BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

A picture containing text, sign

Description automatically generated

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**NGHIÊN CỨU MẠNG CNN ĐỂ GIẢI BÀI TOÁN TỐI ƯU HOÁ TỔ HỢP**

Sinh viên thực hiện:

**1. Đỗ Viết Thịnh** Lớp:DHCNTT03 – K14

2. Nguyễn Hoàng Minh Lớp: DHCNTT06 – K15

3. Hoàng Thị Sao Mai Lớp: DHCNTT06 – K15

4. Nguyễn Thị Thu Thuỷ Lớp: DHCNTT06 – K15

5. Mai Ngọc Định Lớp: DHCNTT05 – K14

Khoa: Công nghệ thông tin

Người hướng dẫn: **TS. Nguyễn Thị Mỹ Bình**

**Hà Nội, 05/2023**

BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

A picture containing text, sign

Description automatically generated

**BÁO CÁO TỔNG KẾT**

**ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN**

**NGHIÊN CỨU MẠNG CNN ĐỂ GIẢI BÀI TOÁN TỐI ƯU HOÁ TỔ HỢP**

Sinh viên thực hiện:

**1. Đỗ Viết Thịnh**  Giới tính: Nam

2. Nguyễn Hoàng Minh Nam

3. Hoàng Thị Sao Mai Nữ

4. Nguyễn Thị Thu Thuỷ Nữ

5. Mai Ngọc Định Nam

Lớp: DHCNTT03 – K14 Năm thứ: 4/4

DHCNTT06 – K15 3/4

DHCNTT06 – K15 3/4

DHCNTT06 – K15 3/4

DHCNTT05 – K14 4/4

Dân tộc: Kinh

Khoa: Công nghệ thông tin

Người hướng dẫn: **TS. Nguyễn Thị Mỹ Bình**

**Hà Nội, 05/2023**

LỜI CẢM ƠN

Đầu tiên, chúng em xin gửi lời cảm ơn chân thành và sâu sắc tới giáo viên hướng dẫn là **TS. Nguyễn Thị Mỹ Bình** đã tận tình chỉ bảo, hướng dẫn và đồng thời mang đến cho chúng em những tài liệu, kiến thức quý giá để giúp chúng em có thể tìm hiểu, trau dồi, nâng cao kiến thức và hoàn thành đề tài nghiên cứu này.

Chúng em xin gửi lời cảm ơn đến Trường Đại học Công nghiệp Hà Nội cũng như khoa Công nghệ thông tin đã tạo điều kiện cho chúng em được tham gia làm nghiên cứu khoa học. Chúng em đã có điều kiện tìm hiểu với các nội dung mới lạ đòi hỏi nhiều kiến thức cao hơn. Bên cạnh việc tạo ra cơ hội cho chúng em tiếp xúc các vấn đề mới thì Nhà trường cùng khoa cũng tạo điều kiện cho chúng em được nghiên cứu tại phòng Lab Optimization, Modeling and Simulation. Phòng Lab với điều kiện vật chất hiện đại, tiện nghi đã giúp chúng em có một tinh thần thoải mái khi tiếp xúc và nghiên cứu một đề tài mới và khó.

***Chúng em xin chân thành cảm ơn!***

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc135525628)

[DANH MỤC THUẬT NGỮ, CÁC TỪ VIẾT TẮT VÀ KÝ HIỆU i](#_Toc135525629)

[DANH MỤC BẢNG BIỂU, BIỂU ĐỒ ii](#_Toc135525630)

[DANH MỤC HÌNH ẢNH iii](#_Toc135525631)

[MỞ ĐẦU 1](#_Toc135525632)

[1.Lý do chọn đề tài 1](#_Toc135525633)

[2.Mục tiêu của đề tài 1](#_Toc135525634)

[3.Đối tượng nghiên cứu 1](#_Toc135525635)

[4.Nội dung nghiên cứu 2](#_Toc135525636)

[5.Cấu trúc của bài báo cáo 2](#_Toc135525637)

[CHƯƠNG 1.CƠ SỞ LÝ THUYẾT 3](#_Toc135525638)

[1.1 Bài toán tối ưu hoá tổ hợp 3](#_Toc135525639)

[1.1.1 Tổng quan về bài toán tối ưu 3](#_Toc135525640)

[1.1.1.1. Lịch sử bài toán tối ưu. 3](#_Toc135525641)

[1.1.1.2. Định nghĩa 3](#_Toc135525642)

[1.1.1.3. Phân loại bài toán tối ưu 4](#_Toc135525643)

[1.2 Convolution Neural Networks 5](#_Toc135525644)

[1.2.1 Các lớp cơ bản của một Convolution Neural Networks. 5](#_Toc135525645)

[1.2.1.1. Convolution Layer. 5](#_Toc135525646)

[1.2.1.2. Lớp Pooling. 7](#_Toc135525647)

[1.2.1.3. Lớp Fully Connected 8](#_Toc135525648)

[1.2.2 Cách Convolutional Neural Network hoạt động. 9](#_Toc135525649)

[1.2.2.1. Khởi tạo Weight. 9](#_Toc135525650)

[1.2.2.2. Quy định network. 10](#_Toc135525651)

[1.2.3 Bộ phân loại ConvNet. 10](#_Toc135525652)

[1.2.3.1. Nhận dạng khuôn mặt dựa trên CNN (Face recognition). 11](#_Toc135525653)

[1.2.3.2. Nhận dạng cử chỉ tay dựa trên CNN (Hand gesture recognition). 11](#_Toc135525654)

[1.2.4 Một số mạng CNN nổi tiếng. 11](#_Toc135525655)

[1.2.4.1. VGG 16 11](#_Toc135525656)

[1.2.4.2. ResNet 12](#_Toc135525657)

[1.2.5 Sử dụng CNN vào bài toán nhận đối tượng. 15](#_Toc135525658)

[1.2.6 Tổng quan YOLO 16](#_Toc135525659)

[1.2.6.1. YOLOv5 17](#_Toc135525660)

[1.2.6.2. YOLOv8 18](#_Toc135525661)

[1.2.7 Đo lường đối tượng thời gian thực 20](#_Toc135525662)

[CHƯƠNG 2.NGHIÊN CỨU MẠNG CNN ĐỂ GIẢI BÀI TOÁN TỐI ƯU TỔ HỢP “NHẬN DIỆN RĂNG TRONG CHỤP X-QUANG” 21](#_Toc135525663)

[2.1 Giải thuật 21](#_Toc135525664)

[2.1.1 Mô tả bài toán 21](#_Toc135525665)

[2.1.2 Mô tả dữ liệu dataset 21](#_Toc135525666)

[2.1.3 Thuật toán tối ưu 22](#_Toc135525667)

[2.1.3.1. Thuật toán Transfer learning 22](#_Toc135525668)

[2.1.3.2. Thuật toán K-mean 24](#_Toc135525669)

[2.2 Kết quả chạy các bộ dữ liệu 26](#_Toc135525670)

[2.2.1 Môi trường thực nghiệm 26](#_Toc135525671)

[2.2.2 Dữ liệu thực nghiệm 26](#_Toc135525672)

[2.2.3 Kết quả chạy các bộ dữ liệu chuẩn 27](#_Toc135525673)

[2.3 Đánh giá giả thuật và dự đoán hướng đi trong tương lai 29](#_Toc135525674)

[KẾT LUẬN VÀ KIẾN NGHỊ 30](#_Toc135525675)

[Kết quả đạt được 30](#_Toc135525676)

[Hướng phát triển trong tương lai 31](#_Toc135525677)

[TÀI LIỆU THAM KHẢO 32](#_Toc135525678)

DANH MỤC THUẬT NGỮ, CÁC TỪ VIẾT TẮT VÀ KÝ HIỆU

|  |  |  |
| --- | --- | --- |
| **STT** | **Các từ viết tắt, ký hiệu** | **Dịch nghĩa** |
| 1 | ANN | Artificial Neural Network |
| 2 | RNN | Recurrent Neural Network |
| 3 | CNN | Convolutional Neural Network |
| 4 | ReLU | Rectified Linear Unit |
| 5 | LSTM | Long Short-Term Memory |
| 6 | R-CNN | Region-based Convolutional Neural Network |
| 7 | SSD | Single Shot MultiBox Detector |
| 8 | YOLO | You Only Look Once |
| 9 | IOU | Intersection over Union |
| 10 | NMS | Non-Maximum Suppression |
| 11 | FLOPs | Floating Point Operations |

DANH MỤC BẢNG BIỂU, BIỂU ĐỒ

[Bảng 2.1. Kết quả huấn luyện và các thông số liên quan 28](#_Toc135514517)

DANH MỤC HÌNH ẢNH

[Hình 1.1. Mô tả về ma trận 5](#_Toc135514507)

[Hình 1.2. Kernel 6](#_Toc135514508)

[Hình 1.3. Ví dụ về các lớp convolution 7](#_Toc135514509)

[Hình 1.4. Pooling operation 8](#_Toc135514510)

[Hình 1.5. Fully connected layer 9](#_Toc135514511)

[Hình 1.6. Minh họa kiến trúc của VGG 16 12](#_Toc135514512)

[Hình 1.7. Mô tả kiến trúc cơ bản của ResNet 14](#_Toc135514513)

[Hình 1.8. Hình ảnh Vanishing Gradients 15](#_Toc135514514)

[Hình 2.1. Ảnh chụp X-Quang răng 22](#_Toc135514515)

[Hình 2.2. Hình ảnh mô tả bộ dữ liệu thực nghiệm 27](#_Toc135514516)

MỞ ĐẦU

1. Lý do chọn đề tài

Trong thập kỷ vừa qua, trí tuệ nhân tạo nói chung hay kỹ thuật học sâu nói riêng đã có những bước phát triển đáng chú ý, các kỹ thuật về mạng noron đã được nghiên cứu và phát triển rất rộng rãi. Ứng dụng của CNN vào các vấn đề như nhận diện đối tượng, phân loại đối tượng là rất tốt. Bên cạnh đó, bài toán tối ưu tổ hợp đã được nghiên cứu rất sớm, từ thời Euler, luôn là một trong những mục tiêu nghiên cứu trọng điểm, ngày nay cùng với sự phát triển nhanh chóng của CNTT, chúng đang được nhiều người quan tâm nghiên cứu và ứng dụng rộng rãi trong các bài toán thực tế. Và nhất là hiện nay tính thẩm mỹ càng trở nên phổ biến và quan trọng nhất là với bộ răng để có nụ cười đẹp. Vì vậy, xuất phát từ nhu cầu thực tiễn, chúng em chọn đề tài là: ***“Ứng dụng mạng CNN vào bài toán tối ưu hoá tổ hợp”.***

1. Mục tiêu của đề tài

Nghiên cứu, áp dụng và đánh giá giải thuật mạng CNN để giải quyết bài toán tối ưu hoá tổ hợp, cụ thể xét bài toán nhận diện răng trong chụp X-Quang.

Đề tài tập trung cụ thể vào:

* Tìm hiểu về bài toán nhận diện răng trong chụp X-Quang.
* Nghiên cứu tổng quan về mạng CNN.
* Nghiên cứu ngôn ngữ Python, cài đặt và đánh giá hiệu quả của thuật toán tối ưu hóa tổ hợp trong giải quyết bài toán nhận diện răng trong chụp X-Quang.

1. Đối tượng nghiên cứu

* Nghiên cứu các thuật toán của lý thuyết độ thị giải bài toán nhận diện răng trong chụp X-Quang toàn cảnh nha khoa đã công bố ở trong và ngoài nước.
* Nghiên cứu các kĩ thuật R-CNN.
* Nghiên cứu tài liệu về ngôn ngữ Python để cài đặt các thuật toán thử nghiệm.
* Cài đặt và đánh giá hiệu quả của thuật toán tiến hóa đa nhiệm vụ cho bài toán nhận diện.

1. Nội dung nghiên cứu

* Nghiên cứu tổng quan các nghiên cứu giải bài toán nhận diện và các hướng tiếp cận giải quyết bài toán này.
* Nghiên cứu thuật toán R-CNN.
* Nghiên cứu ngôn ngữ Python để cài đặt thuật toán.
* Lập trình và đánh giá hiệu quả của thuật toán tiến hóa đa nhiệm vụ cho bài toán nhận diện.
* Viết báo cáo nghiên cứu.

1. Cấu trúc của bài báo cáo

Cấu trúc các Chương bao gồm:

* *Chương 1:* Trình bày các cơ sở lý thuyết về bài toán tối ưu hoá tổ hợp, chi tiết hơn là nội dung bài toán Nhận diện răng trong chụp X-Quang gồm: phát biểu bài toán, các dữ liệu hình ảnh, lý do sử dụng tối ưu hoá tổ hợp để giải quyết bài toán.
* *Chương 2:* Trình bày về áp dụng giải thuật tiến hóa đa nhiệm vụ để giải bài toán người du lịch, kết quả chạy chương trình với các bộ cơ sở dữ liệu khác nhau.
* Cuối cùng chúng em sẽ đưa ra kết luận và đề cập hướng phát triển của đề tài.

# CƠ SỞ LÝ THUYẾT

* 1. Bài toán tối ưu hoá tổ hợp
     1. Tổng quan về bài toán tối ưu
        1. **Lịch sử bài toán tối ưu.**

Vào thế kỷ XVIII, một hướng nghiên cứu bài toán cực trị hàm mục tiêu xuất hiện là phiếm hàm tích phân và được gọi là phép tính biến phân.

Những năm 30-40 của thế kỷ XX xuất hiện lý thuyết Quy hoạch tuyến tính.

Những năm 50- thế kỷ XX xuất hiện Quy hoạch lồi.

Từ những những năm 70 của thế kỷ XX hình thành nhiều hướng nghiên cứu khác nhau như tối ưu không lồi, tối ưu phi tuyến, tối ưu rời rạc, tối ưu tổ hợp và tối ưu đa mục tiêu.

Từ những năm 50-60 của thế kỷ XX xuất hiện lý thuyết điều khiển được và điều khiển tối ưu.

* + - 1. **Định nghĩa**

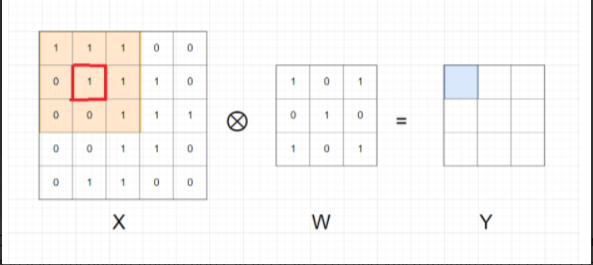
Một cách tổng quát, mỗi bài toán TWTH có thể phát biểu như sau: Cho một bộ ba (𝑆, 𝑓, Ω), trong đó S là tập hữu hạn trạng thái (lời giải tiềm năng hay phương án), f là hàm mục tiêu xác định trên S, còn Ω là tập các ràng buộc. Mỗi phương án s ∈ S thỏa mãn các ràng buộc Ω gọi là phương án (hay lời giải) chấp nhận được. Mục đích của ta là tìm phương án chấp nhận được s ∗ tối ưu hóa toàn cục hàm mục tiêu f. Chẳng hạn với bài toán cực tiểu thì f(s ∗) ≤ f(s) với mọi phương án chấp nhận được.

Trong thực tế xảy ra rất nhiều tình huống và sẽ có rất nhiều cách để chúng ta giải quyết. Do đó yêu cầu chúng ta phải lựa chọn sao cho chọn ra được cách giải quyết tốt nhất. Tuy nhiên quan điểm “tốt nhất” này phụ thuộc vào từng tình huống và từng mục đích, có thể là tối đa (maximum) cũng có thể là tối thiểu (minimum) một tiêu chí nào đó. Ví dụ như để làm ra một món ăn, có người chọn tối ưu hóa về mặt thời gian nghĩa là sẽ làm trong thời gian ngắn nhất. Do đó món ăn có thể sẽ không là ngon nhất. Nhưng cũng có người chọn làm ra món ăn ngon nhất thì lại tốn nhiều thời gian hơn.

Việc tìm phương án tốt nhất trong số những phương án có thể như trên chính là việc giải quyết bài toán tối ưu (optimization problem) trong lĩnh vực toán và công nghệ thông tin. Nhìn chung, các bài toán tối ưu luôn có hai đặc điểm sau:

* Số lời giải nhiều hơn một. Tập các lời giải này được gọi là không gian lời giải hay không gian tìm kiếm.
* Mọi lời giải của một bài toán đều được định lượng độ “tối ưu” bởi cùng một hàm theo tiêu chí của bài toán đó.
  + - 1. **Phân loại bài toán tối ưu**
* Quy hoạch tuyến tính: Hàm mục tiêu và các hàm ràng buộc đều là các hàm tuyến tính. Như vậy miền chấp nhận được là một tập lồi đa diện.
* Quy hoạch phi tuyến (Tối ưu phi tuyến): Tối thiểu có hàm mục tiêu hoặc hàm ràng buộc là phi tuyến. Tối ưu phi tuyến bao gồm: Tối ưu trơn (hàm mục tiêu và ràng buộc là trơn), Tối ưu lồi (hàm mục tiêu và ràng buộc là lồi), Tối ưu không lồi (hàm mục tiêu hoặc miền chấp nhận được không lồi).
* Tối ưu rời rạc hay tối ưu tổ hợp: Miền chấp nhận được là một tập rời rạc. Trường hợp các biến số nhận giá trị nguyên là bài toán quy hoạch nguyên.
* Tối ưu đa mục tiêu: Mục tiêu gồm nhiều hàm không hòa hợp nhau. Tối ưu đa mục tiêu cũng được phân chia thành nhiều bài toán con khác nhau tùy theo tính chất của hàm mục tiêu và tập ràng buộc.
* Quy hoạch ngẫu nhiên: Tức là bài toán tối ưu mà các tham số trong đó không có giá trị xác định mà được mô tả bởi tham số xác suất.
* Quy hoạch động: Tức là bài toán tối ưu mà các đối tượng được xét có thể chia ra nhiều giai đoạn hoặc quá trình phát triển theo thời gian. Ngoài ra còn nhiều bài toán tối ưu hóa khác như: Quy hoạch Lípshitz, quy hoạch nón, tối ưu không trơn . . .
  1. Convolution Neural Networks
     1. **Các lớp cơ bản của một Convolution Neural Networks.**
        1. **Convolution Layer.**
           1. **Phép tính convolution.**

Ta giả sử ma trận cần tình convolution là ma trận X có kích thước n\*m. Và 1 ma trận k có kích thước là x\*x. Kí hiệu phép tính convolution (⊗), kí hiệu Y = X ⊗W Với mỗi phần tử xi j trong ma trận X lấy ra một ma trận có kích thước bằng kích thước của kernel W có phần tử xij làm trung tâm (đây là vì sao kích thước của kernel thường lẻ) gọi là ma trận A. Sau đó tính tổng các phần tử của phép tính element-wise của ma trận A và ma trận W, rồi viết vào ma trận kết quả Y (như hình 1.1).



Hình 1.1. Mô tả về ma trận

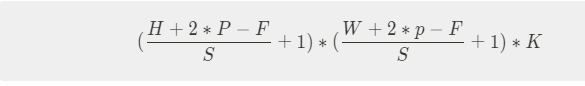
Dễ thấy với phép tính như trên thì shape của ma trân Y sẽ nhỏ hơn shape của ma trận X đầu vào. Với những trường hợp cần ma trận Y có cùng kích thước với ma trận X chúng ta thêm 1 hệ số được gọi là padding vào ma trận X rồi thực hiện phép tính convolution như bình thường.

* + - * 1. **Lớp tích chập (Convolution)**

Tích chập là lớp đầu tiên để trích xuất các tính năng từ hình ảnh đầu vào. Tích chập duy trì mối quan hệ giữa các pixel bằng cách tìm hiểu các tính năng hình ảnh bằng cách sử dụng các ô vuông nhỏ của dữ liệu đầu vào. Nó là 1 phép toán có 2 đầu vào như ma trận hình ảnh và 1 bộ lọc hoặc hạt nhân.

Giả sử input của 1 convolutional layer tổng quát là tensor kích thước H \* W \* D. Kernel có kích thước F \* F \* D (kernel luôn có depth bằng depth của input và F thường là số lẻ vì ô vuông lưới chẵn \* chẵn thì sẽ không có 1 ô vuông ở tâm đối xứng -> giảm độ chính xác), stride: S, padding: P. Convolutional layer áp dụng K kernel.

**Output có kích thước:**



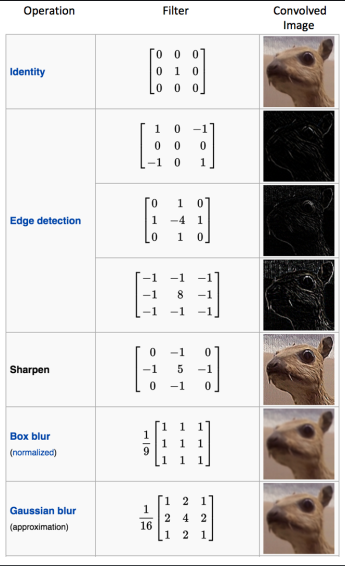
Hình 1.2. Kernel

**Lưu ý**: Ta có thể trồng nhiều lớp convolution lên nhau để lấy được đặc trưng của ảnh. Trước khi output của lớp convolution trước làm input của lớp sau thì ta đưa qua 1 hàm phi tuyến tính.

* + - * 1. **Ý nghĩa của lớp Convolution**

Convolution sẽ giúp làm mờ, làm nét ảnh. Lấy được các đặc trưng của ảnh. Mỗi kernel khác nhau sẽ đều có những tác dụng khác nhau (Hình 1.3).

**Ví dụ:**



Hình 1.3. Ví dụ về các lớp convolution

* + - 1. **Lớp Pooling.**

Giống như convolution layer, dữ liệu đầu vào được convoluted với một bộ lọc để tạo thành một ma trận convolution.

Ví dụ trong một Pooling layer một cụm 3x3 được lấy từ dữ liệu đầu và 5x5. Điều này sẽ được miêu tả ở Hình 2.3, có hai hàm pooling, average hoặc max còn tùy thuộc vào hàm. Giá trị trung bình của tất cả các phần tử sau đó được đặt vào C11 hoặc giá trị tối đa của cụm được đặt vào C11. Quá trình này được lặp đi lặp lại tùy thuộc vào chức năng, từ phần tử C11 cho đến phần tử C33 cho pooling 3x3.

Diagram

Description automatically generated

Hình 1.4. Pooling operation

Pooling layer thường được dùng giữa các convolutional layer, để giảm kích thước dữ liệu nhưng vẫn giữ được các thuộc tính quan trọng. Việc giảm kích thước dữ liệu giúp giảm các phép tính toán trong model. Bên cạnh đó, với phép pooling kích thước ảnh giảm, do đó lớp convolution học được các vùng có kích thước lớn hơn.

* + - 1. **Lớp Fully Connected**

Bằng nhiều convolution và pooling layer, đầu ra được nhận đến một fully connected layer dưới dạng đầu vào như trong Hình 2.4. Một fully connected layer thường được đặt ở phần cuối trong cấu trúc CNN, hoặc ở đâu đó trung gian. Loại layer này rất giống với đa lớp truyền tiếp perceptron (feedforward multilayer perceptron). Trong layer này, tất cả thông tin được lọc, lấy mẫu được thu thập để CNN bắt đầu học. Hầu hết các trọng số cũng nằm trong phần này của mạng.

Diagram

Description automatically generated

Hình 1.5. Fully connected layer

* + 1. **Cách Convolutional Neural Network hoạt động.**

Đối với bất kì nhiệm vụ nào được yêu cầu thực hiện bởi CNN, các thông số cần phải được điều chỉnh tương đương với mỗi nhiệm vụ cần thực thi. Các cơ chế và kỹ thuật khác nhau như khởi tạo trọng số và điều chỉnh mạng được thực hiện để tối ưu hóa mạng nơ-ron.

* + - 1. **Khởi tạo Weight.**

Để đào tạo CNN một cách ổn định, cần khởi tạo một trọng số chính xác, nếu không đều này sẽ dẫn đến những khó khăn khác nhau trong quá trình học, tùy vào phương pháp học được sử dụng. Ví dụ, trong trường hợp sử dụng phương pháp back-propagation, việc khởi tạo trọng số không chính xác sẽ dẫn đến việc biến mất hoặc bùng nổ của vấn đề gradient. Có kỹ thuật này là khởi tạo Gaussian Random và khởi tạo Uniform Random.

* + - * 1. **Khởi tạo Gaussian Random.**

Kĩ thuật khởi tạo này được sử dụng cho các convolution và fully connected layers của CNN với ma trận ngẫu nhiên của các phần tử được lấy mẫu từ Gaussian với giá trị trung bình bằng 0 và độ lệch nhỏ.

* + - * 1. **Khởi tạo Uniform Random.**

Kỹ thuật khởi tạo này được sử dụng cho các convolution và fully connected layers của CNN với ma trận ngẫu nhiên của các phần tử được lấy mẫu từ Uniform với giá trị trung bình bằng 0 và độ lệch nhỏ.

* + - 1. **Quy định network.**

CNN được mong đợi sẽ học và khái quát hóa; tuy nhiên, trong một số trường hợp các mạng không thể học và tổng quát hóa vì chúng có xu hướng phù hợp quá mức với dữ liệu đào tạo trong quá trình học. Mạng được kỳ vọng sẽ là học hỏi thay vì ghi nhớ để có thể khái quát hóa và thích ứng tốt hơn với dữ liệu không nhìn thấy được sử dụng để ngăn chặn overfitting. Data augmentation, dropout, batch normalization, ensemble model averaging, và early stopping là những ví dụ về kỹ thuật chính quy hóa.

ConvNet (convolutional neural network) có thể phân loại dữ liệu ngẫu nhiên thành khác nhau. Dữ liệu được phân loại dựa trên các tính năng, đặc điểm và phẩm chất được chia sẻ với các bộ dữ liệu khác trong các danh mục tương ứng. Khi so sánh với các thuật toán phân loại khác, ConvNet dễ huấn luyện hơn vì nó yêu cầu ít kết nối và tham số hơn, như đã được chứng minh bằng kết quả thu được từ ImageNet Large Scale Visual Recognition Competition (ILSVRC). Trong cuộc thi này deep ConvNet đã đạt được top 1 về hiệu suất, hoạt động tốt hơn đáng kể so với các thuật toán được sử dụng trước đây. Deep ConvNet đã phân loại 1,2 triệu hình ảnh có độ phân giải cao trong cuộc thi thành 1000 lớp khác nhau. ConvNet được sử dụng trong các ứng dụng khác nhau, phân loại các phạm vi dữ liệu khác nhau từ hình ảnh đến văn bản, v…v…. Một số bộ phân loại ConvNet hiện đại sẽ được nêu ra trong phần này.

* + 1. **Bộ phân loại ConvNet.**
       1. **Nhận dạng khuôn mặt dựa trên CNN (Face recognition).**

Trong W. L.-m. Hu Yu-lu, "Hand gesture recognition based on convolutional neural networks," in LIDAR Imaging Detection and Target Recognition 2017, Changchun, 2017, một phương pháp nhận dạng khuyên mặt dựa trên ConvNet được đề xuất, bao gồm 9 layer, 3 trong số đó là convolution layers, 2 là pooling layers, 2 là fully connected layers và SoftMax regression layer. Mạng này đã được thử nghiệm trên Caffe, một khung Deep Learning có khả năng xử lý hơn 60 triệu hình ảnh mỗi ngày. Khung Caffe đã đào tạo và thử nghiệm hai cơ sở dữ liệu khuôn mặt khác nhau. Cơ sở dữ liệu khuôn mặt ORL bao gồm tổng cộng 400 hình ảnh cho 40 người. Cơ sở dữ liệu khuôn mặt AR chứa tổng cộng 2600 hình ảnh cho 100 người. Sau khoảng 200 lặp lại, mạng đạt tỉ lệ chính xác là 90%. Sau 2000 lần lặp, đạt 97%. Cuối cùng, sau 10.000 lần lặp, tỷ lệ nhận dạng cho tất cả dữ liệu khuôn mặt của AR và ORL đạt độ chính xác lần lượt là 99, 82% và 99,78%. Khi so sánh với các phương pháp khác được sử dụng cho cơ sở dữ liệu AR và cơ sở dữ liệu ORL, ConvNet được sử dụng đứng đầu vì độ chính xác và mạnh mẽ.

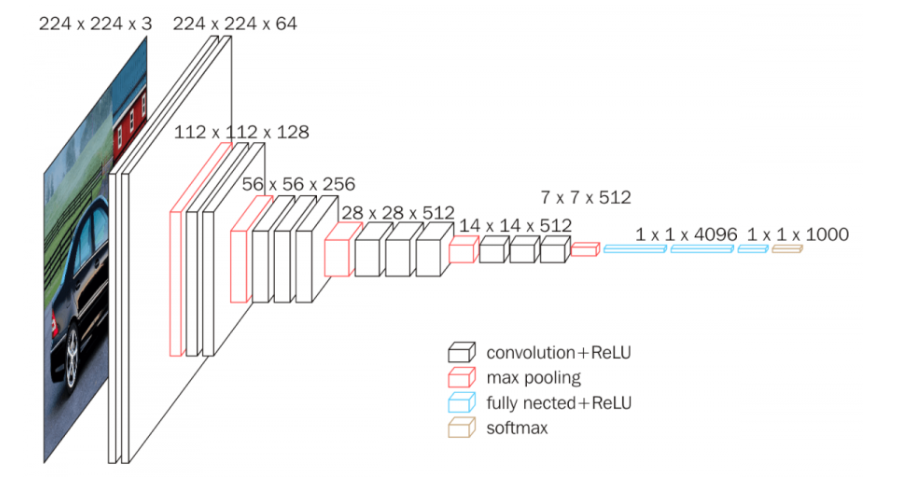
* + - 1. **Nhận dạng cử chỉ tay dựa trên CNN (Hand gesture recognition).**

Trong W. L.-m. Hu Yu-lu, "Hand gesture recognition based on convolutional neural networks," in LIDAR Imaging Detection and Target Recognition 2017, Changchun, 2017. Một thuật toán nhận dạng cử chỉ tay dựa trên ConvNet được đề cử để nhận dạng 10 cử chỉ. Mạng đã được đào tạo bằng cách sử dụng 6000 hình ảnh cử chỉ tay và 1100 mẫu thử nghiệm. Mạng đã đạt tỉ lệ nhận dạng là 98%, đứng đầu các thuật toán nhận dạng khác.

* + 1. **Một số mạng CNN nổi tiếng.**
       1. **VGG 16**

VGG16 là mạng convolutional neural network được đề xuất bởi K. Simonyan and A. Zisserman, University of Oxford. Model sau khi train bởi mạng VGG16 đạt độ chính xác 92.7% top-5 test trong dữ liệu ImageNet

gồm 14 triệu hình ảnh thuộc 1000 lớp khác nhau. Giờ áp dụng kiến thức ở trên để phân tích mạng VGG 16.

****

Hình 1.6. Minh họa kiến trúc của VGG 16

Phân tích về VGG 16:

* Convolutional layer: kích thước 3\*3, padding=1, stride=1. Tại sao không ghi stride, padding mà vẫn biết? Vì mặc định sẽ là stride=1 và padding để cho output cùng width và height với input.
* Pool/2 : max pooling layer với size 2\*2
* 3\*3 conv, 64: thì 64 là số kernel áp dụng trong layer đấy, hay depth của output của layer đấy.
* Càng các convolutional layer sau thì kích thước width, height càng giảm nhưng depth càng tăng.
* sau khá nhiều convolutional layer và pooling layer thì dữ liệu được flatten và cho vào fully connected layer.
  + - 1. **ResNet**

ResNet (Residual Network) được giới thiệu đến công chúng vào năm 2015 và thậm chí đã giành được vị trí thứ 1 trong cuộc thi ILSVRC 2015 với tỉ lệ lỗi top 5 chỉ 3.57%. Không những thế nó còn đứng vị trí đầu tiên trong cuộc thi ILSVRC and COCO 2015 với ImageNet Detection, ImageNet localization, Coco detection và Coco segmentation.Hiện tại thì có rất nhiều biến thể của kiến trúc ResNet với số lớp khác nhau như ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152,...Với tên là ResNet theo sau là một số chỉ kiến trúc ResNet với số lớp nhất định.

* + - * 1. **Lợi thế của Resnet**

Mạng ResNet (R) là một mạng CNN được thiết kế để làm việc với hàng trăm hoặc hàng nghìn lớp chập. Một vấn đề xảy ra khi xây dựng mạng CNN với nhiều lớp chập sẽ xảy ra hiện tượng **Vanishing Gradient** dẫn tới quá trình học tập không tốt.

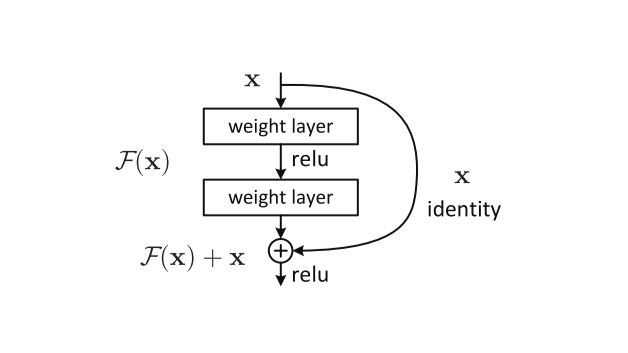
Backpropagation Algorithm (thuật toán lan truyền ngược) là một kĩ thuật thường được sử dụng trong trong quá trình training DNNs. Ý tưởng chung của thuật toán là sẽ đi từ output layer đến input layer và tính toán gradient của cost function tương ứng cho từng parameter (weight) của network. Gradient Descent, sau đó, sẽ được sử dụng để cập nhật các parameter đó.

Quá trình trên sẽ được lặp lại cho tới khi các parameter của network hội tụ. Thông thường chúng ta sẽ có một hyperparameter định nghĩa cho số lượng vòng lặp để thực hiện quá trình trên. Hyperparameter đó thường được gọi là số Epoch (hay số lần mà training set được duyệt qua một lần và weights được cập nhật). Nếu số lượng vòng lặp quá nhỏ, DNN có thể sẽ không cho ra kết quả tốt, và ngược lại thì thời gian training sẽ quá dài nếu số lượng vòng lặp quá lớn. Ở đây ta có một tradeoff giữa độ chính xác và thời gian training.

Tuy nhiên trên thực tế gradients thường sẽ có giá trị nhỏ dần khi đi xuống các layer thấp hơn. Kết quả là các cập nhật thực hiện bởi Gradient Descent không làm thay đổi nhiều weights của các layer đó, khiến chúng không thể hội tụ và DNN sẽ không thu được kết quả tốt. Hiện tượng này được gọi là **Vanishing Gradients => Resnet ra đời để giải quyết vấn đề này.**

* + - * 1. **Kiến trúc của mạng Resnet.**

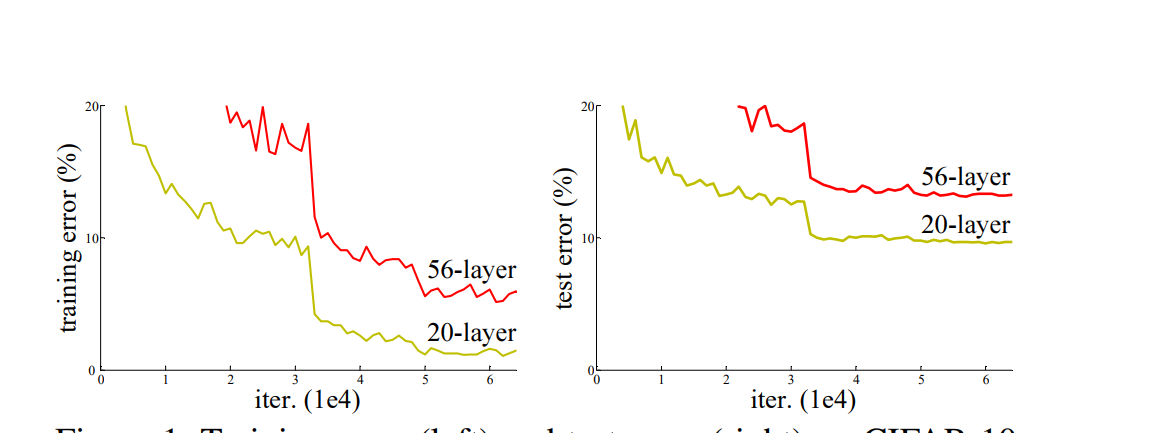
Giải pháp mà ResNet đưa ra là sử dụng kết nối "tắt" đồng nhất để xuyên qua một hay nhiều lớp. Một khối như vậy được gọi là một Residual Block, như trong hình 4.2 :



Hình 1.7. Mô tả kiến trúc cơ bản của ResNet

ResNet gần như tương tự với các mạng gồm có convolution, pooling, activation và fully-connected layer. Ảnh bên trên hiển thị khối dư được sử dụng trong mạng. Xuất hiện một mũi tên cong xuất phát từ đầu và kết thúc tại cuối khối dư. Hay nói cách khác là sẽ bổ sung Input X vào đầu ra của layer, hay chính là phép cộng mà ta thấy trong hình minh họa, việc này sẽ chống lại việc đạo hàm bằng 0, do vẫn còn cộng thêm X. Với H(x) là giá trị dự đoán, F(x) là giá trị thật (nhãn), chúng ta muốn H(x) bằng hoặc xấp xỉ F(x).

Theo như nghiên cứu [1] Resnet có thể sử dụng được 1 số lượng layer rất lớn mà không xảy ra hiện tượng mất đạo hàm (**Vanishing Gradients**).



Hình 1.8. Hình ảnh Vanishing Gradients

* + 1. **Sử dụng CNN vào bài toán nhận đối tượng.**

Một trong những lĩnh vực quan trọng của Trí tuệ nhân tạo (Artificial Intelligence) là thị giác máy (Computer Vision). Computer Vision là một lĩnh vực bao gồm các phương pháp thu nhận, xử lý ảnh kỹ thuật số, phân tích và nhận dạng các hình ảnh, phát hiện các đối tượng, tạo ảnh, siêu phân giải hình ảnh và nhiều hơn vậy. Object Detection có lẽ là khía cạnh sâu sắc nhất của thị giác máy đo số lần sử dụng trong thực tế.

Object Detection đề cập đến khả năng của hệ thống máy tính và phần mềm để định vị các đối tượng trong một hình ảnh và xác định từng đối tượng. Object Detection đã được sử dụng rộng rãi để phát hiện khuôn mặt, phát hiện xe, đếm số người đi bộ, hệ thống bảo mật và xe không người lái. Có nhiều cách để nhận diện đối tượng có thể được sử dụng cũng như trong nhiều lĩnh vực thực hành. Giống như mọi công nghệ khác, một loạt các ứng dụng sáng tạo và tuyệt vời của Object Detection sẽ đến từ các lập trình viên và các nhà phát triển phần mềm.

Bắt đầu sử dụng các phương pháp nhận diện đối tượng hiện đại trong các ứng dụng và hệ thống, cũng như xây dựng các ứng dụng mới dựa trên các phương pháp này. Việc triển nhận diện đối tượng sớm liên quan đến việc sử dụng các thuật toán cổ điển, giống như các thuật toán được hỗ trợ trong OpenCV, thư viện computer vision phổ biến. Tuy nhiên, các thuật toán cổ điển này không thể đạt được hiệu suất đủ để làm việc trong các điều kiện khác nhau.

Việc áp dụng đột phá và nhanh chóng của deep learning vào năm 2012 đã đưa vào sự tồn tại các thuật toán và phương pháp phát hiện đối tượng hiện đại và chính xác cao như R-CNN, Fast-RCNN, Faster-RCNN, RetinaNet và nhanh hơn nhưng rất chính xác như SSD và YOLO. Sử dụng các phương pháp và thuật toán này, dựa trên deep learning và cũng dựa trên việc học máy đòi hỏi rất nhiều kiến thức về toán học và việc học sâu. Có hàng triệu chuyên gia lập trình và các nhà phát triển phần mềm muốn tích hợp và tạo ra các sản phẩm mới sử dụng object detection. Nhưng công nghệ này xa tầm tay của họ và phức tạp để hiểu và sử dụng thực tế của nó.

* + 1. Tổng quan YOLO

Về độ chính xác thì YOLO có thể không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong các lớp mô hình object detection. Nó có thể đạt được tốc độ gần như real time mà độ chính xác không quá giảm so với các model thuộc top đầu. YOLO là thuật toán object detection nên mục tiêu của mô hình không chỉ là dự báo nhãn cho vật thể như các bài toán classification mà nó còn xác định location của vật thể. Do đó YOLO có thể phát hiện được nhiều vật thể có nhãn khác nhau trong một bức ảnh, thay vì chỉ phân loại duy nhất một nhãn cho một bức ảnh.

Trong đề tài này, nhóm chúng tôi sẽ ứng dụng một trong những thế hệ mới nhất của thuật toán YOLO ở thời điểm hiện tại – YOLOv5. Mô hình YOLOv5 được phát hành sau khi công bố phiên bản thứ tư khoảng 1 tháng, cả hai nhà nghiên cứu là Glenn Jocher (tác giả chính của YOLOv5) và Alexey Bochkovsky (tác giả chính của YOLOv4) đều áp dụng những sáng tạo tiên tiến nhất trong lĩnh vực thị giác máy tính vào thời điểm đó. Điều đó làm cho kiến ​​trúc của YOLOv4 và YOLOv5 rất giống nhau và nó khiến nhiều người không hài lòng với cái tên YOLOv5 (thế hệ thứ 5 của YOLO) khi nó không chứa đựng nhiều cải tiến vượt trội so với phiên bản YOLOv4 trước đó. Bên cạnh đó, Glenn đã không công bố bất kỳ bài báo nào cho YOLOv5, khiến nhiều người nghi ngờ hơn về YOLOv5. Tuy nhiên, sau một thời gian, YOLOv5 đã chứng tỏ hiệu suất cao hơn YOLOv4 trong một số hoàn cảnh nhất định và phần nào tạo được niềm tin trong cộng đồng thị giác máy tính bên cạnh YOLOv4.

* + - 1. YOLOv5
         1. **Tổng quan**

Bên cạnh đó, Glenn Jocher cũng là người phát minh ra phương pháp tăng cường dữ liệu Mosaic và được Alexey Bochkovsky thừa nhận trong bài báo YOLOv4 (Bochkovskiy, et al., 2020). Tuy nhiên, mô hình YOLOv5 của ông đã gây ra nhiều tranh cãi trong cộng đồng thị giác máy tính vì tên gọi và những cải tiến của nó.

YOLOv5 sở hữu những lợi thế về kỹ thuật. YOLOv5 được viết bằng ngôn ngữ lập trình Python thay vì C như các phiên bản trước. Điều đó làm cho việc cài đặt và tích hợp trên các thiết bị IoT dễ dàng hơn. Ngoài ra, cộng đồng PyTorch cũng lớn hơn cộng đồng Darknet, đồng nghĩa với việc PyTorch sẽ nhận được nhiều đóng góp và tiềm năng phát triển hơn trong tương lai. Do được viết bằng 2 ngôn ngữ khác nhau trên 2 framework khác nhau nên việc so sánh hiệu năng giữa YOLOv4 và YOLOv5 rất khó chính xác. Nhưng sau một thời gian, YOLOv5 đã chứng tỏ hiệu suất cao hơn YOLOv4 trong một số hoàn cảnh nhất định và phần nào tạo được niềm tin trong cộng đồng thị giác máy tính bên cạnh YOLOv4.

* + - * 1. **Điểm đáng chú ý ở YOLOv5**

Như đã đề cập ở trên, kiến trúc YOLOv5 đã tích hợp những cải tiến mới nhất tương tự như kiến trúc YOLOv4, do đó không có nhiều sự khác biệt về mặt lý thuyết. Tác giả đã không xuất bản bài báo chi tiết mà chỉ đưa ra một kho lưu trữ trên Github và cập nhật các cải tiến ở đó. Bằng cách mổ xẻ mã cấu trúc của nó trong tệp yaml, mô hình YOLOv5 có thể được tóm tắt như sau (Jocher, 2020):

* Backbone: Focus structure, CSP network
* Neck: SPP block, PANet
* Head: YOLOv3 head sử dụng GIoU-loss

Điểm đáng chú ý mà tác giả YOLOv5 đề cập là sự khác biệt về mặt kỹ thuật. Joseph Redmon đã giới thiệu cấu trúc hộp neo trong YOLOv2 và quy trình chọn hộp neo có kích thước và hình dạng gần giống với hộp giới hạn chân lý mặt đất trong tập huấn luyện. Bằng cách sử dụng thuật toán phân cụm k-mean với các giá trị 𝑘 khác nhau, các tác giả đã chọn 5 hộp neo phù hợp nhất cho tập dữ liệu COCO (chứa 80 lớp) và sử dụng chúng làm mặc định. Điều đó làm giảm thời gian đào tạo và tăng độ chính xác của mạng. Tuy nhiên, khi áp dụng 5 hộp neo này cho một tập dữ liệu duy nhất (chứa một lớp không thuộc 80 lớp trong tập dữ liệu COCO), các hộp neo này không thể nhanh chóng thích ứng với các hộp giới hạn chân lý cơ bản của tập dữ liệu duy nhất này. Ví dụ: tập dữ liệu về hươu cao cổ phù hợp với các hộp neo có hình dạng mỏng và cao hơn hộp hình vuông. Để giải quyết vấn đề này, các kỹ sư thị giác máy tính thường chạy thuật toán phân cụm k-mean trên tập dữ liệu duy nhất để có được các hộp neo phù hợp nhất cho dữ liệu trước tiên. Sau đó, các tham số này sẽ được cấu hình thủ công trong kiến ​​trúc YOLO.

Glenn Jocher đề xuất tích hợp quy trình lựa chọn hộp neo vào YOLOv5. Do đó, mạng không phải xem xét bất kỳ tập dữ liệu nào được sử dụng làm đầu vào, nó sẽ tự động "học" các hộp neo tốt nhất cho tập dữ liệu đó và sử dụng chúng trong quá trình đào tạo. (Solawetz, 2020)

* + - 1. **YOLOv8**

YOLOv8 là mô hình YOLO tiên tiến nhất có thể được sử dụng cho các tác vụ phát hiện đối tượng, phân loại hình ảnh và phân đoạn phiên bản. YOLOv8 được phát triển bởi Ultralytics, người cũng đã tạo ra mô hình YOLOv5 có sức ảnh hưởng và định hình ngành. YOLOv8 bao gồm nhiều thay đổi và cải tiến về kiến trúc và trải nghiệm của nhà phát triển so với YOLOv5. YOLOv8 đang được phát triển tích cực tại thời điểm viết bài đăng này. YOLOv8 có những ưu điểm đặc biệt như:

YOLOv8 có tỷ lệ chính xác cao được đo bằng COCO và Roboflow 100, YOLOv8 đi kèm với rất nhiều tính năng thuận tiện cho nhà phát triển, từ CLI dễ sử dụng đến gói Python có cấu trúc tốt.

Có một cộng đồng lớn xung quanh YOLO và một cộng đồng đang phát triển xung quanh mô hình YOLOv8, nghĩa là có nhiều người trong giới thị giác máy tính có thể hỗ trợ bạn khi bạn cần hướng dẫn, YOLOv8 chưa có bài báo nào được xuất bản, vì vậy chúng tôi thiếu thông tin chi tiết trực tiếp về phương pháp nghiên cứu trực tiếp và các nghiên cứu tạo ra nó.

Ở đây chúng tôi tóm tắt nhanh về các bản cập nhật mô hình có tác động và sau đó chúng tôi sẽ xem xét đánh giá của mô hình, YOLOv8 là một mô hình không có neo. Điều này có nghĩa là nó dự đoán trực tiếp tâm của một đối tượng thay vì phần bù từ hộp neo đã biết, các hộp neo là một phần khét tiếng phức tạp của các mô hình YOLO trước đó, vì chúng có thể đại diện cho việc phân phối các hộp của điểm chuẩn mục tiêu nhưng không phải là sự phân phối của tập dữ liệu tùy chỉnh, Chuyển đổi đầu tiên của gốc 6x6 được thay thế bằng 3x3, khối xây dựng chính đã được thay đổi và C2f thay thế C3. Mô-đun được tóm tắt trong hình bên dưới, trong đó "f" là số lượng tính năng, "e" là tốc độ mở rộng và CBS là một khối bao gồm a Conva BatchNormvà a SiLUsau. Trong C2f, tất cả các đầu ra từ Bottleneck (tên ưa thích cho hai 3x3 convs với các kết nối còn lại) được nối với nhau. Trong khi C3chỉ đầu ra của cái cuối cùng Bottleneck được sử dụng C2f Mô-đun YOLOv8 mới Giống Bottleneck như trong YOLOv5 nhưng kích thước hạt nhân của chuyển đổi đầu tiên đã được thay đổi từ 1x1thành 3x3. Từ thông tin này, chúng ta có thể thấy rằng YOLOv8 đang bắt đầu trở lại khối ResNet được xác định vào năm 2015. Ở phần cổ, các tính năng được nối trực tiếp mà không bắt buộc các kích thước kênh giống nhau. Điều này làm giảm số lượng tham số và kích thước tổng thể của các tenxơ. YOLOv8 tăng cường hình ảnh trong quá trình đào tạo trực tuyến. Tại mỗi thời điểm, mô hình sẽ thấy một biến thể hơi khác nhau của hình ảnh mà nó đã được cung cấp. Một trong những phần mở rộng đó được gọi là phần mở rộng khảm. Điều này liên quan đến việc ghép bốn hình ảnh lại với nhau, buộc mô hình tìm hiểu các đối tượng ở các vị trí mới, che khuất một phần và dựa trên các pixel xung quanh khác nhau.

* + 1. **Đo lường đối tượng thời gian thực**

Đối với bài toán xác định nhóm người thì việc khoảng cách giữa các cá đối tượng người là điều kiện để xác định thế nào là một nhóm người. Ở đây chúng tôi không sử dụng tính khoảng cách cụ thể ngoài thực tế.

Điều này có thể làm phức tạp bài toán, việc xác định khoảng cách chiều sâu nếu 2 người bị đè Bbox là rất khó khăn vì phải xử lí cả chiều sâu của ảnh, làm cho việc xử lý thông tin chậm chạp hơn. Chúng tôi đã sử dụng cách tiếp cận đơn giản hơn dựa trên khái niệm đám đông của nhóm. Đối với môi trường lí tưởng, góc quay phù hợp thì việc xác định đám đông chỉ cần dựa vào điểm ảnh pixel.

Việc xây dựng hàm khoảng cách này khá đơn giản, dựa vào đầu ra của YOLOv5, tôi đã lấy các giá trị xy tọa độ của Bbox để thực hiện, kết hợp với thư viện OpenCV, là một thư viện mã nguồn mở hàng đầu cho thị giác máy tính (computer vision), xử lý ảnh và máy học, và các tính năng tăng tốc GPU trong hoạt động thời gian thực.

Dựa vào các điểm ảnh pixel chúng tôi thực nghiệm và tìm ra độ dài pixel trong các ảnh phù hợp với khoảng cách góc quay, điều này làm cho khoảng cách pixel trong gần tương đương khoảng cách trong thời gian thực.

# NGHIÊN CỨU MẠNG CNN ĐỂ GIẢI BÀI TOÁN TỐI ƯU TỔ HỢP “NHẬN DIỆN RĂNG TRONG CHỤP X-QUANG”

* 1. **Giải thuật**
     1. **Mô tả bài toán**

Bài toán nhận diện răng trong chụp X-quang là một bài toán trong lĩnh vực xử lý ảnh y sinh. Bài toán này có thể được giải quyết bằng các phương pháp như phân đoạn ảnh, phân tích hình học và các phương pháp học sâu. Mỗi chiếc răng cần phải được đặt vào đúng vị trí của nó trong tổ hợp vị trí các chiếc răng. Đầu ra của bài toán là những chiếc răng đang ở sai vị trí trong tổ hợp vị trí cá chiếc răng. Mục đích của việc chụp X-quang răng là để phát hiện những vấn đề như răng sâu, chân răng bị gãy, xương răng bị tổn thương, phát hiện những vị trí răng mọc lệch hay những chiếc răng mọc sâu vào trong nướu, tìm ra u nang và theo dõi sự tăng trưởng bất thường.

Dữ liệu đầu vào của bài toán là một hình ảnh chụp X-quang. Đầu ra của bài toán là những chiếc răng đang ở sai vị trí trong tổ hợp vị trí cá chiếc răng.

* + 1. **Mô tả dữ liệu dataset**

Để đào tạo mô hình phát hiện răng, chúng tôi đã tạo tập dữ liệu về người của riêng mình, bằng cách chuẩn bị khoảng … hình ảnh về răng khi chụp X-quang răng từ web. Để đạt được mô hình YOLO mạnh mẽ, chúng tôi sẽ tối đa ảnh để giảm lỗi xác định sai. Vì tập dữ liệu của chúng tôi chưa đủ lớn, nên tối sẽ thu hẹp quy trình đào tạo bằng cách sử dụng các kĩ thuật transfer learning.



Hình 2.1. Ảnh chụp X-Quang răng

Hầu hết các nền tảng chú thích đều hỗ trợ xuất ở định dạng ghi label YOLO, cung cấp một tệp văn bản chú thích cho mỗi hình ảnh. Mỗi tệp văn bản chứa một chú thích hộp giới hạn cho mỗi đối tượng trong hình ảnh. Các chú thích được chuẩn hóa theo kích thước hình ảnh và nằm trong phạm vi từ 0 đến 1. Chúng được thể hiện ở định dạng sau:

<object-class-ID> <X center> <Y center> <Chiều rộng hộp> <Chiều cao hộp>

* + 1. **Thuật toán tối ưu**
       1. **Thuật toán Transfer learning**

Transfer learning là một kỹ thuật trong lĩnh vực học máy giúp áp dụng kiến thức đã học được từ một nhiệm vụ vào một nhiệm vụ khác. Thay vì huấn luyện một mô hình hoàn toàn mới từ đầu cho mỗi tác vụ, ta có thể sử dụng một mô hình đã được huấn luyện trước đó và tinh chỉnh lại cho các tác vụ mới.

Khi huấn luyện một mô hình trong lĩnh vực học máy, mô hình được điều chỉnh dựa trên dữ liệu đào tạo được cung cấp. Tuy nhiên, khi giải quyết một tác vụ mới, việc tìm kiếm dữ liệu đào tạo thích hợp để huấn luyện một mô hình hoàn toàn mới có thể rất khó khăn, đòi hỏi thời gian và nỗ lực lớn. Thay vì vậy, ta có thể sử dụng các mô hình đã được huấn luyện trước đó và tinh chỉnh chúng cho các tác vụ mới.

Các mô hình được sử dụng cho transfer learning thường được huấn luyện trên các nhiệm vụ lớn và phức tạp, chẳng hạn như nhận dạng hình ảnh hoặc xử lý ngôn ngữ tự nhiên. Chúng ta có thể sử dụng các mô hình này để phát triển các mô hình nhỏ hơn, cần ít dữ liệu hơn và đáp ứng được nhu cầu thực tế.

Kỹ thuật transfer learning có thể được áp dụng trong nhiều trường hợp, chẳng hạn như:

Sử dụng một mô hình đã được huấn luyện trước đó trên một tập dữ liệu lớn như ImageNet để giải quyết bài toán nhận dạng hình ảnh cho một tập dữ liệu nhỏ hơn.

Sử dụng một mô hình đã được huấn luyện trên tập dữ liệu văn bản lớn để giải quyết các bài toán xử lý ngôn ngữ tự nhiên khác nhau như phân loại tin tức, dịch thuật hoặc tóm tắt văn bản.

Sử dụng một mô hình đã được huấn luyện trên tập dữ liệu tín hiệu y tế để giải quyết bài toán phân tích hình ảnh y học.

Một trong những ưu điểm của transfer learning là nó có thể giảm thời gian và chi phí huấn luyện mô hình, đồng thời cải thiện khả năng dự đoán của mô hình. Bằng cách sử dụng mô hình đã được huấn luyện trước đó, ta đã giảm bớt phần lớn thời gian và công sức cần thiết để huấn luyện một mô hình mới. Ngoài ra, việc sử dụng các mô hình đã được huấn luyện trước đó cũng có thể giúp cải thiện khả năng dự đoán của mô hình, bởi vì các mô hình này đã học được nhiều kiến thức từ các nhiệm vụ trước đó và có thể cung cấp những thông tin quan trọng cho các tác vụ mới.

Tuy nhiên, transfer learning cũng có những hạn chế. Một trong số đó là khi các tác vụ mới khác hoàn toàn với các tác vụ đã được huấn luyện trước đó, kỹ thuật này có thể không hiệu quả. Ngoài ra, nếu không chọn mô hình cơ sở phù hợp hoặc không áp dụng các kỹ thuật tinh chỉnh thích hợp, transfer learning cũng có thể gây ra các vấn đề về quá khớp hoặc hiệu suất kém.

Tóm lại, transfer learning là một kỹ thuật quan trọng trong lĩnh vực học máy giúp tận dụng kiến thức đã học được từ các nhiệm vụ trước đó và áp dụng chúng vào các tác vụ mới. Kỹ thuật này có thể giảm thời gian và chi phí huấn luyện mô hình, đồng thời cải thiện khả năng dự đoán của mô hình. Tuy nhiên, để sử dụng transfer learning hiệu quả, ta cần lựa chọn mô hình cơ sở phù hợp và áp dụng các kỹ thuật tinh chỉnh thích hợp.

Bắt đầu đào tạo từ mô hình trước:

Vì tập dữ liệu về ảnh răng chụp X-quang của chúng tôi tương đối nhỏ (~ 1000 hình ảnh), việc transfer learning được mong đợi sẽ tạo ra kết quả tốt hơn so với huấn luyện từ đầu. Mô hình mặc định của Ultralytic đã được đào tạo trước qua tập dữ liệu COCO, mặc dù cũng có hỗ trợ cho các mô hình được đào tạo trước khác (VOC, Argoverse, VisDrone, GlobalWheat, xView, Objects365, SKU-110K). COCO là một tập dữ liệu phát hiện đối tượng với hình ảnh từ các cảnh hàng ngày. Nó chứa 80 lớp, Mô hình của chúng tôi sẽ được khởi tạo với các trọng số từ một mô hình COCO được đào tạo trước, bằng cách chuyển tên của mô hình cho đối số 'weights'. Mô hình được đào tạo trước sẽ được tự động tải xuống.

Weights & Biases là nền tảng học máy dành cho các nhà phát triển để xây dựng các mô hình tốt hơn nhanh hơn. Sử dụng các công cụ nhẹ, có thể tương tác của W&B để nhanh chóng theo dõi các thử nghiệm, phiên bản và lặp lại trên tập dữ liệu, đánh giá hiệu suất mô hình, tái tạo mô hình, trực quan hóa kết quả và hồi quy tại chỗ, đồng thời chia sẻ phát hiện với đồng nghiệp.

* + - 1. **Thuật toán K-mean**

Thuật toán K-mean là một trong những thuật toán phổ biến nhất trong lĩnh vực học không giám sát (unsupervised learning), được sử dụng để phân nhóm dữ liệu. Ý tưởng của thuật toán K-mean là tìm các cụm (clusters) của dữ liệu sao cho tổng bình phương khoảng cách giữa các điểm dữ liệu và trung tâm cụm tương ứng là nhỏ nhất.

Để hiểu rõ hơn về thuật toán K-mean, chúng ta sẽ đi vào chi tiết các bước trong thuật toán.

1.Chọn số lượng cụm cần tìm (K) và khởi tạo các trung tâm cụm ngẫu nhiên:

* Trong bước này, chúng ta cần xác định số lượng cụm cần tìm (K), đây là một tham số quan trọng của thuật toán. Số lượng cụm cần tìm phụ thuộc vào bài toán và không có công thức nào để tính toán số lượng cụm tối ưu. Thông thường, người sử dụng thuật toán sẽ thử nghiệm với nhiều giá trị khác nhau của K để tìm ra giá trị tối ưu nhất.
* Sau khi xác định được số lượng cụm cần tìm, chúng ta cần khởi tạo các trung tâm cụm ngẫu nhiên. Việc khởi tạo trung tâm cụm đầu tiên rất quan trọng vì nó ảnh hưởng đến kết quả cuối cùng của thuật toán.

2. Gán từng điểm dữ liệu cho cụm có trung tâm gần nó nhất

* Trong bước này, chúng ta tính toán khoảng cách giữa từng điểm dữ liệu và trung tâm của các cụm đã khởi tạo ở bước trước.
* Sau đó, chúng ta gán từng điểm dữ liệu vào cụm có trung tâm gần nó nhất.

3. Tính toán lại trung tâm của các cụm mới được hình thành

* Sau khi đã gán các điểm dữ liệu vào các cụm, chúng ta cần tính toán lại trung tâm của các cụm mới được hình thành.
* Trung tâm của một cụm được tính bằng cách lấy trung bình của tất cả các điểm dữ liệu trong cụm đó.

4. Lặp lại bước 2 và 3 cho đến khi trung tâm các cụm không thay đổi hoặc đạt được điều kiện dừng nào đó.

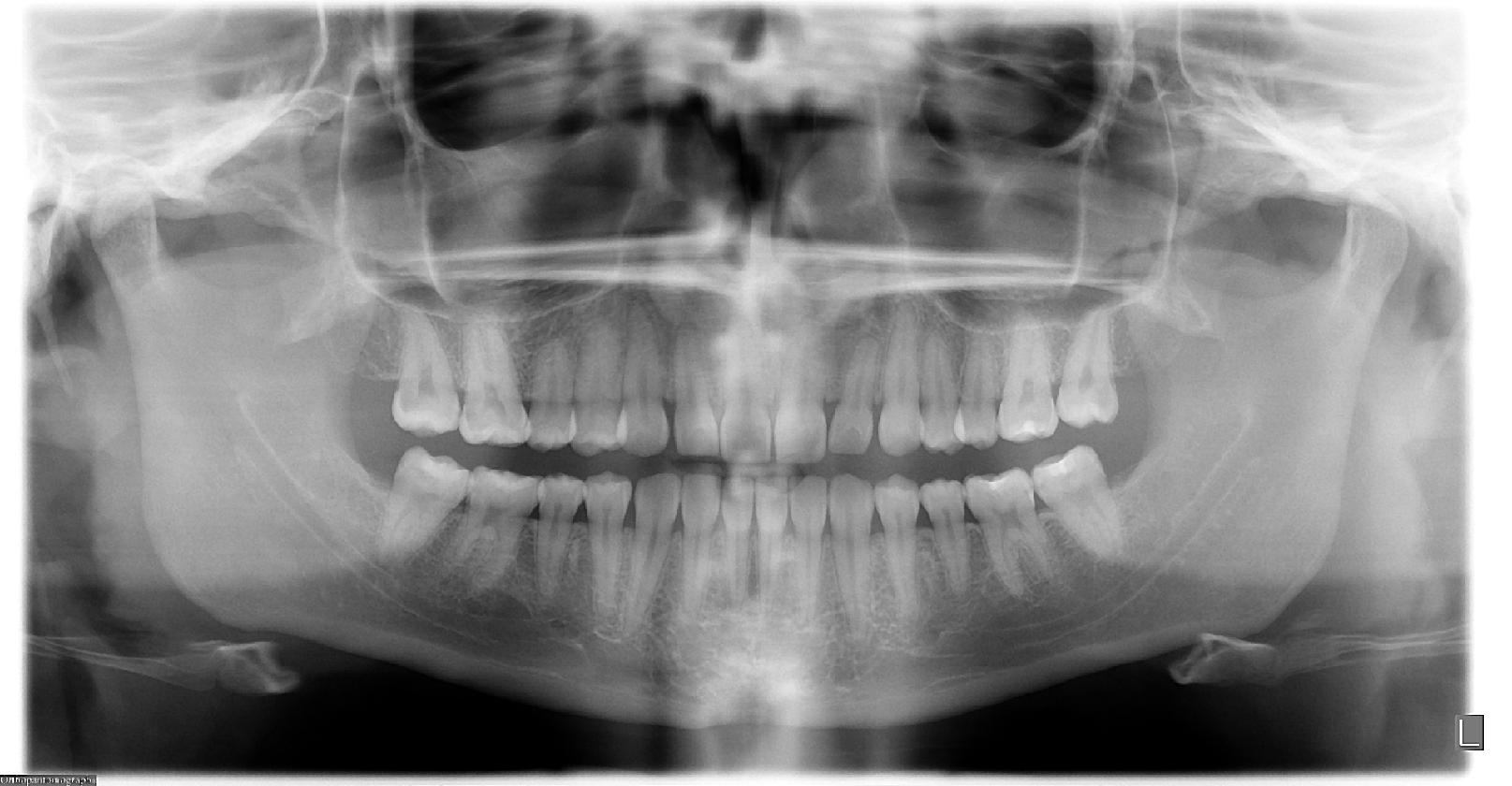
* Thuật toán sẽ tiếp tục lặp lại bước 2 và 3 cho đến khi trung tâm của các cụm không thay đổi hoặc đạt được điều kiện dừng nào đó. Các điều kiện dừng thường được sử dụng là số lần lặp hoặc sai số trung bình giữa các lần tính toán trung tâm cụm.

Thuật toán K-mean có nhiều ưu điểm:

* Đơn giản và dễ hiểu: K-mean là một thuật toán đơn giản, dễ hiểu và dễ triển khai.
* Hiệu quả với dữ liệu lớn: K-mean có thể xử lý dữ liệu lớn một cách hiệu quả.
* Độ phức tạp thấp: Độ phức tạp của K-mean là O(nkt), với n là số lượng điểm dữ liệu, k là số lượng cụm và t là số lần lặp lại của thuật toán. Vì vậy, K-mean có thể xử lý các bài toán với dữ liệu lớn.
* Phù hợp với dữ liệu số: K-mean được sử dụng rộng rãi cho việc phân cụm dữ liệu số, đặc biệt là khi số lượng chiều của dữ liệu lớn.
* Dễ mở rộng: Thuật toán K-mean có thể được mở rộng để xử lý các bài toán phân cụm không phân tán và có tính toán song song.
  1. **Kết quả chạy các bộ dữ liệu**
     1. **Môi trường thực nghiệm**
* Mô hình được huấn luyện trên máy chủ với cấu hình:
* GPU: 4 GPU M10
* RAM: 32 GB
* CPU: Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz
* Hệ điều hành: Ubuntu 18.04
* Cuda: 11.8
* TensorRT: 8.5.1.7
* Quản lí thư viện: Anaconda
  + 1. **Dữ liệu thực nghiệm**

Trong phần này, nhóm mô tả tập dữ liệu được sử dụng trong các thử nghiệm phát hiện đối tượng. Bộ dữ liệu bao gồm các hình ảnh được thu thập từ nhiều nguồn khác nhau, bao gồm cả internet và gán nhãn thủ công. Ngoài ra, các điều kiện khác nhau về ánh sáng, chất lượng máy chụp, … đã được đưa vào tập dữ liệu để đảm bảo tập dữ liệu toàn diện và đa dạng, Các kỹ thuật tăng cường dữ liệu, bao gồm nhiễu, cắt xén và xoay hình ảnh, cũng được sử dụng để mô phỏng các điều kiện giao thông khác nhau.

Các hình ảnh được xử lý trước ở kích thước chuẩn 640x640 trước khi được sử dụng để huấn luyện.



Hình 2.2. Hình ảnh mô tả bộ dữ liệu thực nghiệm

* + 1. **Kết quả chạy các bộ dữ liệu chuẩn**

Các mô hình YOLOv5n, YOLOv5s, YOLOv5m, và YOLOv8n, YOLOv8s, YOLOv8m được huấn luyện từ các pre-trained model sẵn có trên mã nguồn mở, với số lượng epoch là 160 và được huấn luyện trên cùng một tập dữ liệu. Batch size của mô hình là 16. Tất cả 3 mô hình được huấn luyện trên cùng môi trường thực nghiệm đã trình bày ở bên trên

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Params  (M) | FLOPs  (B) | Size | Speed  (ms) | mAP  50 | mAP  50 - 95 |
| YOLOv5n | 2.0 | 7.1 | 640 | 13.2 | 92% | 75% |
| YOLOv5s | 7.6 | 26.4 | 640 | 88.3 | 93.7% | 79% |
| YOLOv5m | 22.0 | 70.8 | 640 | 207.5 | 95.2% | 85% |
| YOLOv8n | 3.0 | 8.1 | 640 | 17.4 | 92.6% | 77.8% |
| YOLOv8s | 11.2 | 28 | 640 | 65,4 | 94% | 83% |
| YOLOv8m | 25.9 | 78.1 | 640 | 114.1 | 97% | 88.5% |

Bảng 2.1. Kết quả huấn luyện và các thông số liên quan

Chú thích:

* *Model*: tên của các mô hình được được thử nghiệm.
* *params*: số lượng tham số của mô hình.
* *FLOPs* (Floating Point Operations): số phép toán số học dấu phẩy động mà một mô hình thực hiện.
* *Size*: kích thước của ảnh sau khi được chuẩn hóa để phục vụ quá trình suy luận.
* *Speed*: tốc độ suy luận của mô hình trên mỗi khung hình.
* *mAP 50*: độ chính xác trung bình của mỗi class, với IoU threshold là 50%
* *mAP 50-95*: độ chính xác trung bình của mỗi class, được tính bằng trung bình của các mAP 50, mAP 55, mAP 60, …, mAP 95

Trong bảng 2.1, ta thấy YOLOv8m là mô hình có chỉ số mAP-50 cao nhất, nhưng lại chạy với tốc độ chậm nhất. Trong công bố chính thức, YOLOv8m có thể đạt tốc độ suy luận lên đến 120 FPS. Tuy nhiên, điều này chỉ đúng khi nó được triển khai trên thiết bị phần cứng GPU V100. Trong bối cảnh hiện tại, khi chạy trên môi trường thực nghiệm, YOLOv8m lại cho ra kết quả không mấy khả quan, khi mà tốc độ suy luận trung bình lên tới 144.1ms, thấp hơn rất nhiều so với kỳ vọng. Điều này cũng một phần là do, YOLOv8m vốn dĩ là một mô hình khá nặng, khi mà số lượng tham số của mô hình lên tới 25.9 triệu tham số, gấp gần 13 lần phiên bản nhẹ nhất của YOLOv5

Trong tình huống này, YOLOv5m lại là mô hình tốt khi trung hòa được cả 2 yếu tố, độ chính xác và tốc độ. Tuy nhiên, trong nhiệm vụ này, YOLOv5n cũng đã làm rất tốt, khi đạt được tốc độ nhanh nhất, và cho ra độ chính xác chấp nhận được. Cùng với độ lớn chỉ khoảng 1.9 triệu tham số và số phép tính chấm động cần thực hiện là ít nhất (FLOPs = 4.5B), đây thực sự là một mô hình khả thi để có thể đưa vào phát triển tích hợp trên các thiết bị biên trong tương lai.

* 1. **Đánh giá giải thuật và dự đoán hướng đi trong tương lai**

Giải thuật đã đáp ứng cơ bản các bước và chạy ra kết quả. Kết quả sau chạy chương trình cho thấy kết quả tối ưu trong các trường hợp.

Giải thuật là một giải pháp tốt và hiệu quả và trong vấn đề này. Trong tương lai, chúng em sẽ tìm hiểu và cài đặt giải thuật CNN cho nhiều bài toán khác nhau hơn nữa.

KẾT LUẬN VÀ KIẾN NGHỊ

Kết quả đạt được

Về mặt lý thuyết, bài báo cáo đã trình bày được các nội dung sau:

* Tìm hiểu về mạng CNN.
* Tìm hiểu về bài toán tối ưu tổ hợp.
* Áp dụng bài toán đó vào nhận diện răng trong chụp X-Quang.
* Tìm hiểu, nghiên cứu, cài đặt thử nghiệm các thuật toán học sâu tiên tiến liên quan đến các bài toán phát hiện đối tượng, theo dõi đối tượng, đo lường khoảng cách.

Về mặt thực nghiệm, bài báo cáo đã thu được một số kết quả:

* Cài đặt thành công giải thuật nhận diện răng trong chụp X-Quang.
* Thực nghiệm giải thuật với các tham số và đưa ra đánh giá, nhận xét.
* Đã xây dựng và ứng dụng một bộ dữ liệu mới dành cho bài toán phát hiện chiếc răng.
* Kết hợp các sự phát minh mới trong sự phát triển của khoa học kĩ thuật hay AI nói riêng, đồng thời sử dụng các thuật toán khác áp dụng vào tình hình xã hội.
* Hệ thống nhận diện răng trong chụp X-Quang đã hoạt động tốt với độ chính xác trên 80% với các đối tượng hình ảnh răng trong thực tế.

Tuy nhiên, do còn nhiều hạn chế về kiến thức, kinh nghiệm và thời gian thực nghiệm, bài báo cáo còn một số thiếu sót sau:

* Thực nghiệm còn chưa đầy đủ.
* Bài báo cáo dừng ở mức tiếp cận vấn đề, chưa đi sâu giải quyết vấn đề.
* Hệ thống vẫn còn rất sơ sài chưa thực sự tối ưu, chưa có giao diện tinh chỉnh và sử dụng dễ dàng. Vì vậy nên nhóm rất mong được các thầy cô đóng góp thêm các ý kiến để nhóm có thể cải thiện hệ thống này, tiến tới ứng dụng thực tiễn trong tương lai.

Hướng phát triển trong tương lai

Bài báo cáo đã mang đến cách tiếp cận gần hơn với phương pháp giải quyết vấn đề nhận diện răng trong chụp X-Quang. Vì vậy trong tương lai, chúng em sẽ tiếp tục phát triển, đi sâu vào việc thử nghiệm với nhiều bài toán tối ưu khác và đồng thời, lên ý tưởng về việc giải quyết nhiều bài toán có hàm mục tiêu khác nhau để tạo ra một cách giải quyết cho yêu cầu giải đồng thời nhiều bài toán, nhiều tác vụ cùng một lúc của khoa học máy tính hiện đại.

TÀI LIỆU THAM KHẢO

1. Vittorio Murino, “Groups and Crowds: Detection, Tracking and Behavior Analysis of People Aggregations”, trong *VISIGRAPP Conference*, Rome, Italy, 2016.
2. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
3. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, UC Berkeley, 2014. Rich feature hierarchies for accurate object detection and semantic segmentation.
4. Ross Girshick, 2015. Fast R-CNN.
5. Bochkovskiy, A., Wang, C.-Y., & Mark Liao, H.-Y. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection.
6. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial Pyramid Pooling in Deep Convolutional.
7. Huang, G., Liu, Z., & Maaten, L. v. (2018). Densely Connected Convolutional Networks.
8. Huang, Z., & Wang, J. (2019). DC-SPP-YOLO: Dense Connection and Spatial Pyramid Pooling Based YOLO for Object Detection.
9. Hui, J. (2020). YOLOv4. Medium.
10. ODSC Medium, 2018. Overview of the YOLO Object Detection Algorithm.
11. Hoffmann, J., Navarro, O., Kastner, F., Janßen, B., & Hubner, M. (2017). A survey on CNN and RNN implementations. In PESARO 2017: The Seventh International Conference on Performance, Safety and Robustness in Complex Systems and Applications (No. 3).
12. Martello, S., Minoux, M., Ribeiro, C., & Laporte, G. (Eds.).(2011). Surveys in combinatorial optimization. Elsevier.
13. Oliveira, C. A., & Pardalos, P. M. (2005). A survey of combinatorial optimization problems in multicast routing. Computers & Operations Research, 32(8), 1953-1981.
14. Schrijver, A. (2005). On the history of combinatorial optimization (till 1960). Handbooks in operations research and management science, 12, 1-68.