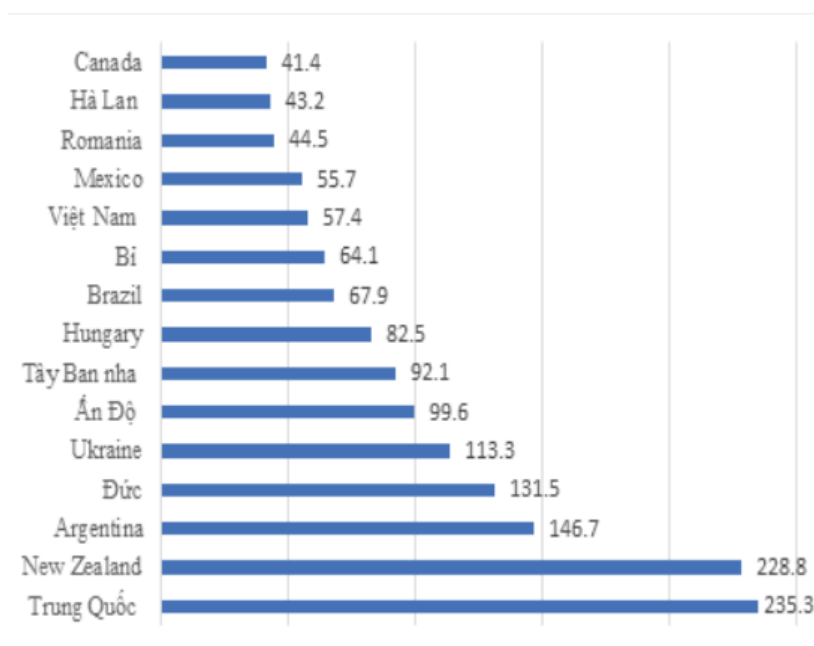


## CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

### 1.1 Giới thiệu chung về đề tài

Ong mật có vai trò rất quan trọng trong đời sống hàng ngày của con người. Chúng không chỉ cung cấp các sản phẩm có giá trị cao cả về dinh dưỡng và thương mại như mật ong, phấn hoa, sữa ong chúa và sáp ong mà còn có ảnh hưởng to lớn đối với ngành nông nghiệp. Các tài liệu nghiên cứu đã chỉ ra rằng, ong mật thụ phấn cho khoảng 35% các loại cây lương thực toàn cầu. Có thể khẳng định rằng, ong mật và nghề nuôi ong mật góp phần đáng kể trong phát triển nông nghiệp bền vững và tăng trưởng kinh tế của nhiều quốc gia trên thế giới. Biểu đồ 1 mô tả về kim ngạch xuất khẩu mật ong của 15 nước lớn nhất thế giới.



Hình 1.1. Biểu đồ kim ngạch xuất khẩu mật ong 15 nước lớn nhất thế giới (triệu USD)

Nhu cầu tiêu thụ mật ong và các sản phẩm liên quan hiện nay có xu hướng tăng lên nhanh chóng. Tuy nhiên sản lượng mật ong những năm gần đây không thể đáp ứng kịp nhu cầu ngày càng tăng này, thậm chí đang có xu hướng giảm. Nguyên nhân của tình trạng giảm này một phần là do việc thâm canh và lạm dụng thuốc trừ sâu trong nông nghiệp cũng như sự xuất hiện của nhiều loại dịch bệnh, sâu bệnh dẫn đến số lượng cá thể ong và đàn ong đang giảm xuống và sự biến mất nhanh chóng của nhiều vùng nuôi ong. Thực trạng này đặt ra một yêu cầu đối với các nhà nghiên cứu, những người nuôi ong trong việc nghiên cứu, phát triển, và ứng dụng các quy trình sản xuất hiện đại, các công nghệ tiên tiến trong tất cả các khâu của quá trình nuôi, từ nhân giống, chăm sóc, sản xuất tới tiêu thụ mật ong để hạn chế các yếu tố tiêu cực tác động, góp phần nâng cao sản lượng và giá trị sản phẩm.

Yêu cầu về nuôi ong chất lượng cao ngoài việc đảm bảo nguồn thức ăn đầy đủ (nguồn hoa) còn cần phải theo dõi giám sát đàn ong nhằm đạt được hiệu quả

cho mật lớn nhất. Việc kiểm tra, giám sát thường xuyên để nắm được thông tin về tình trạng sức khỏe đàn ong, tình hình dịch bệnh, số lượng ong trong đàn...là rất quan trọng. Các phương pháp theo dõi thủ công truyền thống phụ thuộc nhiều vào kinh nghiệm của người nuôi ong, tốn nhiều thời gian, công sức và không đem lại hiệu quả cao. Chẳng hạn, việc tháo dỡ bên ngoài để kiểm tra sẽ tạo nên sự căng thẳng, hoảng loạn trong đàn ong hay thậm chí có thể làm gián đoạn vòng đời của ong trong đàn. Ở nhiều vùng sản xuất, các tổ ong có thể đặt ở xa nơi sinh sống, dẫn đến việc phát sinh các chi phí di chuyển và vận chuyển. Những người nuôi ong cũng không thể thực hiện việc giám sát đàn ong một cách liên tục để phát hiện kịp thời các vấn đề phát sinh. Công nghệ IoT - AI đã được nghiên cứu áp dụng để thay thế cho các phương pháp giám sát truyền thống. Ý tưởng chung của giải pháp công nghệ này là tích hợp một hệ thống thiết bị IoT vào mỗi tổ ong, bao gồm các cảm ứng điện tử để thu thập các dữ liệu liên quan tới đàn ong, như nhiệt độ, độ ẩm, trọng lượng buồng ong, âm thanh phát ra, số lượng ong bay ra, bay vào trong một khoảng thời gian. Mỗi loại dữ liệu đều chứa đựng những thông tin quan trọng về sức khỏe đàn ong. Thu thập được các dữ liệu này và áp dụng các thuật toán AI để phân tích, xử lý chúng sẽ giúp người quản lý kịp thời cập nhật và biết được tình trạng của đàn ong một cách chính xác, hay có thể dự báo một số tình huống trong tương lai, từ đó đưa ra những biện pháp phòng ngừa hợp lý. Giải pháp công nghệ mới này tạo nên một giám sát từ xa hiệu quả, cho phép trích xuất thông tin quan trọng về trạng thái và hoạt động của bầy ong mà không cần kiểm tra trực tiếp, phát hiện kịp thời sự tác động của dịch bệnh, sâu bệnh hay các yếu tố gây hại đến đàn ong cũng như các vấn đề quản lý khác.

## **1.2 Mục tiêu**

Đề tài: **“Đếm và dự đoán số lượng ong ra vào tổ sử dụng cảm biến hình ảnh”** có mục đích như sau:

- Phát hiện và đếm số lượng ong trước cửa tổ từ hình ảnh.
- Theo vết ong.
- Nhận dạng một số hành vi của ong: Bay vào, bay ra, bò loanh quanh tổ.

## **1.3 Đối tượng và phạm vi nghiên cứu**

Tổng quan hệ thống bao gồm:

- Camera có lens dạng gương cầu lồi, có thể thu được hình ảnh ở góc rộng;
- Máy tính có thể chạy chương trình các thư viện cần thiết.

Phạm vi áp dụng: Đề tài có thể áp dụng cho các trang trại nuôi ong có thùng

## 1.4 Phương pháp và nội dung

Đề tài được thực hiện bằng phương pháp xử lý hình ảnh: Sử dụng mạng học sâu để phát hiện và đếm ong. Sử dụng các thuật toán để theo vết ong, sau đó phân tích đường theo vết để nhận dạng hành vi ong mật.

Cấu trúc của đồ án bao gồm:

**Chương 1: Tổng quan về đề tài.** Chương này sẽ trình bày vấn đề cần giải quyết, mục tiêu cần đạt được, hướng giải quyết và phạm vi nghiên cứu của đồ án.

**Chương 2: Các nghiên cứu liên quan đến bài toán phát hiện và theo vết ong.** Chương này trình bày các kỹ thuật phát hiện và theo vết ong trên ảnh/video.

**Chương 3: Cơ sở lý thuyết.** Nội dung chương này sẽ giới thiệu về những lý thuyết cơ bản trong học máy, học sâu cũng như các thuật toán sử dụng trong đồ án.

**Chương 4: Triển khai kỹ thuật, thử nghiệm và đánh giá kết quả.** Trình bày về quá trình thu thập dữ liệu và tổng quan về dữ liệu, quá trình huấn luyện, tính toán và các kết quả đạt được.

**Chương 5: Kết luận.** Tổng hợp các nội dung đã thực hiện của đề tài. Những vấn đề còn chưa giải quyết được và hướng phát triển trong quá trình nghiên cứu tiếp theo.

## CHƯƠNG 2. CÁC NGHIÊN CỨU LIÊN QUAN ĐẾN BÀI TOÁN PHÁT HIỆN VÀ THEO VẾT ONG

### 2.1 Các phương pháp phát hiện ong trên ảnh/video

Phát hiện đối tượng là một công nghệ thị giác máy tính và xử lý ảnh dùng để giải quyết bài toán phát hiện các trường hợp đối tượng ngữ nghĩa của một lớp nhất định (ví dụ như con người, tòa nhà hoặc ô tô) trong hình ảnh và video kỹ thuật số.

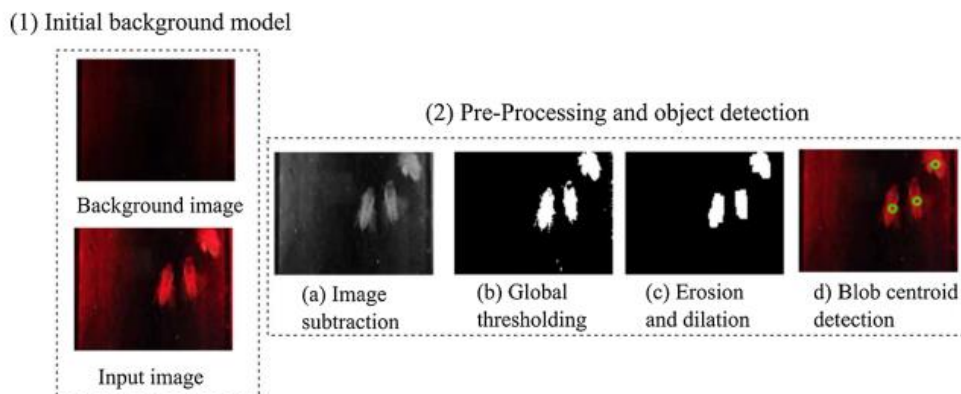
Các cách tiếp cận phát hiện đối tượng thường được chia làm hai nhánh:

- Hướng tiếp cận học máy: các phương pháp này đòi hỏi việc định nghĩa đặc trưng của đối tượng cần nhận dạng như đặc trưng Haar, SIFT, HOG, ... hoặc đặc trưng tự định nghĩa, sau đó phân lớp đặc trưng đó bằng các mô hình học máy cổ điển.
- Hướng tiếp cận học sâu: sử dụng các mạng nơron tích chập (SSD, Yolo, RetinaNet, ...).

Các phương pháp học máy truyền thống có ưu điểm là đơn giản và nhanh hơn, các yêu cầu về tài nguyên phần cứng cũng thấp hơn. Ngược lại, các phương pháp học sâu rất phức tạp nhưng linh hoạt hơn vì các mạng nơron có thể tự học ra các đặc trưng cần thiết của đối tượng mà không cần định nghĩa trước. Chúng cũng có thể nhận dạng được đối tượng ở nhiều hình thái như kích cỡ, góc quay, hướng chụp, độ che khuất, ... khác nhau; đồng thời cũng thường cho kết quả tốt hơn, tùy thuộc vào chất lượng của dữ liệu dùng để huấn luyện.

#### 2.1.1 Phương pháp trừ nền

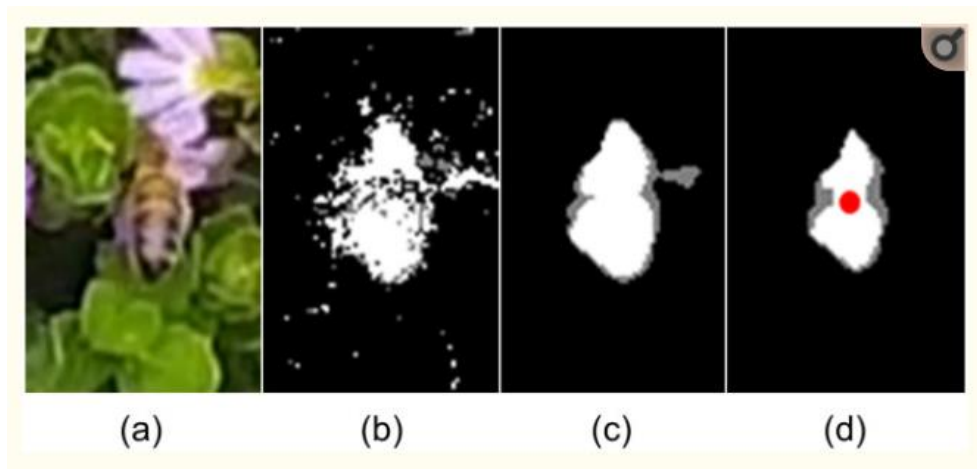
Thuật toán trừ nền [1][5] xác định mức xám của ảnh Video từ một camera tĩnh. Phương pháp trừ nền này khởi tạo một nền tham khảo với một số frame đầu tiên của Video đầu vào. Sau đó, nó trừ giá trị cường độ của mỗi điểm ảnh trong ảnh hiện thời cho giá trị tương ứng trong ảnh nền tham khảo.



Hình 2.1. Quá trình xử lý ảnh phát hiện đối tượng bằng phương pháp trừ nền [1]

Sau khi trừ nền thu được kết quả ở bước (a) sử dụng phát hiện cạnh để xác định các mục tiêu là ong (b). Do kết quả thu được còn nhiều nên sử dụng thêm một vài

kỹ thuật xử lý ảnh cơ bản như Erosion(xói mòn) và Dilation(giãn nở) để loại bỏ các đốm nhiễu (c).



Hình 2.2 Phát hiện ong sử dụng trừ nền và KNN [9]

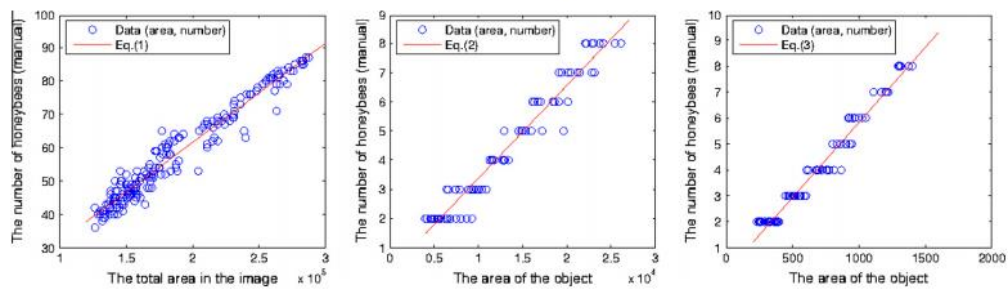
Video #	Window	Overlap (%)	BeePIV	Human Count	Error
5	17	12	75	75	0
6	17	18	231	239	8
7	12	25	71	74	3
8	10	40	360	361	1
10	12	25	368	357	11
11	12	1	374	373	1
12	17	12	339	348	9
13	8	25	174	176	2
14	12	25	208	208	0
15	10	40	456	456	0
16	10	40	263	270	7
17	12	17	157	154	3
18	12	1	294	294	0
19	17	24	14	17	3
20	12	1	65	60	5
21	8	25	419	432	13
22	10	20	101	101	0
23	12	30	249	247	2
24	12	30	168	168	0
25	17	12	90	90	0

Bảng 2.1. Bảng kết quả đếm ong sử dụng trừ nền và thuật toán PIV [10]

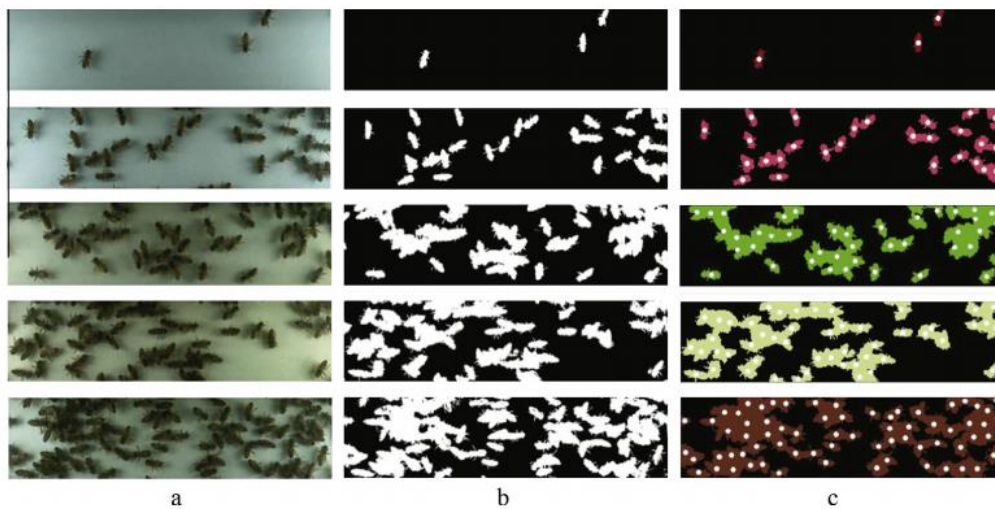
- **Ưu điểm:** Nhanh và đơn giản do việc khởi tạo background image chỉ đơn giản là việc lấy khung hình trước đó khi chưa có ong.
- **Nhược điểm:** Không đếm chính xác được số lượng ong với mật độ dày vì chúng chồng lên nhau

Để khắc phục nhược điểm của trừ nền các nhà nghiên cứu của trường đại học Aarhus [7] đã thiết lập một phương trình tuyến tính biểu diễn quan hệ giữa số lượng ong và tổng diện tích ảnh đã xám được phân đoạn.

G.J. Tu et al./Computers and Electronics in Agriculture 122 (2016) 10–18



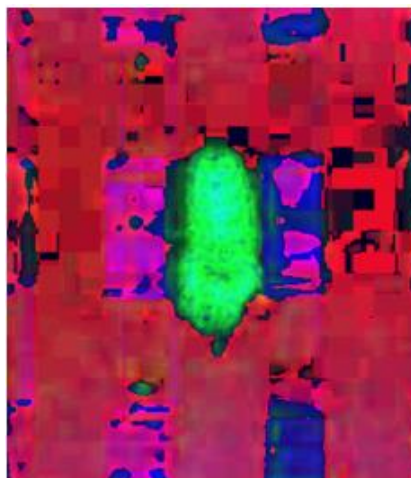
Hình 2.3 Kết quả huấn luyện đánh giá số lượng ong [7]



Hình 2.4 Kết quả dự đoán số lượng ong [7]

### 2.1.2 Phương pháp tách ong tử nền sử dụng độ bão hòa hình ảnh

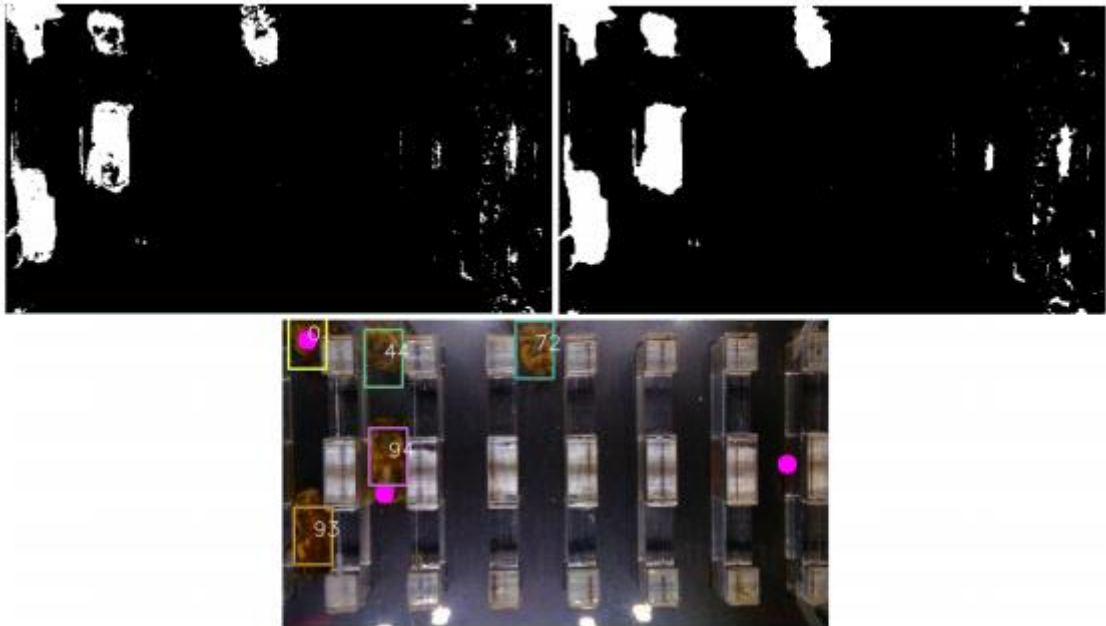
Đây là phương pháp xử lý ảnh cơ bản dùng để chuyển không gian màu RGB sang không gian màu HSV [2]. Minh họa như hình dưới đây, vùng màu xanh là hình ảnh hiển thị con ong trong không gian màu mới.



Hình 2.5. Hình ảnh ong biến đổi trong không gian màu HSV [2]



Sau khi chuyển ảnh sang không gian màu mới thì khoanh vùng phân loại ong và chuyển ảnh sang mức xám ta thu được kết quả như ở hình 2.3 góc trên bên trái. Sử dụng các thuật toán xử lý ảnh như đóng(Closing) và lấp(Holefilling) để loại bỏ nhiễu sẽ thu được kết quả như hình 2.3 góc trên bên phải.

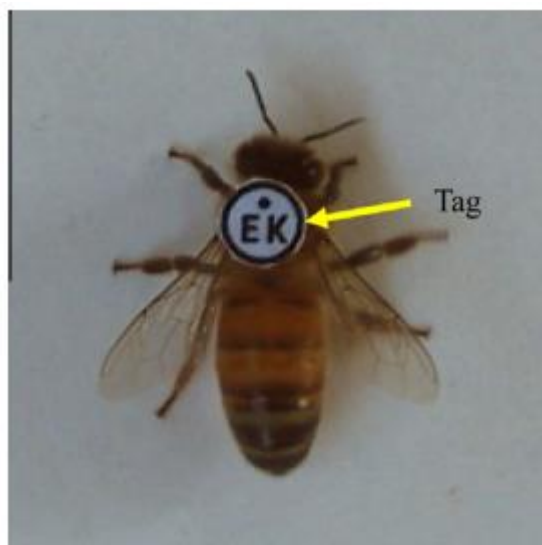


Hình 2.6. Kết quả thu được sau khi sử dụng phương pháp độ bão hòa hình ảnh [2]

- **Ưu điểm:** Đơn giản do chỉ sử dụng các thuật toán xử lý ảnh cơ bản.
- **Nhược điểm:** Kết quả có độ chính xác không cao vẫn còn nhiều nhiễu không loại bỏ được. Chỉ hiệu quả trong các chuồng ong được thiết kế một cách đặc biệt, không hiệu quả với đàn ong có mật độ lớn.

### 2.1.3 Phương pháp sử dụng tag

Mỗi con ong sẽ được gán một nhãn riêng biệt , sau đó sử dụng SVM nhận dạng ký tự để phát hiện ong [8].



Hình 2.7 Con ong được gán tag [8]



Hình 2.8. Nhận dạng tag [8]

**Nhận xét:** Phương pháp này có thể sử dụng được nhưng rất tốn thời gian để gán tag mỗi con ong với mật độ đàn ong dày đặc, biện pháp này chỉ khả thi trong quá trình nghiên cứu.

#### 2.1.4 Phương pháp sử dụng mạng học sâu

Việc áp dụng đột phát 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. Mô hình học sâu thông thường sẽ được thực hiện thông qua hai quá trình:

- Quá trình huấn luyện: Xác định các tham số của mạng nơ-ron thông qua bộ cơ sở huấn luyện dữ liệu.
- Quá trình kiểm nghiệm: Dùng mô hình đã được huấn luyện để đánh giá kết quả

Trong quá trình huấn luyện, các video đầu vào sẽ trải qua quá trình tiền xử lý để phù hợp với yêu cầu của mô hình. Sau đó thực hiện trích chọn đặc trưng biểu diễn hoạt động. Các đặc trưng này có thể được thiết kế thủ công bằng tay hoặc cũng có thể do những mô hình học sâu khác đã được huấn luyện đưa ra. Từ những đặc trưng đã được trích chọn này, chúng ta có thể dễ dàng huấn luyện một mô hình phù hợp với yêu cầu.



**Nhận xét:**

- Tốc độ nhanh, độ chính xác cao trong điều kiện mật độ lớn so với các phương pháp sử dụng xử lý ảnh cơ bản.
- Không yêu cầu phần cứng cao có thể giảm chi phí lắp đặt.
- Có thể nhận dạng được đối tượng ở nhiều hình thái như kích cỡ, góc quay, hướng chụp, độ che khuất, ... khác nhau.

**2.2 Các phương pháp theo vết ong trên ảnh/video**

Theo vết đối tượng là bài toán theo dõi một hoặc nhiều đối tượng chuyển động theo thời gian trong một video. Hiểu một cách đơn giản nhất, nó là bài toán ở mức độ cao hơn so với phát hiện, khi đối tượng được xử lý không đơn giản là một hình ảnh mà là một chuỗi các hình ảnh: video.

Theo vết đối tượng có thể chia thành 2 cách tiếp cận chính:

- Single Object Tracking (SOT): Single Object Tracking tập trung vào việc theo dõi một đối tượng duy nhất trong toàn bộ video. Và tất nhiên, để biết được cần theo dõi đối tượng nào, việc cung cấp một bounding box từ ban đầu là việc bắt buộc phải có.
- Multiple Object Tracking (MOT): Multiple Object Tracking hướng tới các ứng dụng có tính mở rộng cao hơn. Bài toán cố gắng phát hiện đồng thời theo dõi tất cả các đối tượng trong tầm nhìn, kể cả các đối tượng mới xuất hiện trong video. Vì điều này, MOT thường là những bài toán khó hơn SOT và nhận được rất nhiều sự quan tâm của giới nghiên cứu.

Một phương pháp Multiple Object Tracking cố gắng hướng đến việc theo dõi tất cả các đối tượng xuất hiện trong khung hình bằng việc phát hiện và gán định danh cho từng đối tượng. Bên cạnh đó, các ID đã được gán cho 1 đối tượng cần đảm bảo nhất quán qua từng frame. Một số khó khăn trong quá trình theo vết đối tượng là:

- Đối tượng bị che khuất 1 phần hoặc toàn bộ.
- Các đối tượng có quỹ đạo chuyển động giao nhau hoặc chồng chéo lên nhau.
- Chưa phát hiện tất cả đối tượng.

Có rất nhiều thuật toán đã được đưa ra nhằm giải quyết các bài toán theo vết ong như thuật toán trù nền, Kalman Filter, Optical Flow, Mean Shift, Cam Shift, SORT, Deep SORT.....

### 2.2.1 Mean Shift

Meanshift [2] hay Mode seeking là một thuật toán phổ biến chủ yếu được sử dụng để clustering hoặc những vấn đề liên quan tới unsupervised. Phân cụm KMeans là ứng dụng khai thác dữ liệu phân vùng n quan sát thành các cụm k. Mỗi quan sát thuộc về cụm với giá trị trung bình gần nhất. Trong phân cụm KMeans có thể chỉ định số lượng cụm được tạo, trong khi đó trong cụm MeanShift, số cụm được tự động phát hiện dựa trên số lượng mật độ trung tâm được tìm thấy trong dữ liệu. Meanshift tương tự như K-Means, nhưng thay vì sử dụng centroid để tính toán các tâm cụm thì Meanshift chuyển các điểm dữ liệu lặp đi lặp lại theo chế độ, đây là mật độ điểm dữ liệu cao nhất trung bình. Nó cũng được gọi là thuật toán tìm kiếm chế độ. Mục tiêu của thuật toán là tìm tất cả các chế độ trong phân phối dữ liệu đã cho. Khác với thuật toán K-mean thì Meanshift không yêu cầu phải xác định K nhóm từ trước.

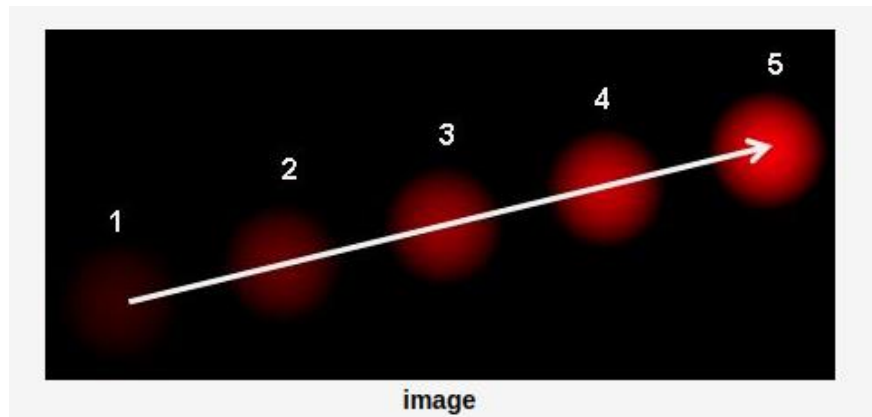
Giả sử trong một video khi mô hình phát hiện được đối tượng trong khung hình và trích xuất ra được các đặc trưng như màu sắc, kết cấu, histogram. Sau khi áp dụng thuật toán Meanshift thì sẽ có một idea chung về phân phối tương đối của các đối tượng ở trạng thái hiện tại. Sau đó có một frame tiếp theo thì các phân phối này có thể thay đổi do sự chuyển động của đối tượng thì meanshift sẽ tìm kiếm theo chế độ vào do đó theo dõi đối tượng.

- **Ưu điểm:** Đơn giản, không yêu cầu bất kì đào tạo nào, theo vết không cần bước phát hiện ong.
- **Nhược điểm:** Theo vết dựa vào đặc trưng màu sắc, các con ong đều giống nhau nên không thể theo vết đàn ong với mật độ đông được.

### 2.2.2 Optical Flow

Ong được theo dõi sử dụng các image brightness variations theo không gian-thời gian ở các cấp độ pixel. Trong thuật toán thì sẽ tập trung vào việc thu được vectơ dịch chuyển cho các đối tượng qua các frame [5].

1. Brightness consistency: Độ sáng xung quanh một vùng nhỏ được cho là gần như không đổi, mặc dù vị trí của vùng có thể thay đổi.
2. Spatial coherence: Các điểm lân cận trong cùng một khung cảnh thường sẽ cùng thuộc một bề mặt do đó sẽ có những sự chuyển động tương tự.
3. Temporal persistence: Các điểm thường sẽ có sự chuyển động dần dần. Khi các tiêu chí trên được thoả mãn thì sẽ sử dụng phương pháp Lucas-Kanade để có được phương trình vận tốc của các điểm nhất định được theo dõi (thường ở đây là những điểm dễ phát hiện). Sử dụng phương trình cùng một số phương pháp dự đoán, thì một đối tượng sẽ được theo dõi trong toàn bộ video.



Hình 2.9 Minh họa chuyển động của ong trong 5 khung hình liên tiếp [5]



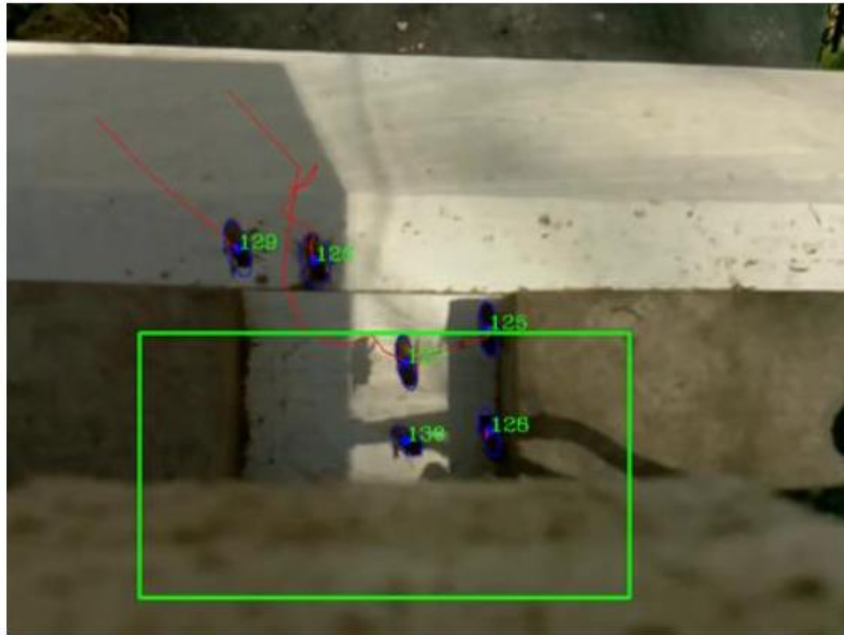
Hình 2.10 Kết quả theo vết sử dụng Optical Flow [5]

- **Ưu điểm:** Đơn giản
- **Nhược điểm:** Độ chính xác thấp, khi mật độ đàn ong đông, ong chồng chéo lên nhau nhiều vectơ dịch chuyển sẽ bị nhiễu gây đổi ID nhiều.

### 2.2.3 Kalman Filter

Bộ lọc Kalman (Kalman Filter)[1][5] là một mô hình Linear-Gaussian State Space Model, được giới thiệu lần đầu năm 1960 và ứng dụng trong rất nhiều lĩnh vực khác nhau: Xe tự lái, thực tế ảo, kinh tế lượng, theo vết, điều khiển tối ưu, ...

Trong theo vết đối tượng, kalman filter được biết đến nhiều nhất với vai trò dự đoán các trạng thái của đối tượng hiện tại dựa vào các track trong quá khứ và update lại các detection sau khi đã được liên kết với các track trước đó.

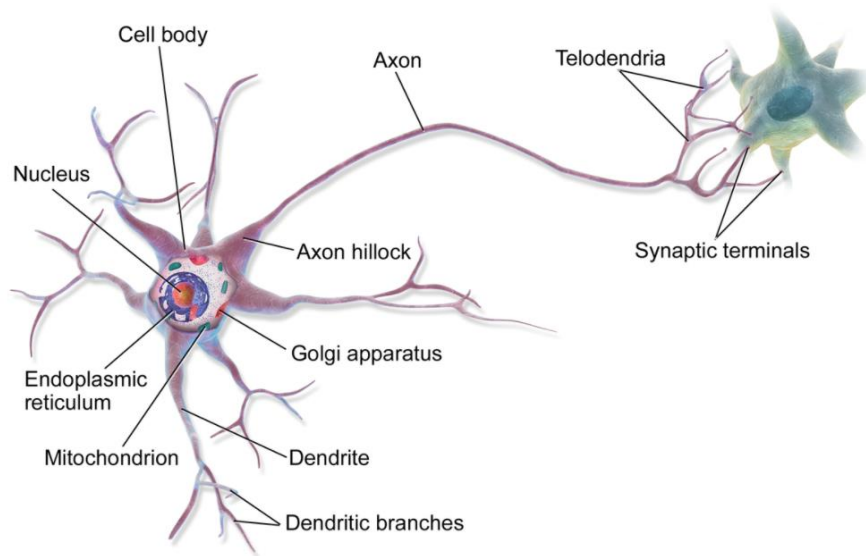


Hình 2.11. Kết quả theo vết ong sử dụng thuật toán trừ nền và kalman filter [5]

- **Ưu điểm:** Kết quả theo vết tương đối tốt khi phân phát hiện đối tượng hoạt động hiệu quả.
- **Nhược điểm:** Khi mật độ đàn ong đông liên kết giữa các detection và track gặp khó khăn dẫn đến việc các con ong bị đổi ID.

## CHƯƠNG 3. CƠ SỞ LÝ THUYẾT CHUNG

### 3.1 Tổng quan về mạng học sâu

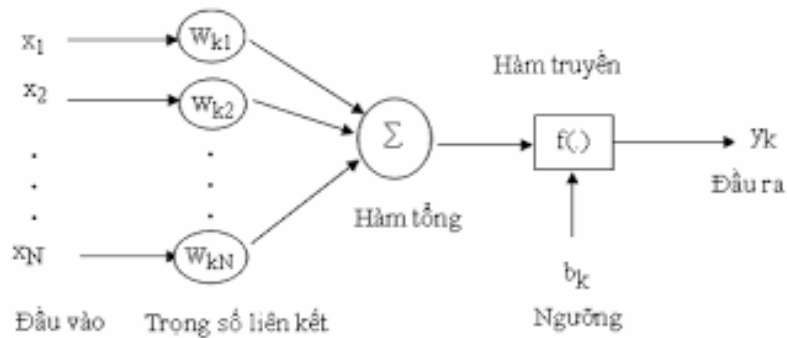


Hình 3.1: Mạng nơ-ron sinh học [11]

Ý tưởng xây dựng các mô hình mạng nơ-ron nhân tạo bắt nguồn từ việc khám phá ra các cơ chế hoạt động đơn giản của mạng nơ-ron sinh học. Trong hệ thống thần kinh sinh học, nơ-ron là tế bào sống và còn là đơn vị lưu trữ cơ bản trong bộ não con người. Các nhà nghiên cứu đã tìm cách chuyển đổi những hiểu biết về cách thức hoạt động của các tế bào thần kinh sinh học thành các mô hình mạng nơ-ron nhân tạo. Một mạng bao gồm rất nhiều các nơ-ron đơn lẻ, được nối với nhau, phối hợp hoạt động với nhau để lưu trữ những kiến thức, kinh nghiệm để nhằm tái sử dụng.

Cấu trúc mạng nơ-ron nhân tạo gồm có 3 phần chính:

- Đầu vào là tập hợp các giá trị, được nhân với các trọng số tương ứng thể hiện độ đóng góp của nó vào nơ-ron;
- Một bộ cộng dùng để tổng hợp các tín hiệu đầu vào tại mỗi nơ-ron và gửi kết quả đi tiếp;
- Một hàm kích hoạt (activation function) dùng để đưa các tín hiệu đầu ra của nơ-ron vào một miền giá trị nhất định hoặc vào một tập hợp các giá trị cố định.

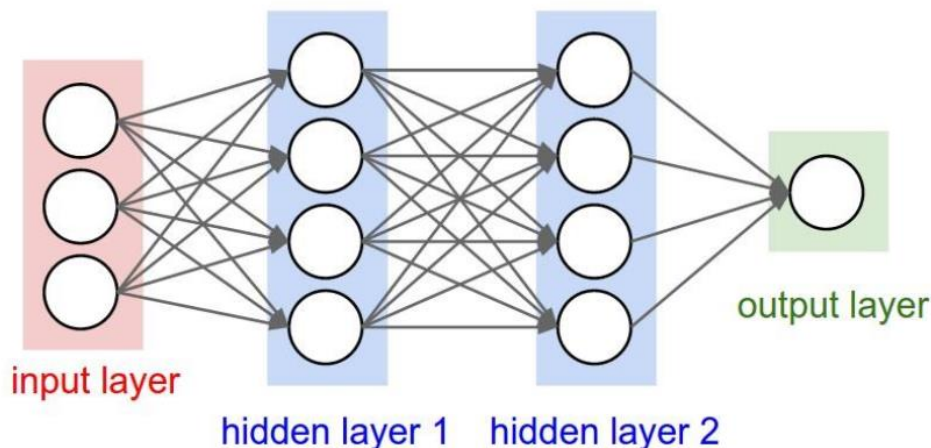


Hình 3.2: Mô hình mạng nơ-ron nhân tạo

Thực nghiệm đã cho thấy số lượng các lớp ẩn càng lớn thì độ chính xác càng cao, đồng thời làm tăng độ phức tạp của mô hình, mỗi lớp sẽ trích xuất ra các đặc trưng khác nhau của dữ liệu.

### 3.2 Kiến trúc mạng cơ bản

Mạng nơ-ron là sự kết hợp của các tầng perceptron hay còn được gọi là perceptron đa tầng (multilayer perceptron) như hình vẽ bên dưới:



Hình 3.3 Minh họa mạng neural với hai lớp ẩn.

Một mạng nơ-ron sẽ có 3 lớp chính:

- Lớp đầu vào (Input layer): Là lớp bên trái cùng của mạng thể hiện cho các đầu vào của mạng.
- Lớp ra (Output layer): Là lớp bên phải cùng của mạng thể hiện cho các đầu ra của mạng.
- Lớp ẩn (Hidden layer): Là lớp nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng.

Một mô hình mạng nơ-ron chỉ có 1 lớp đầu vào, 1 lớp đầu ra nhưng có thể có rất nhiều lớp ẩn. Trong mạng nơ-ron, mỗi nút mạng là một sigmoid nơ-ron nhưng hàm kích hoạt của chúng có thể khác nhau. Tuy nhiên trong thực tế người ta thường để chúng cùng dạng với nhau để tính toán cho thuận lợi.

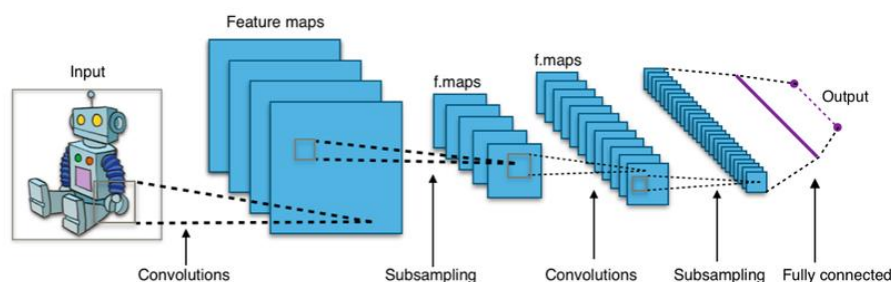
Ở mỗi lớp, số lượng các nút mạng (nơ-ron) có thể khác nhau tùy thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các lớp ẩn có



số lượng nơ-ron bằng nhau. Ngoài ra, các nơ-ron ở các lớp thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (fully-connected network). Khi đó ta có thể tính được kích cỡ của mạng dựa vào số lớp và số nơ-ron.

### 3.3 Mạng neural tích chập

Việc sử dụng mạng nơ-ron không những có số lượng tham số lớn mà còn không đánh giá được mối quan hệ của các điểm lân cận đặc biệt là bài toán xử lý ảnh, ví dụ như bài toán xác định đối tượng trong ảnh, một đối tượng có thể chiếm một vùng lớn trong ảnh, nếu ta chỉ sử dụng mạng nơ-ron để đánh giá các pixel riêng biệt thì chắc chắn sẽ không đạt được độ chính xác, ngược lại với mạng tích chập, việc học trên một vùng ảnh sẽ cho ta kết quả tốt hơn. Cấu trúc của mạng tích chập như hình 10



Hình 3.4: Mô hình mạng tích chập [3]

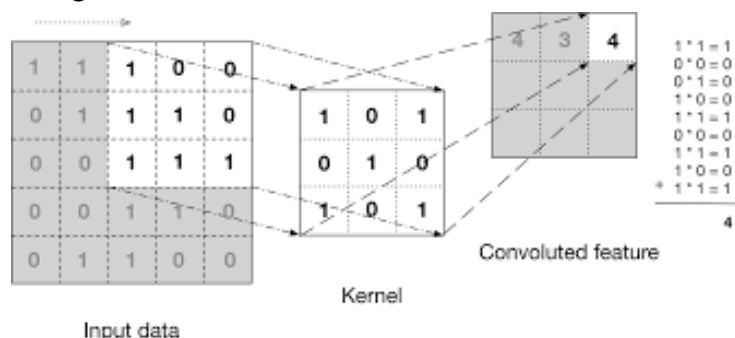
Mạng tích chập cũng được cấu thành từ các lớp, tuy nhiên các lớp được cấu thành không phải từ các nốt mà từ các bộ lọc.

Mạng tích chập thông thường được cấu thành từ các lớp:

- Lớp tích chập;
- Lớp tổng hợp;
- Lớp kết nối đầy đủ hoặc lớp tổng hợp trung bình toàn cục.

#### a) Lớp tích chập

Lớp tích chập là lớp chủ đạo trong mạng tích chập, nó thực hiện hầu hết các tính toán quan trọng của mô hình.



Hình 3.5: Mô hình lớp tích chập [3]

Các tham số của lớp tích chập [14] bao gồm tập các bộ lọc có thể học. Mọi bộ lọc có kích thước nhỏ về mặt không gian (dọc theo chiều rộng và chiều cao), nhưng có thể trượt qua toàn bộ khu vực của ảnh.

Ví dụ: một bộ lọc có kích thước 5x5x3 (tức là chiều rộng và chiều cao là 5 pixel và 3 vì hình ảnh có độ sâu 3, các kênh màu) được trượt qua ảnh từ trái qua phải,

từ trên xuống dưới. Tại mỗi vị trí sẽ thực hiện phép nhân tích chập hay nhân từng giá trị trong bộ lọc với các giá trị trên ảnh tại vị trí bộ lọc đè lên tương ứng như hình 9, đầu ra là tổng trung bình đầu ra của 3 bộ lọc, quá trình này được lặp đi lặp lại cho đến khi bộ lọc trượt qua được hết khu vực của ảnh.

Việc sử dụng lớp tích chập giúp mô hình có trích xuất được đặc trưng của một vùng ảnh, ngoài ra việc chia sẻ trọng số (các giá trị trong bộ lọc) sẽ giúp giảm đi tham số của mô hình, tránh được những hiện tượng như overfit hay vanishing gradient.

Nếu ảnh vào có kích thước  $W_1 \times H_1 \times K$ , bộ lọc có kích thước  $F \times F$ , hệ số trượt là  $S$  và số padding là  $P$  thì kích thước đầu ra là  $W_2 \times H_2 \times D_2$  được xác định như sau:

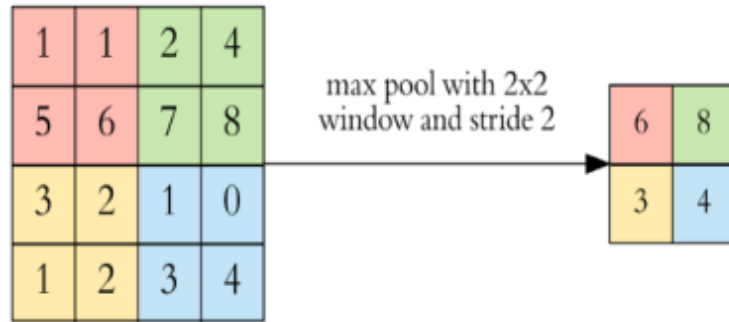
$$W_2 = \frac{W_1 - F + 2P}{S} + 1 \quad (3.1)$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1 \quad (3.2)$$

$$D_2 = K \quad (3.3)$$

#### b) Lớp tổng hợp

Để giảm các trọng số cần tính toán của mô hình và giúp mô hình dễ hội tụ hơn thì người ta tìm sử dụng lớp tổng hợp, trong đó lớp tổng hợp giá trị lớn nhất được sử dụng nhiều nhất.



Hình 3.6: Cấu tạo lớp tổng hợp [3]

Ý tưởng thực hiện cũng sử dụng một bộ lọc có kích thước nhỏ, rồi trượt trên ảnh gốc, tại vùng ảnh bộ lọc đè lên thì chọn giá trị lớn nhất.

Với ma trận đầu vào có kích thước  $W'_1 \times H'_1 \times D'_1$ , bộ lọc có kích thước  $F' \times F'$  với bước trượt  $S'$  pixel thì ta được ma trận đầu ra  $W'_2 \times H'_2 \times D'_2$  trong đó:

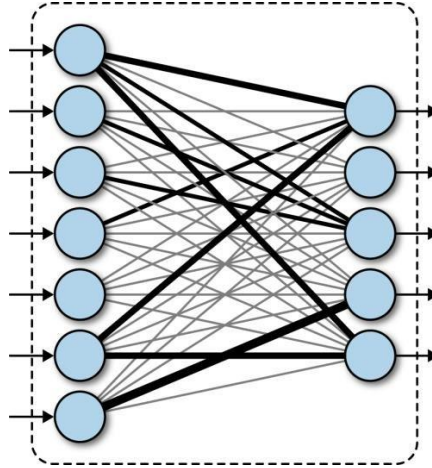
$$W'_2 = \frac{W'_1 - F'}{S'} + 1 \quad (3.4)$$

$$H'_2 = \frac{H'_1 - F'}{S'} + 1 \quad (3.5)$$

$$D'_2 = D'_1 \quad (3.6)$$

c) Lớp kết nối đầy đủ

Lớp này có cấu tạo từ các nôt giống như một lớp ở mạng nơ-ron, đầu vào của lớp này thông thường là đầu ra của lớp tổng hợp, các phân tử được duỗi thành dạng vector 1 chiều, mỗi nôt biểu diễn xác suất mà phân tử đó thuộc về.



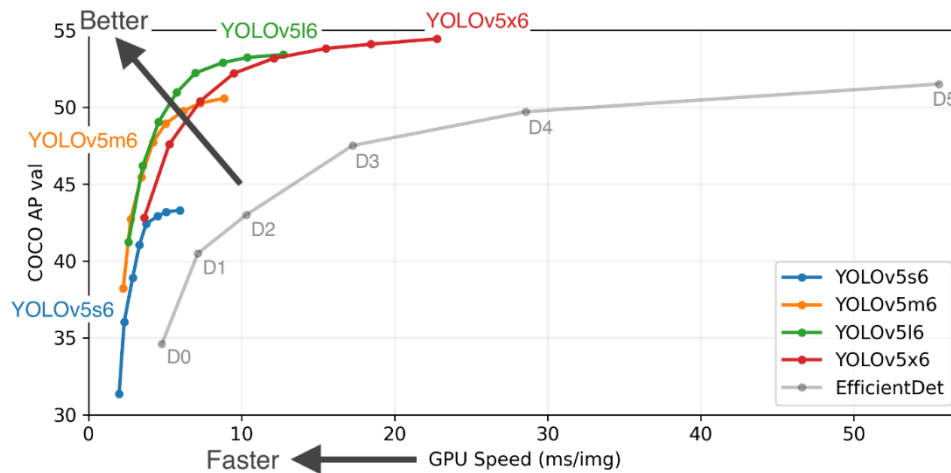
Hình 3.7: Cấu tạo lớp đầy đủ [12]

Hiện nay, các mô hình hiện đại sử dụng lớp tổng hợp trung bình toàn cục thay thế cho lớp kết nối đầy đủ, lý do là do lớp kết nối đầy đủ chứa quá nhiều tham số khi kích thước của đầu vào là quá lớn, về cơ bản lớp tổng hợp trung bình toàn cục giống như lớp tích chập nhưng có kích thước 1x1.

### 3.4 Mô hình phát hiện ong

#### 3.4.1 Mô hình YOLOv5

YOLO (you only look one) – bạn chỉ nhìn một lần, tính đến thời điểm hiện tại, có một số thuật toán xác định đối tượng có khả năng chạy realtime như YOLO, SSD, ... Tất cả các thuật toán được xây dựng dựa trên cấu trúc mạng nơ-ron tích chập



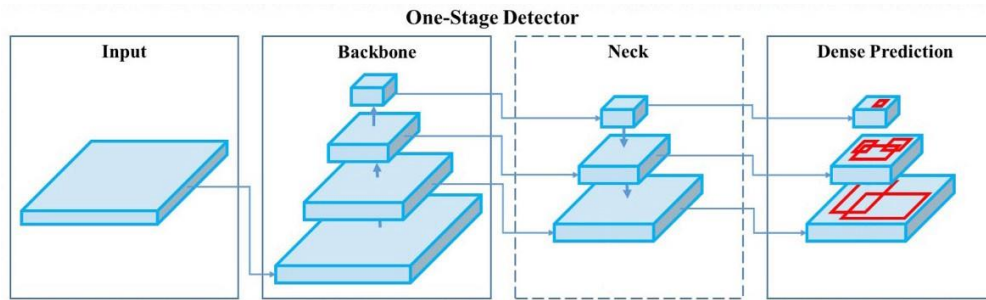
Hình 3.8: Bảng so sánh các mô hình YOLOv5

Như hình trên ta có thể thấy YOLOv5s đang có khả năng xử lý nhanh nhất và độ chính xác khá là cao. So với mô hình YOLOv4, phiên bản thứ 5 đã có những thay đổi lớn về mặt tốc độ và độ chính xác, YOLOv5 có thể được huấn luyện trên một GPU với số lượng batch size nhỏ, thuật toán có thể được huấn luyện rất nhanh và chính xác khi sử dụng card đồ họa 1080Ti hoặc 2080Ti GPU của hãng NVIDIA, điều mà gần như không thể làm với các model state-of-art hiện tại.

Phiên bản thứ 5 này có độ chính xác cao hơn 10% và tốc độ FPS cao hơn 12% so với phiên bản thứ 4. Tóm lại, YOLOv5 có một số cải tiến hơn so với YOLOv4 như sau:

- Phát triển một mô hình detection vừa đạt được độ chính xác cao vừa cho tốc độ chạy realtime;
- Áp dụng các kỹ thuật như hiện đại như Bag-of-Freebies và Bag-of-Special trong lúc huấn luyện mô hình;
- Thay đổi một số cấu trúc giúp phù hợp với việc huấn luyện trên GPU như CBN, PAN, SAM, ...

## b) Cấu trúc của mô hình YOLOv5



Hình 3.9: Cấu trúc mô hình YOLOv5 [3]

Về cơ bản, YOLOv5 bao gồm 4 phần:

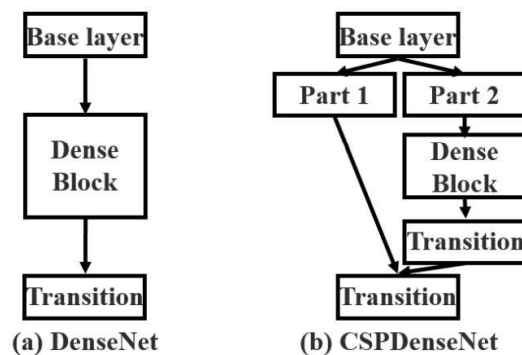
- **Input**: ảnh đầu vào;
- **Backbone**: giúp trích xuất ra các thuộc tính đặc trưng của ảnh, được huấn luyện trên tập dữ liệu phân loại ảnh như ImageNet. Một số backbone đạt được độ chính xác cao như VGG16, ResNet-50, SpineNet, CSPResNeXt50, CSPDarknet53, ...;
- **Neck**: giúp tìm ra các vùng có chứa đối tượng, một số thuật toán được sử dụng như FPN (Feature pyramid network), PAN, NAS-FPN, BiFPN, ...;
- **Heads**: Là một mạng có nhiệm vụ xác định vị trí của các hộp (các hộp bao quanh đối tượng) và xác suất xuất hiện của đối tượng ở trong hộp, thường sử dụng như YOLO, SSD, RetinaNET.

Thuật toán được hoạt động như sau: ảnh đầu vào được cho đi qua khối backbone để thu được các ma trận đặc trưng, khối neck giúp tinh chỉnh ma trận đặc trưng thêm cô đọng và mang nhiều thông tin hơn, khối head sẽ sử dụng các thuật toán để xác định đối tượng và phân loại đối tượng trong ảnh.

### Backbone

Ở phiên bản trước, backbone được sử dụng là Darknet53 tuy nhiên ở phiên bản này, tác giả đã sử dụng CSPDarknet53 [16], mô hình này giúp chạy tốt hơn trên GPU và cho độ trích xuất thông tin tốt hơn.

Mạng sử dụng cấu trúc CSP (Cross-Stage-Partial connection), ý tưởng là phân chia lớp hiện tại thành 2 phần, 1 phần sẽ cho đi qua khối tích chập và phần còn lại thì không, kết quả là sự kết hợp của 2 đầu ra của 2 phần, chúng ta có thể xem ví dụ với DenseNet [17] như hình 3.10:



Hình 3.10: Ví dụ về cấu trúc của mạng CSP

Tương tự áp dụng cho mạng Darknet, ta có mạng Darknet như hình 3.11:

	Type	Filters	Size	Output
	Convolutional	32	$3 \times 3$	$256 \times 256$
	Convolutional	64	$3 \times 3 / 2$	$128 \times 128$
1x	Convolutional	32	$1 \times 1$	
	Convolutional	64	$3 \times 3$	
	Residual			$128 \times 128$
	Convolutional	128	$3 \times 3 / 2$	$64 \times 64$
2x	Convolutional	64	$1 \times 1$	
	Convolutional	128	$3 \times 3$	
	Residual			$64 \times 64$
	Convolutional	256	$3 \times 3 / 2$	$32 \times 32$
8x	Convolutional	128	$1 \times 1$	
	Convolutional	256	$3 \times 3$	
	Residual			$32 \times 32$
	Convolutional	512	$3 \times 3 / 2$	$16 \times 16$
8x	Convolutional	256	$1 \times 1$	
	Convolutional	512	$3 \times 3$	
	Residual			$16 \times 16$
	Convolutional	1024	$3 \times 3 / 2$	$8 \times 8$
4x	Convolutional	512	$1 \times 1$	
	Convolutional	1024	$3 \times 3$	
	Residual			$8 \times 8$
	Avgpool		Global	
	Connected		1000	
	Softmax			

Hình 3.11: Cấu trúc mạng Darknet [3]

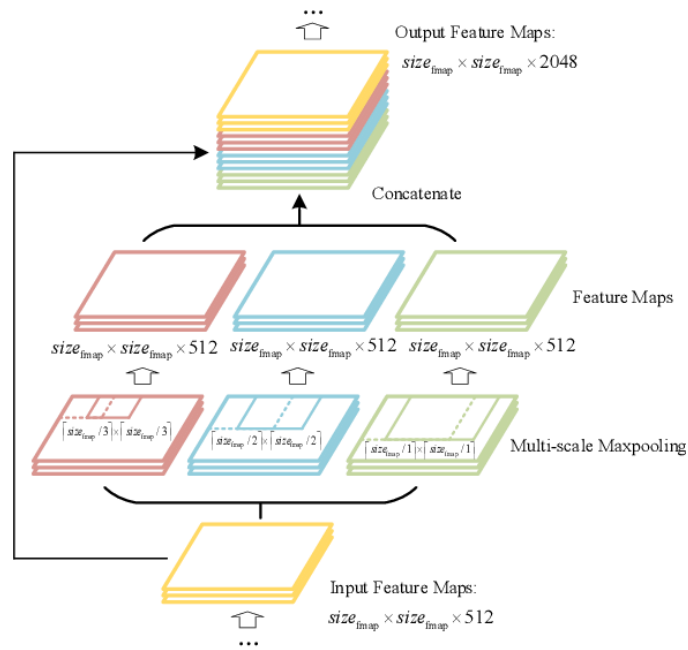
Mạng này tạo ra nhiều mức kích thước của đầu ra, giúp mang thông tin về ngữ nghĩa cao hơn khi mạng chứa nhiều lớp hơn. Điều này cũng quan trọng khi sẽ được sử dụng ở phần sau của mô hình.

### Neck

Khối này được sử dụng để trích xuất ra các lớp đặc trưng có kích thước khác nhau.

Vì YOLOv5 cho phép đầu vào có các kích thước khác nhau nên cần phải có một lớp để đưa các lớp đặc tính về cùng 1 hình dáng, Spatial Pyramid Pooling (SPP) được sử dụng để làm điều đó. Cấu trúc mạng như hình 3.12:





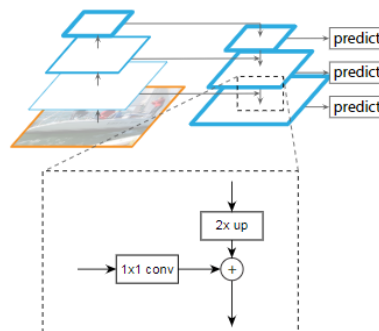
(b)

Hình 3.12: Cấu trúc SPP [3]

ở phiên bản YOLOv4 thì mô hình sử dụng FPN [3] (feature pyramid network). Cấu trúc mô hình như hình 18. Về cơ bản thì thuật toán đi từ dưới lên và kết hợp giữa đi từ trên xuống, mạng giúp có thể nhận diện được đối tượng với các kích thước khác nhau. Như trên hình 16 nếu ta gọi các lớp ở bên trái từ dưới lên lần lượt là F1, F2, F3, F4 và các lớp bên ở bên phải từ trên xuống lần lượt là P4, P3, P2 thì ta có:

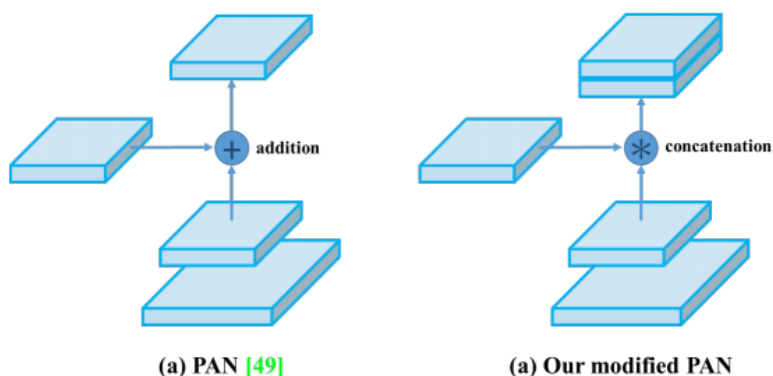
- F4 thực chất sẽ là P4 và sẽ được sử dụng để xác định đối tượng ở trong đó;
- P4 sử dụng bộ upsampling để đưa kích thước tăng lên gấp đôi, cộng với lớp F3 sẽ cho ra lớp P3, ta cũng thực hiện bước xác định đối tượng trên P3;
- Tương tự lớp P3 sẽ được upsampling, cộng với lớp F2 sẽ tạo ra lớp P2, ta lại thực hiện bước xác định đối tượng trên P2.

Kết quả là ta có thể có được các hộp cho các đối tượng có kích thước khác nhau, từ nhỏ đến lớn. Lớp P4 có kích thước nhỏ nên ta chỉ xác định được các đối tượng lớn, lớp P2 có kích thước lớn hơn sẽ cho phép ta xác định các đối tượng có kích thước nhỏ hơn.



Hình 3.13: Cấu trúc FPN [18]

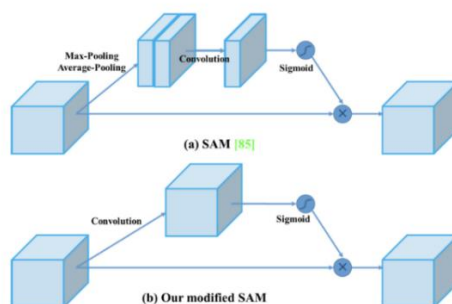
Ở phiên bản thứ 5, YOLOv5 sử dụng PANet (Path Aggregation Network). Ý tưởng là kết hợp thông tin để có độ chính xác cao nhất.



Hình 3.14: Cấu trúc PAN [3]

Ở phiên bản PAN gốc thì đầu ra từ các lớp backbone và các lớp cho vào cùng kích thước sẽ cộng vào với nhau, tuy nhiên ở bản dùng trong YOLOv5 thì các lớp được gán với nhau thay vì cộng, điều đó làm kích thước mạng nó tăng lên.

Một công nghệ khác được sử dụng ở đây nữa là Spatial Attention Module (SAM), ở đây mô hình tận dụng phương thức chú ý [19]. Thuật toán chỉ tập trung vào các phần đặc biệt của đầu vào, các phần này sẽ có ảnh hưởng lớn đến mô hình. Cấu trúc như trên hình 3.15

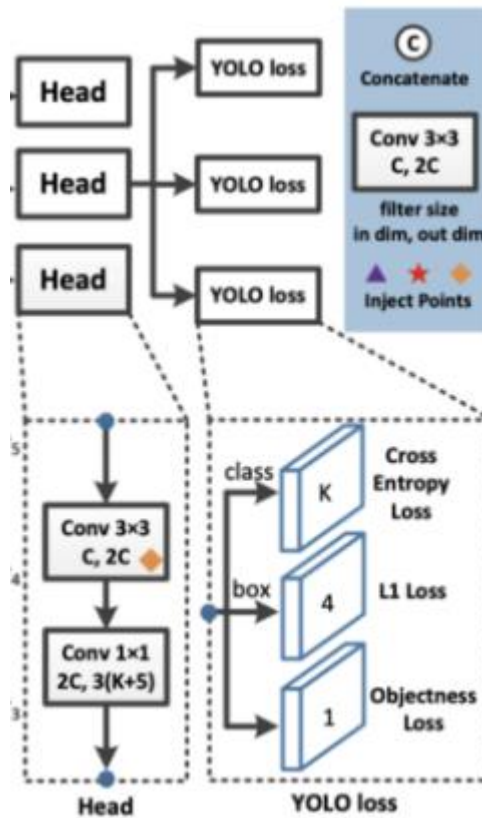


Hình 3.15: Cấu trúc thuật toán SAM [3]

## Head

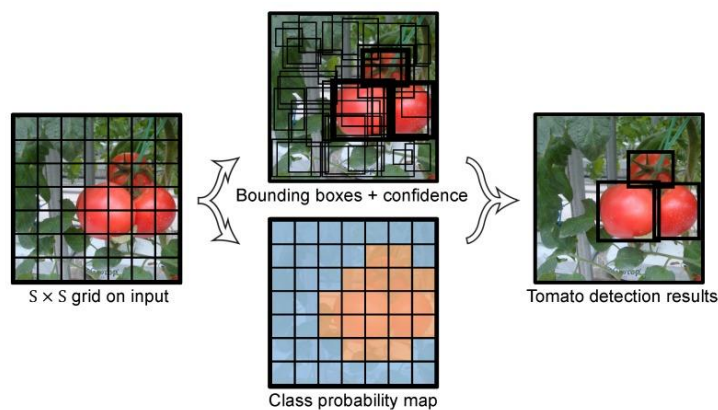
Khối này có nhiệm vụ thực sự thực hiện chức năng xác định vị trí của đối tượng và nhãn của đối tượng. YOLOv5 sử dụng thuật toán YOLO để thực hiện.

Head được chia làm 2 phần riêng biệt. Một phần để xác định tọa độ và một phần để xác định nhãn. Như hình 3.16 bạn có thấy cấu trúc của khối head.



Hình 3.16: Cấu trúc Head [3]

Hầu hết các thuật toán xác định đối tượng đều cần phải sử dụng một cửa sổ trượt qua các vùng của ảnh để xác định xem có đối tượng hay không. Điều đó rất tốn thời gian để thực hiện.



Hình 3.17: Ảnh cà chua được xác định sử dụng thuật toán YOLO [20]

YOLO sẽ chia ảnh đầu vào thành  $S \times S$  ô. Ví dụ như ảnh trên hình 25 được chia thành  $7 \times 7$ . Tâm của object sẽ rơi vào tâm của ô và ô này có trách nhiệm xác định đối tượng.

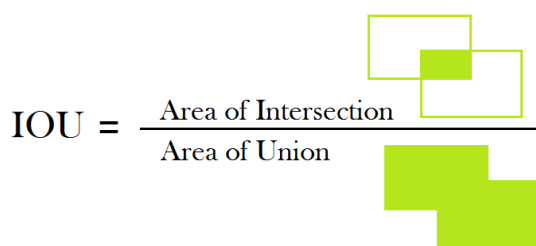
YOLO sẽ phải phân loại và xác định vị trí cho mỗi  $7 \times 7 = 49$  ô. Mỗi ô chỉ có thể phân loại và xác định cho 1 đối tượng nên dẫn đến YOLO có 1 số vấn đề sau:

- Vì sử dụng  $7 \times 7 = 49$  ô nên chúng ta chỉ có thể xác định được tối đa là 49 đối tượng;
- Nếu 1 ô có nhiều hơn 1 đối tượng thì YOLO không thể xác định được tất cả đối tượng  $\Rightarrow$  bị bỏ sót đối tượng;

- Mỗi đối tượng có thể nằm trên nhiều ô vì thế đối tượng sẽ được xác định nhiều hơn 1 lần (giống như ô tô σ trên hình nằm trong 15 ô nhỏ). Vấn đề này được giải quyết bằng non-max suppression (NMS). Chúng ta sẽ tìm hiểu nó ở phần dưới.

Để giải quyết vấn đề đó định nghĩa anchor ra đời. Với mỗi ô trong số 49 ô sẽ dự đoán B hộp (YOLO chọn B=3), và với mỗi hộp thì model phải đưa ra confidence score (độ chắc chắn có đối tượng trong hộp). Dùng chỉ số này sẽ giúp ngăn chặn mô hình dự đoán các cảnh khác đối tượng. Nếu không có đối tượng trong ô thì chỉ số này bằng không.

Ngoài ra YOLO đưa ra một định nghĩa IoU (intersec over union). IoU là tỉ số giữa diện tích phần giao giữa 2 hộp chia cho tổng diện tích hợp giữa 2 hộp. Tỉ số này phải  $\geq 0$  và  $\leq 1$ . Hình 22 sẽ giúp bạn có cái nhìn trực quan hơn



$$IOU = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

Hình 3.18: Công thức IoU [11]

Vậy tại sao chúng ta cần IOU? Bởi vì sẽ có rất nhiều hộp được dự đoán. Chúng ta cần xác định IOU để biết có bao nhiêu phần trăm hộp sẽ trùng với hộp thực tế. Nếu IOU cao thì đồng nghĩa xác suất có đối tượng xuất hiện trong hộp là cao và ngược lại. Ta sẽ loại bỏ đi nhưng hộp có confidence thấp.

Những hộp có hệ số confidence lớn hơn 0.7 được gán là positive hộp, ngược lại những hộp có confidence nhỏ hơn 0.3 thì được gán là negative hộp, positive hộp và negative hộp được sử dụng để huấn luyện mô hình.

Ngoài việc xác định confidence score thì model cần đưa ra 4 chỉ số(x,y,w,h) lần lượt là vị trí tâm, chiều dài, chiều rộng của hộp cần dự đoán. Chú ý:  $0 < (x,y,w,h) < 1$ .

Nếu một ảnh đưa vào được chia nhỏ thành lưới có kích thước SxS thì đầu ra sẽ có kích thước:

$$S \times S \times (B * 5 + C) \quad (3.7)$$

trong đó:

- B là số anchor hộp;
- 5 = 4 (tạo độ x, y, w, h) + 1 (confidence score).

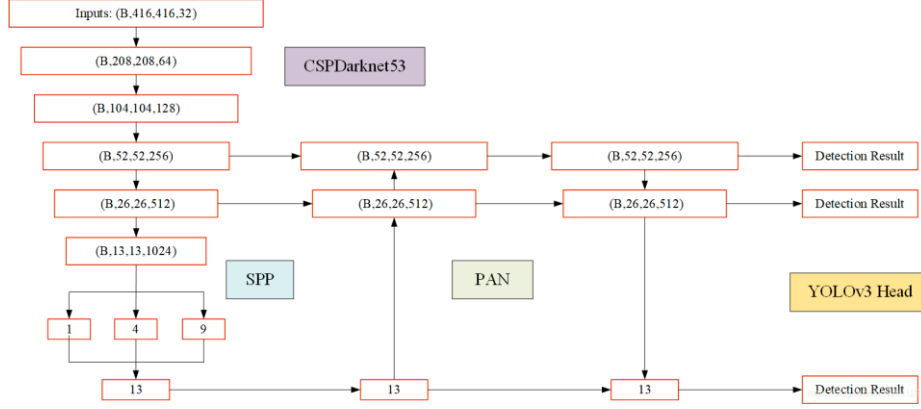
Confidence score được tính theo công thức

$$Pr(obj) * IoU \quad (3.8)$$

Trong đó Pr(obj) là xác suất có đối tượng trong hộp hay không

$$\text{Pr}(\text{obj}) = \begin{cases} 1 & \text{nếu tồn tại đối tượng} \\ 0 & \text{nếu không tồn tại đối tượng} \end{cases}$$

Trước khi đi vào tìm hiểu hàm mất mát của thuật toán, chúng ta cùng nhìn lại sơ đồ hoạt động của nó.



Hình 3.19: Sơ đồ hoạt động của YOLOv5 [20]

### 3.4.2 Hàm mất mát YOLOv5

YOLO sử dụng tổng của 3 hàm loss function tên là sum-squared error(SSE) bao gồm:

- Hàm mất mát vị trí: Dùng để tính toán sai lệch tọa độ của hộp;
- Hàm mất mát phân loại: Dùng để tính toán độ sai lệch của xác suất xuất hiện của các lớp;
- Hàm mất mát độ chắc chắn: Dùng để tính toán xác suất của các lớp.

Trước khi đi vào tìm hiểu ta cần làm rõ một số điều sau:

- Hàm loss function sẽ tính toán minimum hàm này chỉ khi có đối tượng còn nếu không có đối tượng thì mình không quan tâm;
- Bởi vì ta nhiều anchor hộp nên chúng ta cần chọn ra hộp có confidence cao nhất để thực hiện tính toán hoàn;
- Trọng số của hàm mất mát vị trí sẽ phải cao hơn mất mát phân loại.

a) Hàm mất mát độ chắc chắn

$$\mathcal{L}_{Con} = \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \quad (3.9)$$

Do mô hình lưới của YOLO, mạng tạo ra một tập hợp lớn các hộp không chứa bất kỳ đối tượng nào, các hộp dự đoán này có Confidence score = 0. vì số hộp không chứa vật thể nhiều hơn rất nhiều so với số hộp có vật thể, điều này dẫn đến

quá trình training bị phân kì sớm. Để giảm điều này thì ta thêm tham số  $\lambda_{noobj}=0.5$  để giảm độ quan trọng của đối tượng background.

Để đảm bảo độ quan trọng của hàm confidence loss khi có đối tượng chúng ta đề:

$$\mathbb{1}_{ij}^{obj} = \begin{cases} 1 & \text{nếu tồn tại đối tượng} \\ 0 & \text{nếu không tồn tại đối tượng} \end{cases}$$

Tương tự để giảm độ quan trọng của hàm confidence loss khi hộp không chứa đối tượng ta đề:

$$\mathbb{1}_{ij}^{noobj} = \begin{cases} 1 & \text{nếu tồn tại đối tượng} \\ 0 & \text{nếu không tồn tại đối tượng} \end{cases}$$

#### b) Hàm mất mát phân loại

Yolo dùng khoảng cách euclid để tính toán hàm mất mát. Với hàm này ta cũng chỉ trừng phạt những hộp có đối tượng ở trong. Do đó ta lại sử dụng công thức 2.29.

Công thức của hàm mất mát:

$$\mathcal{L}_{Cla} = \sum_{i=0}^{S^2} \mathbb{1}_i^{obj} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3.10)$$

Trong đó  $p_i$  là xác suất của đầu ra,  $\hat{p}_i$  là xác suất của nhãn.

#### c) Hàm mất mát vị trí

Ở phiên bản YOLOv4 tác giả sử dụng hàm mất mát khoảng cách Euclid, tuy nhiên sử dụng hàm này thì tất cả các tọa độ của hộp được xem như các biến độc lập, không có quan hệ gì với nhau, điều này làm mất đi tính quan hệ và bảo toàn của đối tượng, để cải thiện điều đó, hàm mất mát IoU [1] được đề xuất, hàm này có sự quan tâm đến vùng của hộp bao và vùng thật của đối tượng. Tác giả đã cải tiến thuật toán, quan tâm hơn đến vùng chồng lên nhau, khoảng cách giữa các điểm trung tâm và tỉ lệ của các hộp và quyết định sử dụng hàm CIoU [2] loss để làm hàm mất mát. Ta có hàm mất mát để tính toán tỉ lệ các hộp như sau:

$$\mathcal{R}_{CIoU} = \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \quad (3.11)$$

Với  $\alpha$  là tham số hệ số,  $v$  tính toán sự nhất quán của tỷ lệ của hộp bao, có công thức như sau:

$$v = \frac{4}{\pi^2} \left( \arctan \frac{w^{gt}}{h^{gt}} - \arctan \frac{w}{h} \right)^2 \quad (3.12)$$

Hàm mất mát CIoU được định nghĩa như sau:

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \quad (3.13)$$

Với tham số hệ số  $\alpha$  được xác định như sau:

$$\alpha = \frac{v}{(1-IoU)+v} \quad (3.14)$$

IoU được định nghĩa như hình 22.

#### d) Hàm mất mát tổng quát

Hàm mất mát tổng quát sẽ là tổng của 3 hàm mất mát trên:

$$\mathcal{L} = \mathcal{L}_{con} + \mathcal{L}_{clas} + \mathcal{L}_{CIoU} \quad (3.14)$$



### 3.5 Mô hình theo vết ong

#### 3.5.1 Simple Online Realtime Tracking

SORT [6] là thuật toán thuộc dạng theo vết dựa vào kết quả phát hiện đối tượng. Một đặc điểm của lớp các thuật toán là tách object detection ra như một bài toán riêng biệt và cố gắng tối ưu kết quả trong bài toán này. Công việc sau đó là tìm cách liên kết các bounding box thu được ở mỗi frame và gán ID cho từng đối tượng. Do đó, có một quá trình xử lý với mỗi khung hình mới như sau:

- **Detect:** phát hiện vị trí các đối tượng trong frame
- **Predict:** Dự đoán vị trí mới của các đối tượng dựa vào các khung hình trước đó
- **Associate:** Liên kết các vị trí detected với các vị trí dự đoán được để gán ID tương ứng

SORT sử dụng bộ lọc Kalman và thuật toán Hungary làm backbone. Thuật toán Hungary có thể khắc phục điểm yếu của Kalman Filter giúp các track và detection liên kết tốt hơn.

#### 3.5.2 Giải thuật Hungary

Giải thuật Hungary [21] được phát triển và công bố vào năm 1955, đề xuất để giải bài toán phân công công việc (assignment problem).

Phát biểu bài toán phân công: Có  $n$  người ( $i = 1, 2, \dots, n$ ) và  $n$  công việc ( $j = 1, 2, \dots, n$ ). Để giao cho người  $i$  thực hiện một công việc  $j$  cần một chi phí  $c$ . Bài toán đặt ra là cần giao cho người nào làm việc gì (mỗi người chỉ làm một việc và mỗi việc chỉ do một người làm) sao cho chi phí tổng cộng là nhỏ nhất.

**Liên hệ theo vết đối tượng:**

Có  $n$  detection ( $i = 1, 2, \dots, n$ ) và  $n$  track predicted ( $j = 1, 2, \dots, n$ ). Để liên kết một detection  $i$  với một track  $j$  giả sử dựa vào 1 độ đo  $D$  -  $D$  là khoảng cách giữa  $i$  và  $j$  trong không gian vector. Bài toán đặt ra là cần liên kết mỗi detection với mỗi track tương ứng sao cho sai số của việc liên kết là nhỏ nhất.

**Thuật toán:**

Giả sử:

$$z = \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij} \rightarrow \min \quad (3.15)$$

Trong đó:

$$\begin{cases} \sum_{i=1}^n x_{ij} = 1, i = 1, 2, \dots \\ \sum_{j=1}^n x_{ij} = 1, j = 1, 2, \dots \\ x_{ij} \geq 0 \end{cases} \quad (3.16)$$

Các số  $x_{ij}$  thỏa mãn các điều kiện trên gọi là một phương án phân công, hay ngắn gọn là một phương án, một phương án đạt cực tiểu của  $z$  được gọi là một phương án tối ưu hay lời giải của bài toán.

Để tìm  $x_{ij}$  chúng ta sẽ dựa vào 2 định lý sau:

[Định lý 1]. Giả sử ma trận chi phí của bài toán giao việc là không âm và có ít nhất  $n$  phần tử bằng 0. Hơn nữa nếu  $n$  phần tử 0 này nằm ở  $n$  hàng khác nhau và  $n$  cột khác nhau thì phương án giao cho người  $i$  thực hiện công việc tương ứng với số 0 này ở hàng  $i$  sẽ là phương án tối ưu (lời giải) của bài toán.

[Định lý 2]. Cho  $C=[c_{ij}]$  là ma trận chi phí của bài toán giao việc ( $n$  người,  $n$  việc) và  $X^*=[x_{ij}]$  là một lời giải của bài toán này. Giả sử  $C'$  là ma trận nhận được từ  $C$  bằng cách thêm số  $\alpha=0$  (dương hoặc âm) vào mỗi phần tử ở hàng  $r$  của  $C$ . Khi đó  $X^*$  cũng là lời giải của bài toán giao việc với ma trận chi phí  $C'$ .

Thuật toán Hungary dựa vào hai định lý này, từ đó hình thành được hướng xử lý bài toán : Biến đổi ma trận (cộng trừ vào các hàng hoặc cột) để đưa về ma trận có  $n$  phần tử bằng 0 nằm ở các hàng và cột khác nhau, sau đó, lấy ra phương án tối ưu là các vị trí chứa các phần tử 0 này.

Cụ thể hơn, có thể chia thuật toán thành các bước sau:

- Bước 1 (Bước chuẩn bị). Trừ các phần tử trên mỗi hàng của  $C$  cho phần tử nhỏ nhất trên hàng đó, tiếp theo trừ các phần tử trên mỗi cột cho phần tử nhỏ nhất trên cột đó. Kết quả ta nhận được ma trận  $C'$  có tính chất: trên mỗi hàng, cột có ít nhất một phần tử 0 và bài toán giao việc với ma trận  $C'$  có cùng lời giải như bài toán với ma trận  $C$ .
- Bước 2: Vẽ một số tối thiểu các đường thẳng trên dòng và cột để đảm bảo mọi phần tử 0 đều được đi qua.
- Bước 3: Nếu có  $n$  đường thẳng được vẽ, kết thúc thuật toán và tiến hành phân công công việc. Nếu số đường thẳng được vẽ nhỏ hơn  $n$ , vẫn chưa tìm được phương án phân công tối ưu, tiến hành bước tiếp theo.
- Bước 4: Mỗi hàng (hoặc cột) có đường thẳng vẽ qua, ta gọi các hàng (cột) đó là các hàng (cột) thiết yếu. Các hàng (cột) còn lại là các hàng (cột) không thiết yếu. Tìm phần tử nhỏ nhất không nằm trong các hàng (cột)

thiết yếu, tiến hành trừ mỗi hàng không thiết yếu cho phần tử nhỏ nhất ấy và cộng giá trị nhỏ nhất ấy cho cột thiết yếu. Ta được ma trận  $C''$  có cùng lời giải với ma trận  $C'$ . Sau đó quay lại Bước 2

### 3.5.3 Bộ lọc Kalman

Bộ lọc Kalman (Kalman Filter) [22] là một mô hình Linear-Gaussian State Space Model, được giới thiệu lần đầu năm 1960 và ứng dụng trong rất nhiều lĩnh vực khác nhau: Xe tự lái, thực tế ảo, kinh tế lượng, theo vết, điều khiển tối ưu, ...

Trong bài toán theo vết đối tượng, kalman filter được biết đến nhiều nhất với vai trò dự đoán các trạng thái của đối tượng hiện tại dựa vào các track trong quá khứ và update lại các detection sau khi đã được liên kết với các track trước đó.

Quá trình cần xử lý là 1 quá trình ngẫu nhiên với các mô hình đã được định nghĩa từ trước :

$$\begin{aligned} x_k &= \theta_{k-1}(x_{k-1}, u_{k-1}), x_k \in R^n \\ z_k &= h_{k-1}(x_k, w_k), z_k \in R^m \end{aligned} \quad (3.17)$$

Ở đây:

- $x_k$  là giá trị biến trạng thái của quá trình, thường là các giá trị ẩn, không thể quan sát được
- $z_k$  là giá trị đo được, quan sát được của quá trình.
- $\theta_k, h_k$  là các mô hình định nghĩa từ trước.
- $u_k, w_k$  lần lượt là nhiễu của quá trình và nhiễu trong lúc đo đạc

Một số bộ lọc Kalman chính như: Linear Kalman Filter, Extended Kalman filter, Unscented Kalman filter, ..

Linear Kalman Filter giả định các mô hình của quá trình  $(\theta_k, h_k)$  đều là các mô hình tuyến tính. Khi đó:

$$\begin{aligned} x_k &= F_{k-1}x_{k-1} + u_{k-1} \\ z_k &= H_{k-1}x_k + w_k \end{aligned} \quad (3.18)$$

Các bước xử lý tiếp theo của Kalman Filter có thể chia làm 2 phần chính (cách tiếp cận dựa trên xác suất) :

- **Dự đoán (Prediction):**

$$x_k = F_{k-1}x_{k-1} + u_{k-1} \quad (3.19)$$

Để dự đoán các giá trị trạng thái của quá trình ngẫu nhiên, ta dự đoán các giá trị mean  $\hat{x}$  và covariance  $\Sigma$ . Theo tính chất kỳ vọng và ma trận hiệp phương sai với vector ngẫu nhiên, ta có :

$$\hat{x}_k = E[x_k] = E[F_{k-1}x_{k-1} + u_{k-1}] = F_{k-1}E[x_{k-1}] + E[u_{k-1}] = F_{k-1}\hat{x}_{k-1}$$

$$\Sigma_k = \text{Var}(x_k) = \text{Var}(F_{k-1}x_{k-1} + u_{k-1}) = F_{k-1}\Sigma_{k-1}F_{k-1}^T + Q_{k-1}$$

- **Hiệu chỉnh (Update):**

Quá trình hiệu chỉnh phức tạp hơn 1 chút, ta có :

$$z_k = H_{k-1}x_k + w_k \quad (3.20)$$

Áp dụng định lí Bayes cho Linear Gaussian System, ta có :

$$\Sigma_k^F = \left( \Sigma_k^{-1} + H_k^T R_k^{-1} H_k \right)^{-1}$$

$$\hat{x}_k^F = \left( \Sigma_k^{-1} + H_k^T R_k^{-1} H_k \right)^{-1} \left[ H_k^T R_k^{-1} (z_k - H_k \hat{x}_k) + \Sigma_k^{-1} \hat{x}_k \right]$$

Tiếp tục áp dụng đồng nhất thức ma trận Woodbury, ta có :

$$\Sigma_k^F = \left( \Sigma_k^{-1} + H_k^T R_k^{-1} H_k \right)^{-1} = \Sigma_k - \Sigma_k H_k^T \left( R_k + H_k \Sigma_k H_k^T \right)^{-1} H_k \Sigma_k$$

$$\hat{x}_k^F = \left[ \Sigma_k - \Sigma_k H_k^T \left( R_k + H_k \Sigma_k H_k^T \right)^{-1} H_k \Sigma_k \right] \left[ H_k^T R_k^{-1} (z_k - H_k \hat{x}_k) + \Sigma_k^{-1} \hat{x}_k \right]$$

$$= \hat{x}_k + \Sigma_k H_k^T \left( R_k + H_k \Sigma_k H_k^T \right)^{-1} (z_k - H_k \hat{x}_k)$$

Để rút gọn 2 biểu thức trên, chúng ta đặt:

$$K_k = \Sigma_k H_k^T \left( R_k + H_k \Sigma_k H_k^T \right)^{-1} \quad (3.21)$$

$K_k$  được gọi là hệ số Kalman tại thời điểm k, khi đó:

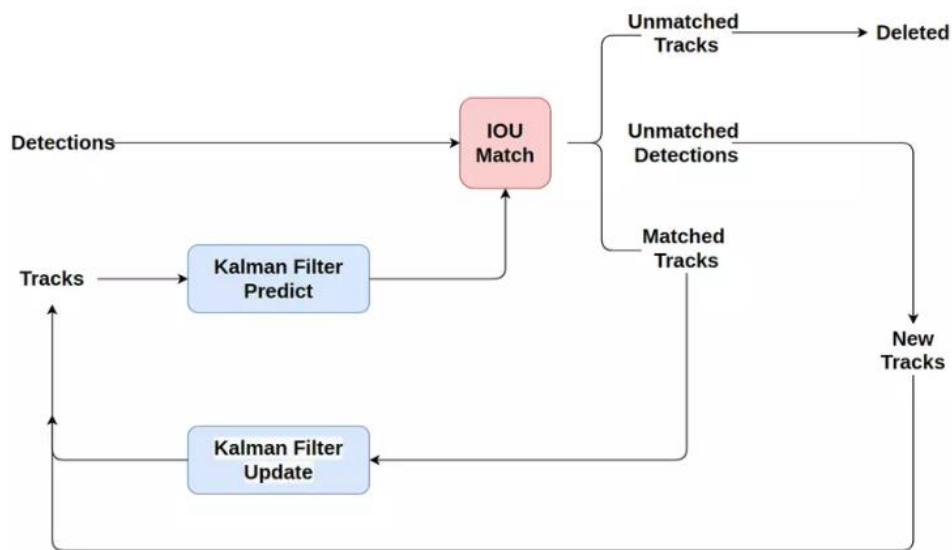
$$\hat{x}_k^F = \hat{x}_k + K_k (z_k - H_k \hat{x}_k)$$

$$\Sigma_k^F = \Sigma_k (I - K_k H_k)$$

### 3.5.4 Giải thích SORT

SORT - Simple Online Realtime Object Tracking, được giới thiệu lần đầu năm 2016, chỉnh sửa bổ sung v2 vào năm 2017, đề xuất giải pháp cho theo vật thể, đồng thời giải quyết cả 2 vấn đề theo vết nhiều đối tượng và theo vết thời gian thực.

SORT tập trung vào vấn đề liên kết giữa các detection và track sau khi đã phát hiện được từ khung hình, do đó, phần phát hiện đối tượng có thể là bất cứ mô hình detector nào hiện nay như : **YOLO** (v1, v2, v3, v4, v5), **SSD**, **RCNN** (RCNN, Fast RCNN, Faster RCNN, Mask RCNN), **CenterNet** - Object as Point, ... . Trong bài báo gốc, tác giả sử dụng bộ phát hiện là **Faster Region CNN** (FrCNN) với backbone là **VGG 16**



Hình 3.20: Luồng xử lý của SORT [6]

Luồng xử lý của SORT:

- **Bước 1:** SORT tiến hành sử dụng Kalman Filter để dự đoán các trạng thái track mới dựa trên các track trong quá khứ.
- **Bước 2:** Sử dụng những track vừa dự đoán được, kết hợp với các detection thu được từ detector, xây dựng ma trận chi phí cho Assignment Problem. Chi phí được sử dụng để đánh giá ở đây là giá trị IOU giữa các hộp giới hạn của track và detection.
- **Bước 3:** Sử dụng giải thuật Hungary giải bài toán giao việc với ma trận chi phí vừa lập
- **Bước 4:** Xử lý, phân loại các detection.
- **Bước 5:** Sử dụng Kalman filter để update những detection đã được liên kết với track.

Để ứng dụng được Kalman Filter, việc xác định được các dạng biến cũng như mô hình ban đầu của quá trình là điều bắt buộc cần có. Với giả định các đối tượng

chuyển động đều, và độc lập với các đối tượng khác, một track được xác định bằng:

$$x=[x, y, s, r, \dot{x}, \dot{y}, \dot{s}]^T$$

Với

- $x$  có ma trận hiệp phương sai ban đầu được khởi tạo với giá trị lớn để thể hiện sự không chắc chắn của trạng thái
- $u, v$  lần lượt là tọa độ của tâm đối tượng (ở đây là tâm bounding box)
- $s$  là diện tích của bounding box
- $r$  là tỉ lệ aspect ratio của bounding box
- $\dot{x}, \dot{y}, \dot{s}$  lần lượt là các giá trị vận tốc tương ứng của  $x, y, s$

Do giả định các đối tượng chuyển động đều, ta có:

$$\begin{cases} x_k = x_{k-1} + \dot{x}_{k-1}dt \\ y_k = y_{k-1} + \dot{y}_{k-1}dt \\ s_k = s_{k-1} + \dot{s}_{k-1}dt \\ r_k = r_{k-1} \\ \dot{x}_k = \dot{x}_{k-1} \\ \dot{y}_k = \dot{y}_{k-1} \\ \dot{s}_k = \dot{s}_{k-1} \end{cases} \quad (3.22)$$

Khi đó, phương trình:

$$x_k = F_{k-1}x_{k-1} + u_{k-1} \quad (3.23)$$

tương đương với ( $u_k$  được khởi tạo tuân theo phân phối chuẩn có mean = 0 và covariance không đổi)

$$\begin{bmatrix} x_k \\ y_k \\ s_k \\ r_k \\ \dot{x}_k \\ \dot{y}_k \\ \dot{s}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ s_{k-1} \\ r_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \\ \dot{s}_{k-1} \end{bmatrix} + u_{k-1} \quad (3.24)$$



Phương trình:

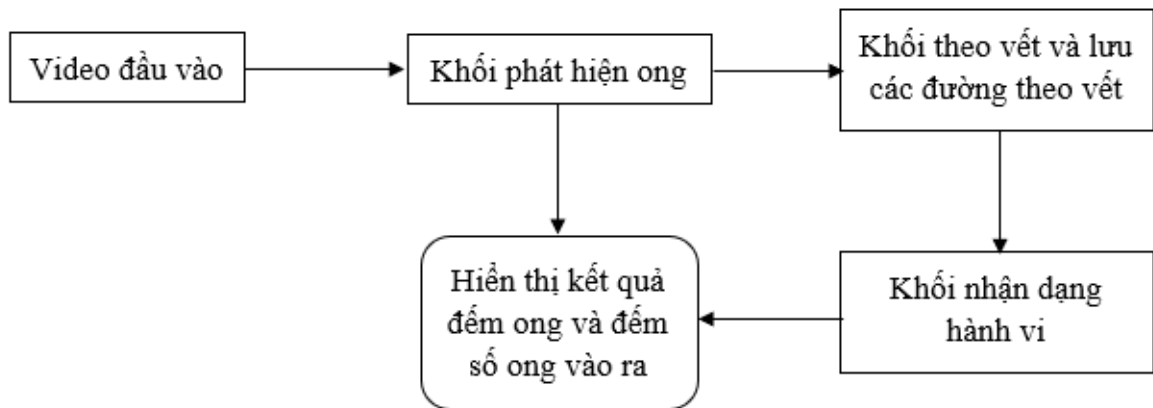
$$z_k = H_{k-1}x_k + w_k \quad (3.25)$$

tương đương với ( $w_k$  được khởi tạo tuân theo phân phối chuẩn có mean = 0 và covariance không đổi)

$$\begin{bmatrix} x'_k \\ y'_k \\ h'_k \\ w'_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} x_k \\ y_k \\ s_k \\ r_k \\ \dot{x}_k \\ \dot{y}_k \\ \dot{s}_k \end{bmatrix} + w_k \quad (3.26)$$

## CHƯƠNG 4. TRIỂN KHAI KỸ THUẬT ĐỀ XUẤT, THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

### 4.1 Sơ đồ tổng quát đề xuất

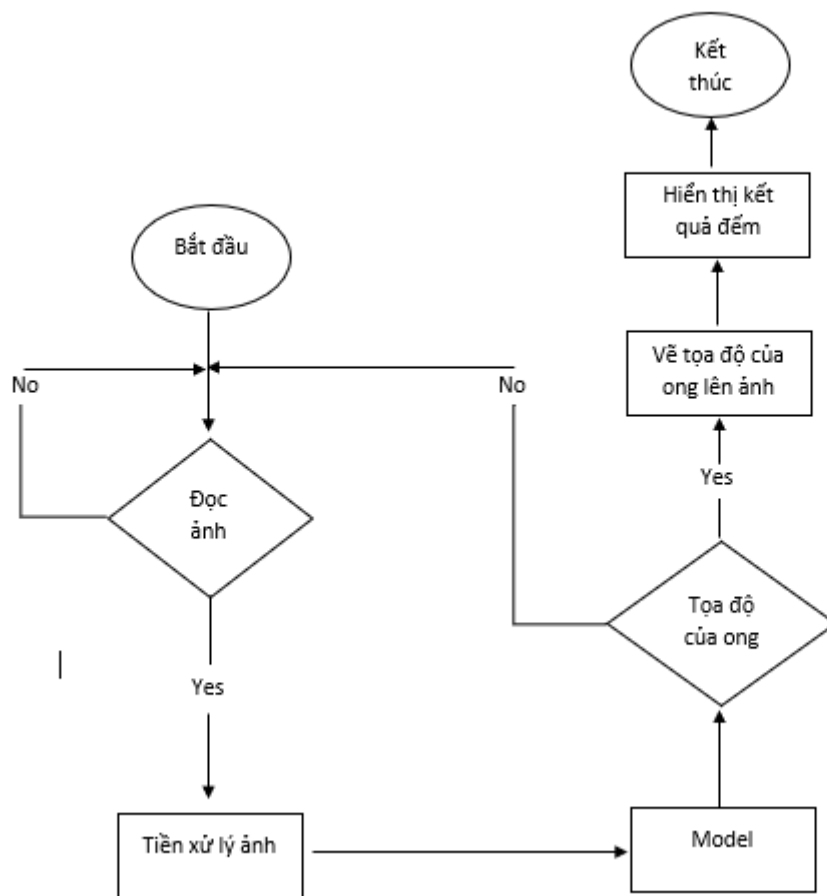


Hình 4.1: Mô hình đề xuất

Mô hình đề xuất gồm có ba phần chính:

- Khởi phát hiện ong dựa vào xử lý hình ảnh, từ đó có thể đếm được số lượng ong trước cửa chuồng.
- Từ kết quả phát hiện ong, sử dụng vị trí của từng con ong trong ảnh để theo vết, gán id cho từng con ong và lưu các tọa độ của chúng trong từng khung hình vào cơ sở dữ liệu.
- Khởi nhận dạng hành vi: Nhận dạng các đường theo vết từ đó có thể nhận dạng các hành vi của từng con ong.

#### 4.1.1 Lưu đồ hoạt động của chương trình phát hiện ong



Hình 4.2: Lưu đồ hoạt động phát hiện và đếm ong

Ảnh được đọc từ camera hoặc từ video, nếu không đọc được ảnh ta sẽ quay lại bước đọc ảnh, nếu đọc được ảnh thì ảnh sẽ được xử lý trước khi đưa vào model ví dụ như sửa kích thước. ảnh sau đó được đưa vào model kết quả đầu ra là tọa độ của hình chữ nhật xác định vị trí của ong. Nếu không có tọa độ nào có nghĩa là không có ong ở trong ảnh thì quay lại bước đọc ảnh, nếu có ong trong ảnh thì sẽ vẽ hộp lên ảnh và đếm số ong ở trong ảnh, trả về tọa độ của tâm giữa hình hộp và số lượng ong. Chương trình kết thúc

Quá trình hoạt động được chia làm 2 giai đoạn:

- Huấn luyện mô hình;
- Sử dụng mô hình.

#### Huấn luyện mô hình

Quá trình huấn luyện mô hình cần thực hiện các bước:

- Thu dữ liệu thực tế ở chuồng ong tại Học Viện Nông Nghiệp và gán nhãn dữ liệu.
- Tìm kiếm thêm dữ liệu có sẵn ở trên mạng.
- Kết hợp cả hai tập dữ liệu, lọc dữ liệu và phân chia dữ liệu thành 3 nhóm chính là huấn luyện, đánh giá và kiểm tra.
- Xây dựng mô hình .

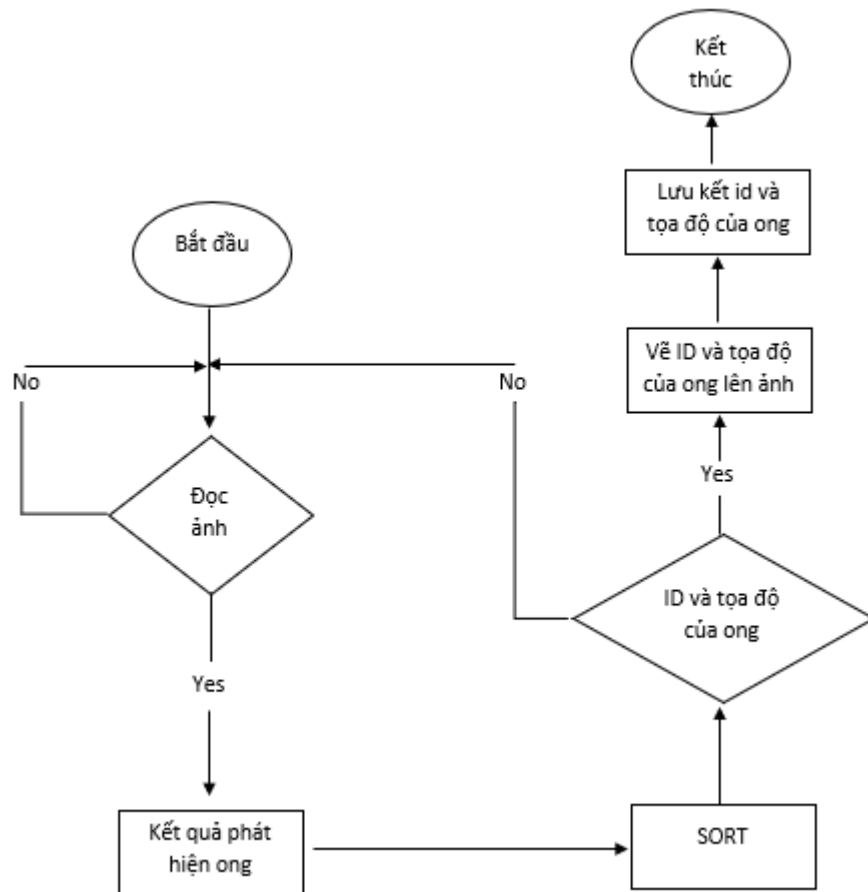
- Huấn luyện mô hình, đánh giá và tinh chỉnh lại các tham số.
- Lưu lại mô hình với độ chính xác cao nhất.

### Sử dụng mô hình

Quá trình sử dụng mô hình cũng trải qua các bước:

- Đọc ảnh từ camera;
- Đọc mô hình vào chương trình;
- Cho ảnh đi qua mô hình vừa được huấn luyện và lấy kết quả xử lý.

#### 4.1.2 Lưu đồ hoạt động của chương trình theo vết ong

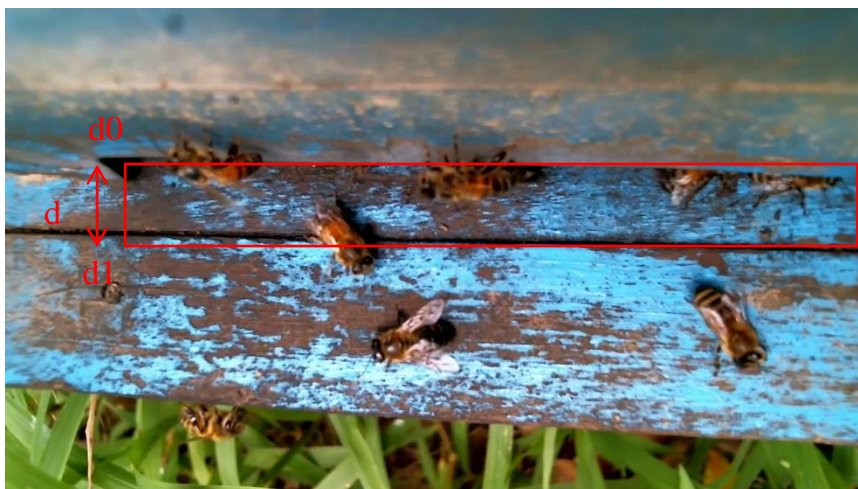


Hình 4.3: Lưu đồ hoạt động theo vết ong

Chương trình theo vết SORT sẽ sử dụng kết quả phát hiện ong để xử lý gán ID và vẽ tọa độ cho từng con ong trong mỗi khung hình. Kết quả sẽ được lưu dưới dạng txt file bao gồm số khung hình, ID của ong và tọa độ tương ứng.

### 4.1.3 Lưu đồ hoạt động của chương trình nhận dạng hành vi

Xét phạm vi của tổ là hình chữ nhật màu đỏ như hình dưới đây. Với bề rộng của tổ là  $d = (d_0, d_1)$ .



Hình 4.4. Minh họa của tổ

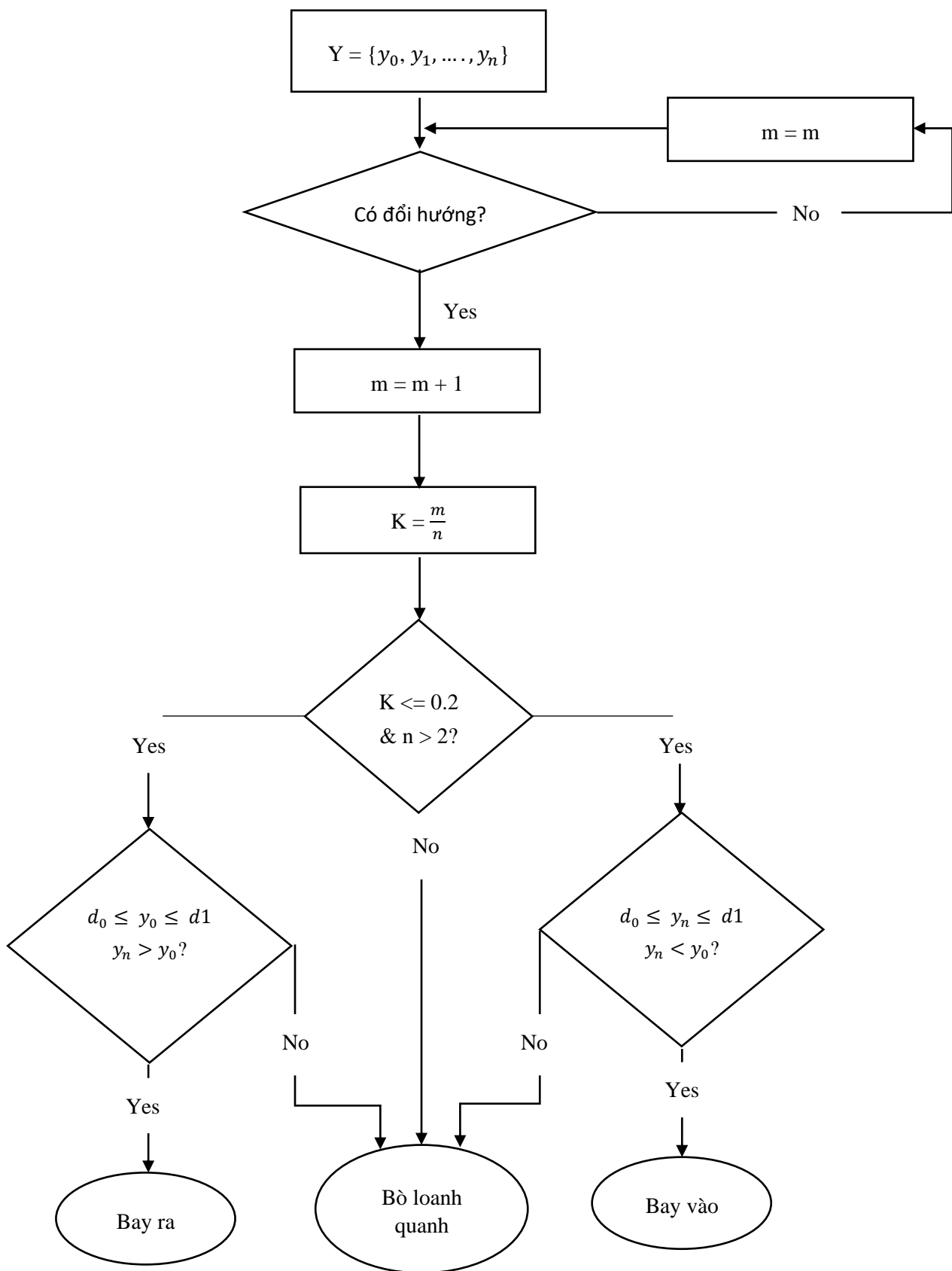
Xét quá trình di chuyển của 1 con ong, mỗi khung hình chúng có tọa độ lần lượt là  $Bee = \{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$ .

Ở đây ta chỉ xét tọa độ trục y của con ong trong cả quá trình di chuyển của chúng  $Y = \{y_0, y_1, \dots, y_n\}$ .

Muốn xét ong có thay đổi hướng trong quá trình di chuyển hay không ta xét 3 điểm y gần nhau xem 3 điểm đó có tạo thành 1 chuỗi tăng hay 1 chuỗi giảm hay không. Gọi m là số lần ong thay đổi hướng di chuyển hay m chính là số lần 3 điểm không tạo thành 1 chuỗi tăng hoặc giảm. Ta có hệ số K:

$$K = \frac{m}{n} \quad (4.1)$$

Dưới đây là lưu đồ thuật toán phân tích hành vi:



## 4.2 Mô hình phát hiện ong

### 4.2.1 Xây dựng bộ dữ liệu

Cơ sở dữ liệu huấn luyện được lấy tách từ video quay trước cửa tổ ong. Sau quá trình thu thập dữ liệu, nhóm thu được 16569 ảnh. Tập dữ liệu được chia làm hai phần huấn luyện và kiểm tra. Phần huấn luyện bao gồm 202 ảnh với số ong được gán nhãn là 2933. Phần kiểm tra gồm 16367 ảnh.

Dữ liệu sau khi gán nhãn bằng tọa độ xmin, ymin, xmax và ymax nhưng mô hình YOLO yêu cầu tập dữ liệu phải ở dạng xcenter, ycenter, w, h.

Với xcenter, y center lần lượt là tọa độ tâm của hộp. w, h lần lượt là chiều rộng và chiều cao của hộp.

Ta cần chuyển lại bằng công thức:

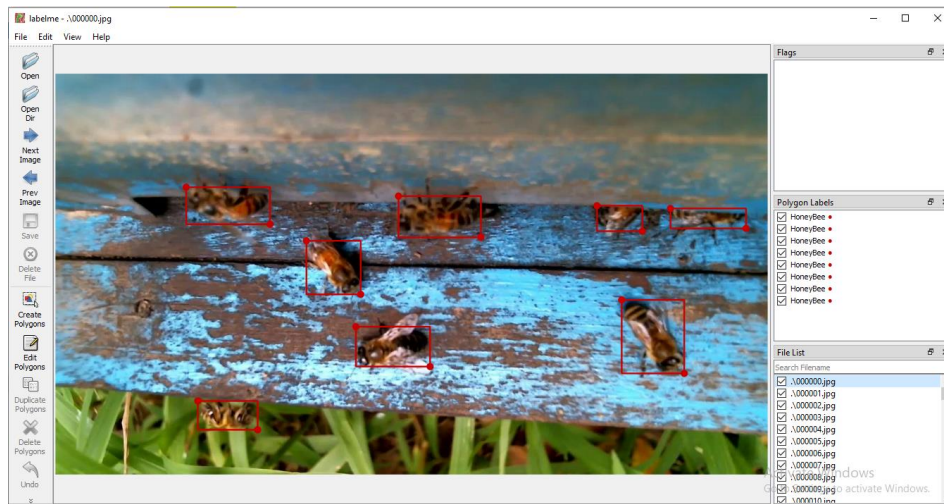
$$xcenter = \frac{xmax - xmin}{2} + xmin \quad (4.2)$$

$$ycenter = \frac{ymax - ymin}{2} + ymin \quad (4.3)$$

$$w = xmax - xmin \quad (4.4)$$

$$h = ymax - ymin \quad (4.5)$$

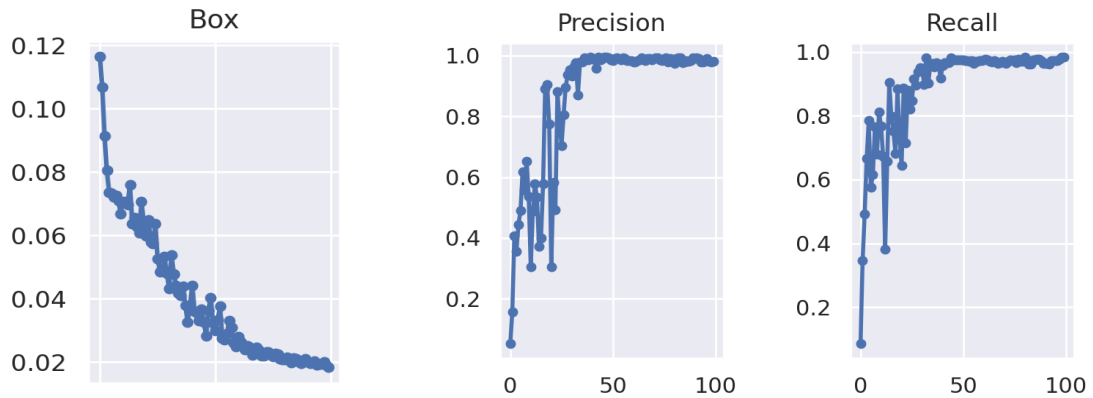
Phần gán nhãn dữ liệu sử dụng phần mềm LabelMe và được gán nhãn như hình sau.



Hình 4.5 Gán nhãn dữ liệu



#### 4.2.2 Kết quả huấn luyện mô hình mạng YOLOv5



Hình 4.6 Kết quả huấn luyện phát hiện ong

- Đồ thị đầu tiên Box biểu thị hàm loss function, giá trị hàm này tiến về 0 sau 100 lần học.
- Đồ thị thứ hai Precision biểu diễn độ chính xác của model sau 100 lần học.
- Đồ thị thứ ba Recall biểu diễn độ nhạy sau 100 lần học.

➤ Cho thấy sự hội tụ của mạng sau khi huấn luyện xong.

#### 4.2.3 Kết quả đánh giá phát hiện và đếm ong

##### a) Thước đo đánh giá

Các thước đo đánh giá cho một thuật toán phát hiện đối tượng được chia thành hai nhóm mức box và mức ảnh.

- Mức box:

- Intersection over Union (IoU): là tỉ lệ thể hiện mức độ trùng khớp giữa box thực sự của vật thể với box dự đoán của hệ thống.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

Hình 4.7. Minh họa thước đo đánh giá IoU

- Mức ảnh:
  - False alarm rate: tỉ lệ phát hiện sai là ong trên tổng số ong được đánh nhãn trong tập kiểm tra.
  - Miss rate: tỉ lệ phát hiện sót ong trên tổng số ong được đánh nhãn trong tập kiểm tra.
  - Sensitivity rate: độ nhạy của của mô hình, bằng  $1 - \text{miss rate}$

		Predicted Label	
		Positive	Negative
Known Label	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)
False alarm rate		$\frac{FP}{TP + FP}$	
Miss rate		$\frac{FN}{TP + FN}$	
Sensitivity rate		$\frac{TP}{TP + FN}$	

Bảng 4.1. Công thức các thước đo đánh giá trên mức ảnh

Đối với bài toán xác định đối tượng thì TP, FP, TN, FN được định nghĩa như sau:

- TP (True Positive): là các predicted hộp với IoU lớn hơn hoặc bằng 1 giá trị IoU cố định
- FP (False Positive): là các predicted hộp với IoU nhỏ hơn 1 giá trị IoU cố định
- FN (False Negative): mô hình không bắt được đối tượng trong ảnh (ứng với ground truth tương ứng);
- TN (True Negative): ta không cần quan tâm đến thông số này.

b) *Kết quả đánh giá trên tập dữ liệu*

Tập dữ liệu đánh giá gồm 507 ảnh với 7658 con ong được đánh nhãn.

IoU	False alarm rate ( % )	Miss rate ( % )	Sensitivity rate ( % )
0.7	1.06	3.59	96.41
0.6	0.71	3.25	96.75
<b>0.5</b>	<b>0.58</b>	<b>3.13</b>	<b>96.87</b>
0.4	0.71	2.96	97.04
0.3	0.97	2.8	97.2

Bảng 4.2 Kết quả đánh giá phát hiện ong trên tập dữ liệu được đánh nhãn

Từ bảng đánh giá trên cho thấy khi IoU bằng 0.5 mô hình có false alarm rate thấp nhất xấp xỉ 0.6% và độ nhạy cao xấp xỉ 97%.

c) *Kết quả đếm*

Dưới đây là một số kết quả đếm ong khi IoU = 0.5 với false alarm rate = 0.6% và sensitivity rate = 97%.



Hình 4.8. Kết quả phát hiện và đếm ong ở tổ ong thứ nhất 1

Kết quả phát hiện và đếm ong ở hình 4.8 rất tốt có tỉ lệ phát hiện nhầm và bỏ sót ong là 0%



Trong đó:

- **FP** (False Positive): tổng số lần xuất hiện một đối tượng được phát hiện mặc dù không có đối tượng nào tồn tại
- **FN** (False Negative): tổng số lần mà đối tượng hiện có không được phát hiện.
- **ID Switches**: tổng số lần 1 đối tượng bị gán cho 1 ID mới trong suốt quá trình theo vết.
- **GT**: tổng số hộp giới hạn thực sự của tất cả đối tượng.
- ❖ **MT** (Most Tracked Target): Số đối tượng được theo vết thành công ít nhất 80% trong hành trình di chuyển của chúng.
- ❖ **ML** (Most Lost Target): Số đối tượng được theo vết thành công nhiều nhất 20% trong hành trình di chuyển của chúng.
- ❖ **MOTP** (Multiple Object Tracking Precision): là trung bình độ chồng chéo hộp giới hạn giữa tất cả các đối tượng được theo vết thành công (True Positives) với box thực sự của chúng hay trung bình tất cả IoU của tất cả các box

$$MOTP = \frac{\sum_{i,t} d_{i,t}^i}{\sum_t c_t} \quad (4.7)$$

Trong đó:

- $d_{i,t}$  là độ chồng chéo hộp giới hạn giữa đối tượng  $i$  và hộp thực sự của đối tượng.
- $c_t$  là số lần đối tượng được theo vết thành công.

b) *Kết quả đánh giá trên tập dữ liệu được đánh nhãn*

Tập dữ liệu gồm 601 ảnh với 8079 con ong được đánh nhãn và 96 hành trình.

FP	FN	GT	ID Switches	$c_t$	MOTA (%)	MOTP (%)	MT	ML
17	156	8079	10	7509	97.78	97.23	89	5

Bảng 4.3. Kết quả đánh giá theo vết trên tập dữ liệu được đánh nhãn

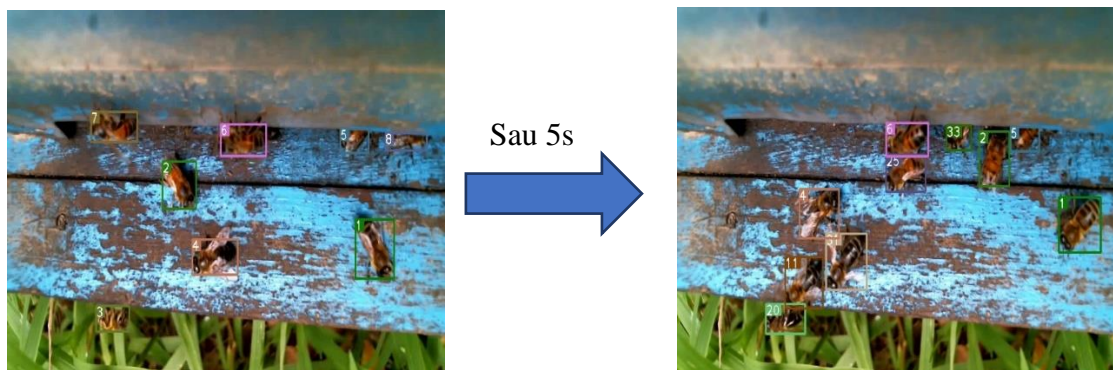


Từ bảng trên có thể thấy:

- Số lần ong bị đổi ID là 10 lần do các con đang bay ở tốc độ cao camera không bắt kịp và vài con ong trùng lên quá lâu.
- Số lần phát hiện sai ong là 17 lần, 156 trường hợp phát hiện sót do các con ong bắt đầu bò ra từ tổ còn khá bé và vài con ong lật ngửa bụng, do thiếu dữ liệu mô hình chưa phát hiện được.
- Số lần theo vết thành công là 7509 trên 8079 track.
- Độ chính xác của quá trình theo vết rất cao là 97,78%.
- Độ chồng chéo hộp giới hạn so với dữ liệu đánh nhãn là 97,23%.
- Với 96 hành trình di chuyển, số hành trình theo vết thành công trên 80% là 89, số hành trình theo vết ít hơn 20% là 5.

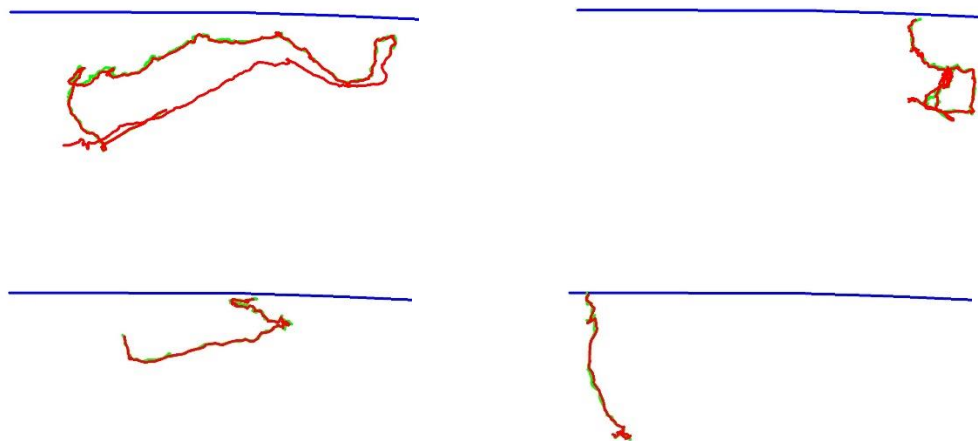
#### 4.3.2 Một số kết quả theo vết ong

Duy trì định danh của một con ong trong điều kiện mật độ lớn rất tốt với độ chính xác theo vết xấp xỉ 98%.



Hình 4.11. Minh họa kết quả theo vết

Dưới đây là một số kết quả so sánh giữa 2 đường theo vết thu được (màu đỏ) và đường theo vết thực tế (màu xanh).



Hình 4.12: Một số kết quả so sánh 2 đường theo vết thu được và thực tế

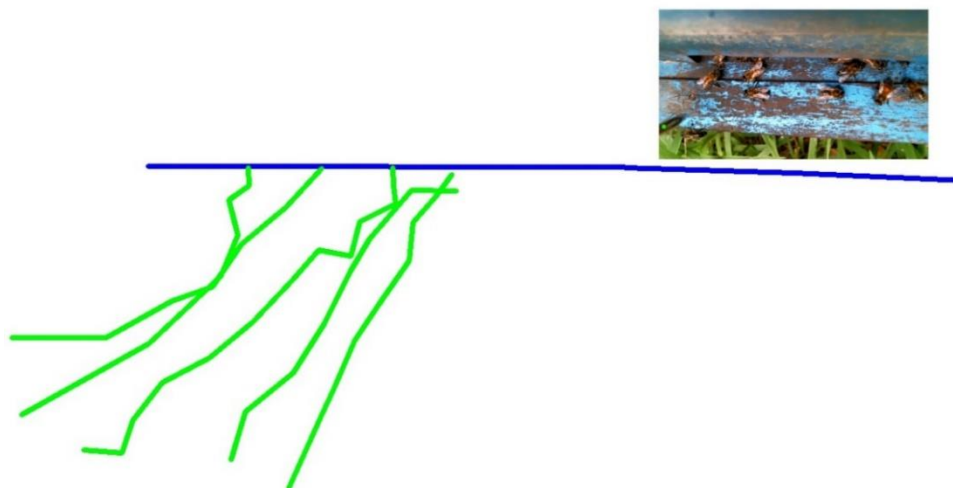
Các kết quả theo vết ong hầu hết đều rất chính xác, tuy nhiên vẫn còn một số trường hợp theo vết không chính xác do các con ong đè lên nhau rất lâu và tốc độ ong bay nhanh làm camera không theo kịp dẫn đến việc bị mất track.

#### 4.4 Kết quả nhận dạng hành vi của ong

##### 4.4.1 Các hành vi của ong

###### 1) Bay vào (Flying in)

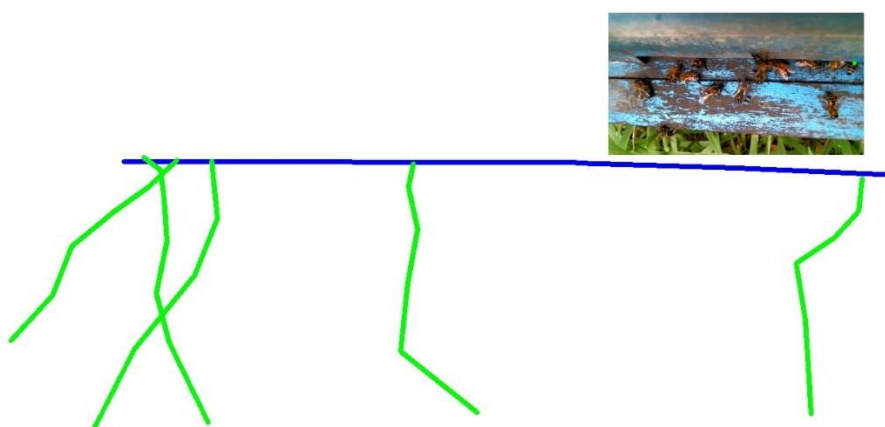
Trong hành trình di chuyển ong thay đổi hướng di chuyển ít và có vị trí track đầu xa mép tổ hơn so với track cuối.



Hình 4.13: Minh họa đường đi của 5 con ong bay vào

###### 2) Bay ra (Flying out)

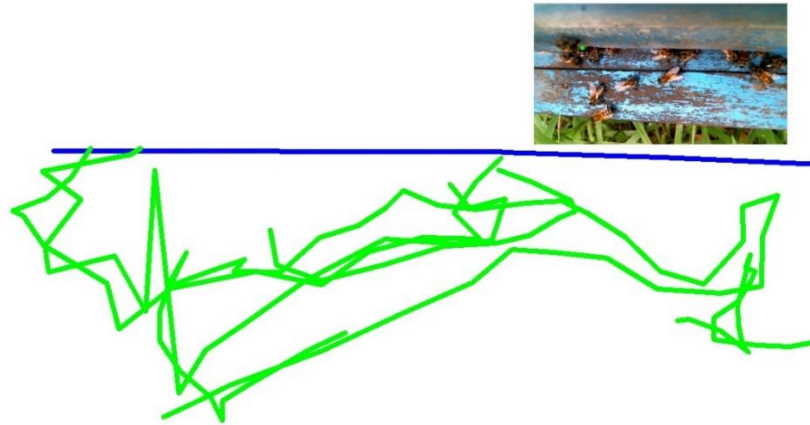
Trong hành trình di chuyển ong thay đổi hướng di chuyển ít và có vị trí track đầu gần mép tổ hơn so với track cuối.



Hình 4.14: Minh họa đường đi của 5 con ong bay ra

### 3) Bò loanh quanh tổ (Crawling)

Trong hành trình di chuyển ong thay đổi hướng đi rất nhiều lần. Theo chuyên gia ong đánh giá những con ong này là ong bảo vệ hoặc đang bị bệnh bại liệt nếu chúng có những đốm bệnh trên người.



Hình 4.15: Minh họa đường đi của 5 con ong bò loanh quanh tổ

#### 4.4.2 Kết quả đánh giá nhận dạng hành vi của ong

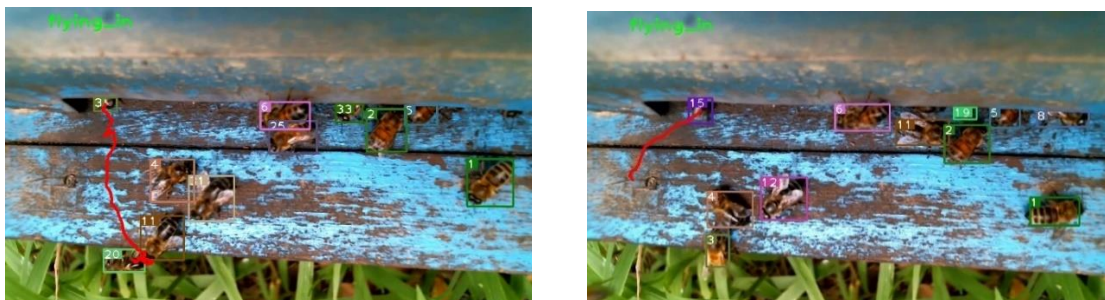
Tập dữ liệu đánh giá là 601 ảnh với 96 hành trình. Số hành vi phân tích đúng là 90 (93.75%).

	Bay vào	Bay ra	Bò loanh quanh
Dự đoán	18	31	47
Thực tế	19	32	45

Bảng 4.4 Kết quả đánh giá nhận dạng hành vi ong mật

#### 4.4.3 Một số kết quả nhận dạng

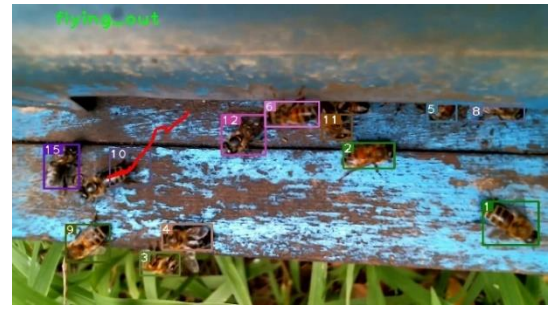
##### 1) Bay vào



Hình 4.16: Kết quả nhận dạng hành vi bay vào



## 2) Bay ra



Hình 4.17: Kết quả nhận dạng hành vi bay ra

## 3) Bò loanh quanh tổ



Hình 4.18: Kết quả nhận dạng hành vi bò loanh quanh tổ

Kết quả nhận dạng khá chính xác so với hành vi thực tế của ong. Chênh lệch giữa các số lượng hành vi máy đếm và thực tế không đáng kể.



Hình 4.19: Kết quả nhận dạng hành vi không chính xác 1

Kết quả nhận dạng hành vi ở hình 4.19 là bò loanh quanh nhưng thực tế con ong này đang bay vào do kết quả theo vết không chính xác.



Hình 4.20: Kết quả nhận dạng hành vi không chính xác 2

Kết quả nhận dạng hành vi ở hình 4.20 là bò loanh quanh nhưng thực tế con ong này đang bay ra. Lý do cho việc nhận dạng sai là chưa hiệu chỉnh camera dẫn đến quá trình nhận dạng hành vi như thuật toán sai lệch. Do thời gian có hạn em vẫn chưa làm phần hiệu chỉnh này.

#### 4) Kết quả đếm trong 1 video gồm 5067 frames



Hình 4.21: Kết quả đếm số hành vi của ong trong 1 video

Kết quả đếm số hành vi trong video trên bao gồm:

- Bay ra: 110 con ong
- Bay vào: 318 con ong
- Bò loanh quanh: 449 con ong.

## KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 1. Kết luận

Nhìn chung đồ án đã đạt được những mong đã đặt ra từ đầu. Dưới sự giúp đỡ của các thầy cô em đã xây dựng được mô hình phát hiện, đếm ong và nhận dạng hành vi của chúng. Các kết quả phát hiện, theo vết và nhận dạng hành vi đạt kết quả rất cao và có thể đáp ứng được thời gian thực. Đây có thể là bước đầu cho việc chuyển đổi số cho nghề ong. Đối với người sản xuất ong, đây là lần đầu tiên, các thông số về ong được đánh giá định lượng và tự động. Qua đồ án em đã học thêm được những kỹ năng như tìm kiếm tài liệu, tổng hợp kiến thức trong những bài báo để vận dụng vào giải quyết các bài toán thực tế, kỹ năng trình bày và viết báo cáo. Đặc biệt với đề tài “Xây dựng hệ thống giám sát và phân tích hành vi ong mật thời gian thực” nhóm em đã đạt được giải ba cuộc thi nghiên cứu khoa học năm 2020 – 2021 cấp bộ môn và lọt vào top 10 đề tài để thi cấp viện Điện vừa qua.

### 2. Hạn chế

Tuy hệ thống hoạt động tốt nhưng do hạn chế về mặt thời gian và dịch bệnh nên đồ án còn một số thiếu sót nhất định. Cụ thể hạn chế trong đồ án em nhận thấy là:

- Dữ liệu huấn luyện còn hạn chế, một số trường hợp ong lật ngửa người lên làm mô hình không phát hiện được.
- Tốc độ ghi hình của camera chưa đủ, một số trường hợp các con ong bay quá nhanh không theo vết được.
- Chưa hiệu chỉnh camera dẫn đến vài trường hợp nhận dạng hành vi còn chưa chính xác.

### 3. Hướng phát triển

Với những kết quả và hạn chế trên, hướng phát triển tiếp theo của đồ án là:

- Tăng dữ liệu huấn luyện để có thêm các đặc trưng mới, cải thiện tốc độ ghi hình và hiệu chỉnh camera giúp mô hình phát hiện ong và nhận dạng hành vi hiệu quả hơn.
- Mở rộng đánh giá trong nhiều tình huống, điều kiện thời tiết thay đổi.
- Xây dựng hệ thống IoT để theo dõi sinh hoạt đàn ong trong một cơ sở sản xuất mật ong có quy mô lớn.

## TÀI LIỆU THAM KHẢO

- [1] Thi Nha Ngo, Kung-Chin Wu, En-Cheng Yang, Ta-Te Lin, "A real-time imaging system for multiple honey bee tracking and activity," 2019.
- [2] Schöorkhuber, Dominik, "Bee Tracking," 1 February 2017.
- [3] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020.
- [4] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016.
- [5] M. CRAWFORD, "AUTOMATED COLLECTION OF HONEY BEE HIVE DATA USING THE RASPBERRY PI," August 2017.
- [6] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft, "SIMPLE ONLINE AND REALTIME TRACKING," 2017.
- [7] Gang Jun Tu, Mikkel Kragh Hansen , Per Kryger , Peter Ahrendt, "Automatic behaviour analysis system for honeybees using computer vision," 2016.
- [8] Chiu Chen, En-Cheng Yang , Joe-Air Jiang , Ta-Te Lin, "An imaging system for monitoring the in-and-out activity of honey bees," 2012.
- [9] Malika Nisal Ratnayake, Adrian G. Dyer, Alan Dorin, "Tracking individual honeybees among wildflower clusters with computer vision-facilitated pollinator monitoring," 2021.
- [10] Vladimir Kulyukin, Sarbajit Mukherjee, Angela Minichiello and Tadd Truscott, "BeePIV: A Method to Measure Apis Mellifera Traffic with Particle Image Velocimetry in Videos," 2021.
- [11] Aston Zhang, Zack C. Lipton, Mu Li, Dive into Deep Learning, Nov 06 2020.
- [12] S. Pattanayak, Pro Deep Learning with Tensorflow, a press, 2019.
- [13] Wei Liu, Andrew Rabinovich, Alexander C. Berg, "ParseNet: Looking Wider to See Better," 2015.
- [14] Guayui Zhou, Fei Jiang, Ruimin Shen, "Who Are Raising Their Hands? Hand-Raiser Seeking Based on Object Detection and Pose Estimation," 2018.
- [15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, "SSD: Single Shot MultiBox Detector," in *arxiv*, 2016.

- [16] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, "CSPNET: A NEW BACKBONE THAT CAN ENHANCE LEARNING CAPABILITY OF CNN," in *arxiv*, 2019.
- [17] Stanford, "CS231 cs231n-convolutional-neural-networks-visual-recognition CourseStanford
- [18] B. Hariharan, P. Arbeláez, R. Girshick, J. Malik, "Hypercolumns for object segmentation and fine-grained localization," *Proceedings of the IEEE conference on computer vision and pattern recognition*, p. 447–456, 2015.
- [19] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement," 2018.
- [20] T. Vu, "Machine Learning coban," [Online]. Available: <https://machinelearningcoban.com/>.
- [21] Karleigh Moore, Nathan Landman, and Jimin Khim, "Hungarian Maximum Matching Algorithm".
- [22] KALMAN, R. E., "A New Approach to Linear Filtering".
- [23] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler, "MOT16: A Benchmark for Multi-Object Tracking," 2016.