máy.
* Nó là một thành phần quan trọng trong việc phát triển và sản xuất các hệ thống máy học, giúp các nhà khoa học dữ liệu và kỹ sư dữ liệu quản lý sự phức tạp của quá trình học máy từ đầu đến cuối và giúp họ phát triển các giải pháp chính xác và có thể mở rộng cho nhiều ứng dụng.
* Machine Learning Pipeline lại nhiều lợi ích:
* Mô-đun hóa : Quy trình cho phép bạn chia nhỏ quy trình học máy thành các bước mô-đun được xác định rõ ràng. Mỗi bước có thể được phát triển, thử nghiệm và tối ưu hóa một cách độc lập, giúp quản lý và duy trì quy trình làm việc dễ dàng hơn.
* Khả năng tái tạo : Các quy trình học máy giúp tái tạo các thí nghiệm dễ dàng hơn. Bằng cách xác định trình tự các bước và tham số của chúng trong quy trình, bạn có thể tạo lại toàn bộ quy trình một cách chính xác, đảm bảo kết quả nhất quán. Nếu một bước không thành công hoặc hiệu suất của mô hình suy giảm, quy trình có thể được định cấu hình để đưa ra cảnh báo hoặc thực hiện các hành động khắc phục.
* Hiệu quả : Đường ống tự động hóa nhiều tác vụ thông thường, chẳng hạn như tiền xử lý dữ liệu, kỹ thuật tính năng và đánh giá mô hình. Hiệu quả này có thể tiết kiệm một lượng thời gian đáng kể và giảm nguy cơ xảy ra lỗi.
* Khả năng mở rộng : Đường ống có thể dễ dàng mở rộng quy mô để xử lý các tập dữ liệu lớn hoặc quy trình công việc phức tạp. Khi độ phức tạp của dữ liệu và mô hình tăng lên, bạn có thể điều chỉnh quy trình mà không cần phải cấu hình lại mọi thứ từ đầu, việc này có thể tốn thời gian.
* Thử nghiệm : Bạn có thể thử nghiệm các kỹ thuật tiền xử lý dữ liệu, lựa chọn tính năng và mô hình khác nhau bằng cách sửa đổi các bước riêng lẻ trong quy trình. Tính linh hoạt này cho phép lặp lại và tối ưu hóa nhanh chóng.
* Triển khai : Quy trình tạo điều kiện thuận lợi cho việc triển khai các mô hình học máy vào sản xuất. Sau khi thiết lập một quy trình được xác định rõ ràng để đào tạo và đánh giá mô hình, bạn có thể dễ dàng tích hợp nó vào ứng dụng hoặc hệ thống của mình.
* Cộng tác : Quy trình giúp các nhóm nhà khoa học và kỹ sư dữ liệu cộng tác dễ dàng hơn. Vì quy trình làm việc được cấu trúc và ghi lại nên các thành viên trong nhóm sẽ dễ dàng hiểu và đóng góp cho dự án hơn.
* Kiểm soát phiên bản và tài liệu : Bạn có thể sử dụng hệ thống kiểm soát phiên bản để theo dõi các thay đổi về mã và cấu hình của quy trình, đảm bảo rằng bạn có thể quay lại các phiên bản trước nếu cần. Một quy trình có cấu trúc tốt sẽ khuyến khích việc ghi chép tốt hơn về từng bước.
* Công nghệ học máy đang phát triển với tốc độ nhanh chóng, nhưng chúng ta có thể xác định một số bước tổng quát liên quan đến quá trình xây dựng và triển khai các mô hình học máy và học sâu.
* Thu thập dữ liệu : Trong giai đoạn đầu này, dữ liệu mới được thu thập từ nhiều nguồn dữ liệu khác nhau, chẳng hạn như cơ sở dữ liệu, API hoặc tệp. Việc nhập dữ liệu này thường liên quan đến dữ liệu thô có thể yêu cầu xử lý trước để có ích.
* Tiền xử lý dữ liệu : Giai đoạn này bao gồm việc làm sạch, chuyển đổi và chuẩn bị dữ liệu đầu vào cho mô hình hóa. Các bước tiền xử lý phổ biến bao gồm xử lý các giá trị bị thiếu, mã hóa các biến phân loại, chia tỷ lệ các đặc tính số và chia dữ liệu thành các tập huấn luyện và kiểm tra.
* Kỹ thuật tính năng : Kỹ thuật tính năng là quá trình tạo ra các tính năng mới hoặc chọn các tính năng có liên quan từ dữ liệu có thể cải thiện khả năng dự đoán của mô hình. Bước này thường đòi hỏi kiến ​​thức về lĩnh vực và sự sáng tạo.
* Lựa chọn mô hình : Trong giai đoạn này, bạn chọn (các) thuật toán học máy thích hợp dựa trên loại vấn đề (ví dụ: phân loại, hồi quy), đặc điểm dữ liệu và yêu cầu về hiệu suất. Bạn cũng có thể xem xét điều chỉnh siêu tham số.
* Huấn luyện mô hình : (Các) mô hình đã chọn được huấn luyện trên tập dữ liệu huấn luyện bằng (các) thuật toán đã chọn. Điều này liên quan đến việc tìm hiểu các mẫu và mối quan hệ cơ bản trong dữ liệu huấn luyện. Các mô hình được đào tạo trước cũng có thể được sử dụng thay vì đào tạo một mô hình mới.
* Đánh giá mô hình : Sau khi đào tạo, hiệu suất của mô hình được đánh giá bằng cách sử dụng tập dữ liệu thử nghiệm riêng hoặc thông qua xác thực chéo. Các số liệu đánh giá phổ biến phụ thuộc vào vấn đề cụ thể nhưng có thể bao gồm accuracy, precision, recall, F1-score, lỗi bình phương trung bình hoặc các số liệu khác.
* Triển khai mô hình : Khi một mô hình thỏa đáng được phát triển và đánh giá, nó có thể được triển khai đến môi trường sản xuất nơi nó có thể đưa ra dự đoán về dữ liệu mới, chưa được nhìn thấy. Việc triển khai có thể liên quan đến việc tạo API và tích hợp với các hệ thống khác.
* Giám sát và bảo trì : Sau khi triển khai, điều quan trọng là phải liên tục theo dõi hiệu suất của mô hình và đào tạo lại mô hình khi cần để thích ứng với các mẫu dữ liệu thay đổi. Bước này đảm bảo rằng mô hình vẫn chính xác và đáng tin cậy trong môi trường thực tế.

**3. Model CNN - Convolutional Neural Network**

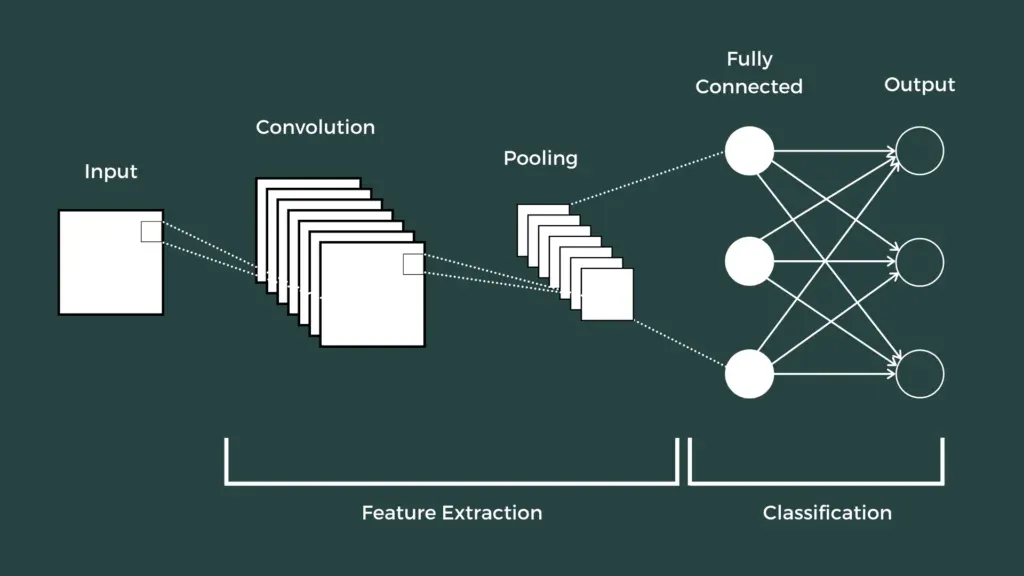
**3.1. Khảo sát cấu trúc và nguyên lý hoạt động của CNN**

CNN là một trong những mô hình học sâu tiên tiến giúp cho chúng ta xây dựng được những hệ thống xử lý thông minh, cho kết quả có độ chính xác cao. Mô hình CNN như hình 1 có các layer liên kết được với nhau thông qua cơ chế tích chập (convolution). Layer tiếp theo là kết quả tích chập từ layer trước đó. Nhờ vậy, ta có được các kết nối cục bộ. Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua tích chập (convolution) từ các bộ lọc.

● Tầng Convolution sử dụng phép tích chập để xử lý dữ liệu bằng cách trượt cửa sổ trượt (slide windows) có kích thước cố định (còn gọi là kernel) trên ma trận dữ liệu đầu vào để thu được kết quả đã được tinh chỉnh. Trong khi đó, tầng Pooling tổng hợp các vector kết quả của tầng Convolution và giữ lại những vector quan trọng nhất.

● Đầu ra cuối cùng của Lớp Pooling cuối cùng đóng vai trò là đầu vào của Lớp Fully Connected trong CNN. Một hoặc nhiều lớp có thể có trong này. Lớp Fully Connected, mọi nút trong lớp ban đầu được kết nối với mọi nút của lớp tiếp theo như minh họa dưới đây.

Một cấu trúc cơ bản nhất của CNN sẽ bao gồm 3 phần chủ yếu, đó là:

* Local receptive field (trường cục bộ): Nhiệm vụ của trường cục bộ là phân tách và lọc dữ liệu cũng như thông tin ảnh, sau đó chọn ra các vùng ảnh có giá trị sử dụng cao nhất.
* Shared weights and bias (trọng số chia sẻ): Trong mạng CNN, thành phần này có tác dụng giảm thiểu tối đa lượng tham số có tác dụng lớn. Trong mỗi convolution sẽ chứa nhiều feature map khác nhau, mỗi feature lại có khả năng giúp nhận diện một số feature trong ảnh.
* Pooling layer (lớp tổng hợp): Pooling layer là lớp cuối cùng, với khả năng đơn giản hóa thông tin đầu ra. Khi đã hoàn tất tính toán và quét qua các lớp, pooling layer sẽ được tạo ra nhằm mục đích lược bớt các thông tin không cần thiết và tối ưu đầu ra. Điều này giúp người dùng nhận được kết quả ưng ý và đúng với yêu cầu hay mong muốn.

**3.2. Ứng dụng của CNN**

Mặc dù CNN chủ yếu được sử dụng cho các vấn đề về computer vision, nhưng điều quan trọng là đề cập đến khả năng giải quyết các vấn đề học tập khác của họ, chủ yếu liên quan đến tích chuỗi dữ liệu. Ví dụ: CNN đã được biết là hoạt động tốt trên chuỗi văn bản, âm thanh và video, đôi khi kết hợp với các mạng khác quả cầu kiến trúc hoặc bằng cách chuyển đổi các chuỗi thành hình ảnh có thể được xử lý của CNN. Một số vấn đề dữ liệu cụ thể có thể được giải quyết bằng cách sử dụng CNN với chuỗi dữ liệu là các bản dịch văn bản bằng máy, xử lý ngôn ngữ tự nhiên và gắn thẻ khung video, trong số nhiều người khác.

**Classification:** Đây là nhiệm vụ được biết đến nhiều nhất trong computer vision. Ý tưởng chính là phân loại nội dung chung của hình ảnh thành một tập hợp các danh mục, được gọi là nhãn. Ví dụ: phân loại có thể xác định xem một hình ảnh có phải là của một con chó, một con mèo hay bất kỳ động vật khác.

**Localization:** Mục đích chính của localization là tạo ra một hộp giới hạn mô tả vị trí của đối tượng trong hình ảnh. Đầu ra bao gồm một nhãn lớp và một hộp giới hạn.

**Detection:** Nhiệm vụ này bao gồm thực hiện localization trên tất cả các đối tượng trong ảnh. Các đầu ra bao gồm nhiều hộp giới hạn, cũng như nhiều nhãn lớp (một cho mỗi hộp). Nhiệm vụ này được sử dụng trong việc chế tạo ô tô tự lái, với mục tiêu là có thể xác định vị trí các biển báo giao thông, đường, ô tô khác, người đi bộ và bất kỳ đối tượng nào khác có thể phù hợp để đảm bảo trải nghiệm lái xe an toàn

**Segmentation:** Nhiệm vụ ở đây là xuất ra cả nhãn lớp và đường viền của mỗi đối tượng hiện diện trong hình ảnh. Điều này chủ yếu được sử dụng để đánh dấu các đối tượng quan trọng của hình ảnh cho phân tích sâu hơn. Ví dụ: tác vụ này có thể được sử dụng để phân định rõ ràng khu vực tương ứng với khối u trong hình ảnh phổi của bệnh nhân.

**III. Thiết kế hệ thống**

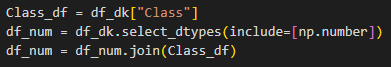
**1. Tiền xử lý dữ liệu (Data Preprocessing)**



Đọc tập dữ liệu, hiển thị 5 hàng đầu tiên của khung dữ liệu, cung cấp cái nhìn tổng quan về cấu trúc dữ liệu và nội dung.



Hiển thị thông tin về cấu trúc và kiểu dữ liệu của tệp dữ liệu đầu vào



Dòng thứ nhất tạo ra một DataFrame mới có tên Class\_df chỉ chứa cột "Class" từ DataFrame gốc df\_dk.

Dòng thứ hai tạo ra một DataFrame mới có tên df\_num chỉ chứa các cột số từ df\_dk. Nó sử dụng phương thức select\_dtypes để lọc các cột dựa trên kiểu dữ liệu của chúng, chỉ giữ lại các cột có giá trị số (int64, float64, v.v.).

Dòng thứ ba nối Class\_df (chứa nhãn lớp) trở lại DataFrame df\_num (chứa các cột số). Nó thực sự hợp nhất hai DataFrame, kết hợp các tính năng số với nhãn lớp tương ứng của chúng.



Loại bỏ cột chỉ số khỏi tập dữ liệu

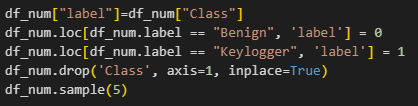


Dòng đầu tiên định lượng mức độ dữ liệu thiếu trong DataFrame, cung cấp những hiểu biết có giá trị về chất lượng dữ liệu và nhu cầu làm sạch dữ liệu tiềm ẩn.

Dòng thứ hai loại bỏ các hàng có giá trị thiếu, nhằm tạo ra một DataFrame có thông tin đầy đủ cho phân tích hoặc mô hình hóa tiếp theo.



Dòng này đếm số lần xuất hiện của mỗi giá trị duy nhất trong cột "Class"



Tạo một cột mới có tên "label" trong DataFrame df\_num, sao chép các giá trị từ cột "Class" hiện có vào cột mới này.

Thay thế giá trị "Benign" trong cột "label" bằng 0

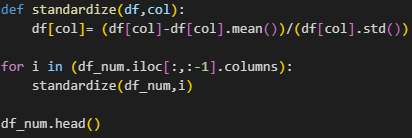
Thay thế giá trị "Keylogger" trong cột "label" bằng 1

Xóa cột "Class" gốc khỏi DataFrame, vì thông tin của nó hiện được lưu trữ trong cột "label" có tính số.

Chọn ngẫu nhiên và hiển thị 5 hàng từ DataFrame df\_num đã sửa đổi.



Duyệt qua tất cả các cột trong df\_num và xóa các cột chỉ chứa một giá trị duy nhất.



Chuẩn hóa dữ liệu là một kỹ thuật tiền xử lý dữ liệu phổ biến trong học máy, bao gồm cả deep learning. Kỹ thuật này nhằm mục đích đưa dữ liệu về một dạng thống nhất, giúp cho các mô hình học máy có thể học tập hiệu quả hơn

Thực hiện chuẩn hóa trên cột được chỉ định:

* Trừ đi trung bình của cột cho mỗi giá trị.
* Chia kết quả cho độ lệch chuẩn của cột.
* Quá trình này tập trung dữ liệu xung quanh 0 và chia tỷ lệ nó để có độ lệch chuẩn là 1.

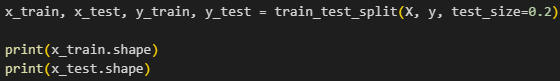


Tách mục tiêu và tính năng: Mã tách các cột của DataFrame thành hai tập hợp riêng biệt:

* target: Cột chứa biến cần dự đoán (kết quả).
* features: Các cột chứa các biến đầu vào được sử dụng để dự đoán.

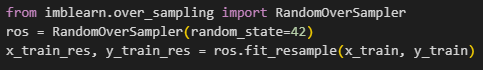
**2. Data Validation**

Chia tệp dữ liệu

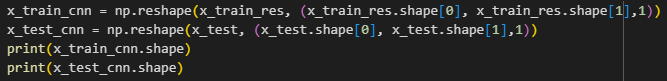


Bộ dữ liệu sẽ được chia thành hai phần: 80% sẽ được sử dụng để huấn luyện mô hình (training set), và 20% phần còn lại sẽ được sử dụng để kiểm thử mô hình (test set).

Cân bằng dữ liệu

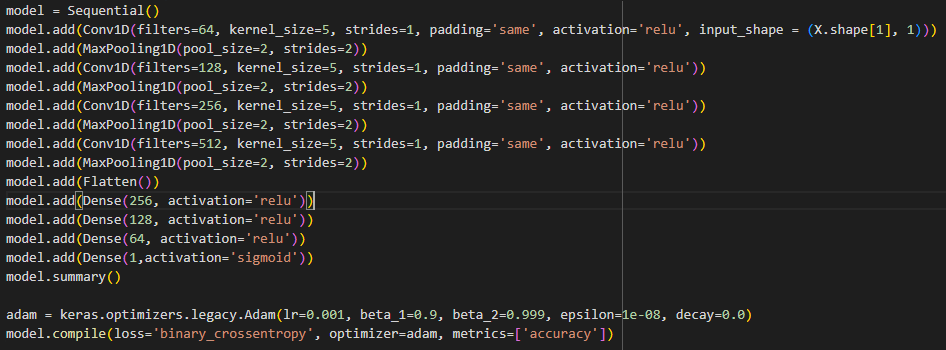


Đoạn code trên là một cách để giải quyết vấn đề mất cân bằng lớp trong tập dữ liệu. Nó sử dụng kỹ thuật oversampling ngẫu nhiên để tạo các mẫu tổng hợp cho lớp thiểu số, nhằm cân bằng phân phối lớp.



Định dạng dữ liệu đầu vào cho mô hình CNN

**3. Xây dựng mô hình CNN**



* model = Sequential()

Tạo một mô hình tuần tự, một ngăn xếp tuyến tính của các lớp.

* Các lớp tích chập:

model.add(Conv1D(filters=64, kernel\_size=5, strides=1, padding='same', activation='relu', input\_shape=(X.shape[1], 1)))

* filters=64: Số lượng bộ lọc (máy dò tính năng) cần học.
* kernel\_size=5: Kích thước của hạt tích chập (cửa sổ) để trích xuất tính năng.
* strides=1: Kích thước bước mà hạt di chuyển trên đầu vào.
* padding='same': Đệm đầu vào để bảo toàn các chiều không gian sau khi tích chập.
* activation='relu': Áp dụng hàm kích hoạt ReLU để đưa ra phi tuyến tính.
* input\_shape=(X.shape[1], 1): Xác định hình dạng đầu vào mong đợi:
* X.shape[1]: Số lượng tính năng (cột) trong dữ liệu đầu vào.
* 1: Một kênh (ví dụ: cho chuỗi thời gian một chiều).

model.add(MaxPooling1D(pool\_size=2, strides=2)) thêm một lớp phân vùng tối đa 1D vào mô hình mạng nơ-ron. Lớp này sẽ giảm kích thước đầu vào đi 1 nửa

* pool\_size=2: Kích thước của cửa sổ phân vùng (vùng phụ để giảm kích thước).
* strides=2: Kích thước bước mà cửa sổ di chuyển trên đầu vào.

Hàm kích hoạt ReLU (Rectified Linear Unit) là một hàm phi tuyến được sử dụng trong các mạng nơ-ron nhân tạo (neural network). Hàm này có dạng:

f(x) = max(0, x)

Hàm ReLU được sử dụng để chuyển đổi đầu vào của một mạng nơ-ron thành một giá trị phi tuyến. Giá trị đầu ra của hàm ReLU sẽ là 0 nếu đầu vào nhỏ hơn hoặc bằng 0, và bằng chính đầu vào nếu đầu vào lớn hơn 0.

Các khối Conv1D và MaxPooling1D tương tự được lặp lại ba lần nữa:

Tăng số lượng bộ lọc (128, 256, 512) trong các lớp tích chập tiếp theo.

Các lớp MaxPooling1D giảm chiều và nắm bắt các tính năng cần thiết.

* Làm phẳng phẳng:

model.add(Flatten())

Làm phẳng đầu ra 3D từ các lớp tích chập thành một vector 1D cho các lớp dày đặc.

* Các lớp dày đặc:

model.add(Dense(256, activation='relu'))

Lớp dày đặc đầu tiên với 256 nơron và kích hoạt ReLU.

Các lớp dày đặc tương tự với 128 và 64 nơron:

Học các mối quan hệ phi tuyến phức tạp hơn giữa các tính năng.

* Lớp đầu ra:

model.add(Dense(1, activation='sigmoid'))

Lớp dày đặc cuối cùng với một nơron duy nhất và kích hoạt sigmoid, tạo ra đầu ra xác suất cho phân loại nhị phân.

Hàm kích hoạt sigmoid là một hàm phi tuyến được sử dụng trong các mạng nơ-ron nhân tạo (neural network). Hàm này có dạng:

f(x) = 1 / (1 + exp(-x))

Hàm sigmoid có một số tác dụng sau:

Tạo ra các giá trị đầu ra trong khoảng [0, 1]: Hàm sigmoid có dạng là một đường cong hình chữ S, với giá trị đầu ra nằm trong khoảng từ 0 đến 1. Điều này rất hữu ích cho các bài toán phân loại, trong đó các kết quả cần được biểu diễn dưới dạng xác suất.

Làm giảm hiện tượng vanishing gradient: Hàm sigmoid có đạo hàm không bằng 0 tại mọi điểm, do đó nó có thể giúp giảm hiện tượng vanishing gradient, một hiện tượng có thể làm giảm hiệu quả của quá trình học.

* model.summary()

In ra tóm tắt kiến trúc của mô hình, bao gồm loại lớp, hình dạng đầu ra và số lượng tham số.

* Dòng mã adam = keras.optimizers.legacy.Adam(lr=0.001, beta\_1=0.9, beta\_2=0.999, epsilon=1e-08, decay=0.0) tạo một đối tượng tối ưu hóa Adam với các tham số sau:
  + lr=0.001: Tốc độ học ban đầu, điều chỉnh mức độ cập nhật trọng số của mô hình trong mỗi bước đào tạo.
  + beta\_1=0.9: Tỷ lệ phân rã mũ cho ước tính momen đầu tiên.
  + beta\_2=0.999: Tỷ lệ phân rã mũ cho ước tính momen thứ hai.
  + epsilon=1e-08: Hằng số nhỏ cho ổn định số.

Tốc độ học ban đầu là một tham số quan trọng cần điều chỉnh cho từng nhiệm vụ cụ thể. Một tốc độ học quá cao có thể khiến mô hình bị quá khớp, trong khi một tốc độ học quá thấp có thể khiến mô hình mất ổn định và không hội tụ.

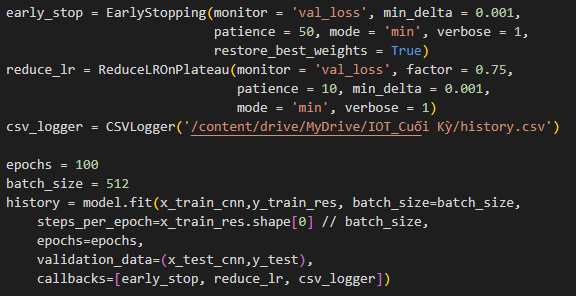
Các tham số beta\_1 và beta\_2 xác định cách Adam điều chỉnh tốc độ học cho các tham số riêng lẻ. beta\_1 kiểm soát mức độ ảnh hưởng của các giá trị ước tính gần đây đối với tốc độ học, trong khi beta\_2 kiểm soát mức độ ảnh hưởng của các giá trị ước tính cũ đối với tốc độ học.

* model.compile(loss='binary\_crossentropy', optimizer=adam, metrics=['accuracy'])

loss='binary\_crossentropy': Hàm mất mát phù hợp cho các nhiệm vụ phân loại nhị phân.

optimizer=adam: Chỉ định trình tối ưu hóa Adam sẽ được sử dụng cho đào tạo.

metrics=['accuracy']: Theo dõi chỉ số độ chính xác trong quá trình đào tạo và đánh giá.



epochs = 100: Thiết lập số lần mô hình sẽ lặp lại toàn bộ tập dữ liệu đào tạo. Nhiều epochs thường dẫn đến học tập tốt hơn, nhưng có thể quá khớp.

batch\_size = 512: Xác định số mẫu đào tạo được xử lý trước khi cập nhật trọng số mô hình. Các lô lớn có thể nhanh hơn nhưng kém ổn định hơn; các lô nhỏ có thể hội tụ tốt hơn nhưng chậm hơn.

history = model.fit(...): Bắt đầu đào tạo mô hình và lưu lịch sử đào tạo trong history. Các đối số chính:

* x\_train\_cnn: Các tính năng đào tạo (dữ liệu đầu vào).
* y\_train\_res: Nhãn đào tạo (sự thật cơ bản).
* batch\_size: Như đã định nghĩa ở trên.
* steps\_per\_epoch: Số lô mỗi epoch, được tính là x\_train\_res.shape[0] // batch\_size.
* epochs: Như đã định nghĩa ở trên.
* validation\_data=(x\_test\_cnn,y\_test): Dữ liệu để đánh giá hiệu suất mô hình trong quá trình đào tạo (không được sử dụng để cập nhật trọng số).
* callbacks=[early\_stop, reduce\_lr, csv\_logger]: Các hàm được gọi tại các điểm cụ thể trong đào
* early\_stop: Dừng đào tạo sớm nếu hiệu suất xác thực không cải thiện trong một khoảng thời gian nhất định, ngăn ngừa quá khớp.
* monitor='val\_loss': Chỉ định rằng chỉ số cần theo dõi để dừng sớm là mất mát xác thực.
* min\_delta=0.001: Chỉ định rằng chỉ số cần cải thiện ít nhất 0,001 để được coi là cải thiện.
* patience=50: Chỉ định rằng đào tạo sẽ tiếp tục trong tối đa 50 epoch mà không có cải thiện trước khi dừng.
* mode='min': Chỉ định rằng đào tạo nên dừng khi chỉ số mất mát xác thực ngừng giảm.
* verbose=1: Chỉ định rằng một thông báo sẽ được in ra khi xảy ra dừng sớm.
* restore\_best\_weights=True: Chỉ định rằng trọng số của mô hình sẽ được khôi phục từ epoch có giá trị mất mát xác thực tốt nhất.

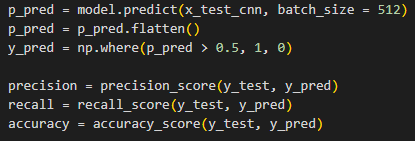
Nếu chỉ số mất mát xác thực không cải thiện trong 50 epoch liên tiếp, đào tạo sẽ dừng lại và trọng số của mô hình từ epoch có giá trị mất mát xác thực tốt nhất sẽ được khôi phục. Điều này giúp ngăn ngừa mô hình bị mắc kẹt trong trạng thái quá khớp, trong đó mô hình học các mẫu đào tạo một cách quá chính xác, nhưng lại không thể dự đoán tốt các mẫu mới.

* reduce\_lr: Giảm tốc độ học nếu hiệu suất xác thực đạt đỉnh, có thể giúp mô hình hội tụ.
* monitor='val\_loss': Chỉ định rằng chỉ số cần theo dõi để giảm tốc độ học là mất mát xác thực.
* factor=0.75: Chỉ định rằng tốc độ học sẽ được giảm xuống 75% của giá trị hiện tại khi phát hiện ra một đỉnh.
* patience=10: Chỉ định rằng đào tạo sẽ tiếp tục trong tối đa 10 epoch mà không có cải thiện trước khi giảm tốc độ học.
* min\_delta=0.001: Chỉ định rằng chỉ số cần cải thiện ít nhất 0,001 để được coi là cải thiện.
* mode='min': Chỉ định rằng tốc độ học nên được giảm khi chỉ số mất mát xác thực ngừng giảm.
* verbose=1: Chỉ định rằng một thông báo sẽ được in ra khi tốc độ học được giảm.

Nếu chỉ số mất mát xác thực không cải thiện trong 10 epoch liên tiếp, tốc độ học sẽ được giảm xuống 75% của giá trị hiện tại. Điều này có thể giúp mô hình thoát khỏi các điểm cực tiểu cục bộ và tiếp tục học hỏi.

* csv\_logger: Lưu các chỉ số đào tạo vào tệp CSV để phân tích.

**4. Evaluation**



p\_pred = model.predict(x\_test\_cnn, batch\_size=512): sử dụng mô hình đã đào tạo để tạo ra các dự đoán xác suất cho dữ liệu thử nghiệm. Dữ liệu được xử lý theo lô 512 để cải thiện hiệu suất.

p\_pred = p\_pred.flatten(): Định hình lại các dự đoán thành mảng một chiều để so sánh dễ dàng hơn với nhãn thực.

y\_pred = np.where(p\_pred > 0.5, 1, 0): sử dụng ngưỡng 0,5 để chuyển đổi các dự đoán xác suất thành dự đoán lớp nhị phân. Giá trị dự đoán lớn hơn hoặc bằng 0,5 được gán là 1, còn giá trị dự đoán nhỏ hơn 0,5 được gán là 0.

