

ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN



BÁO CÁO CUỐI KÌ  
**PHÂN LOẠI CÁ TRONG THỜI GIAN THỰC**

**MAT3508 - NHẬP MÔN TRÍ TUỆ NHÂN TẠO**

**Học kỳ 1, Năm học 2025-2026**

**Giảng viên: TS. Hoàng Anh Đức**

Nhóm thực hiện:

Đặng Hải Bình	23001502
Chu Thị Mai Duyên	23001510
Đỗ Thị Mây	23001536
Nguyễn Trọng Đức	23001961
Nguyễn Quốc Hiệu	23001520

**Hà Nội - 2025**

**ĐẠI HỌC QUỐC GIA HÀ NỘI  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**BÁO CÁO CUỐI KÌ  
PHÂN LOẠI CÁ TRONG THỜI GIAN THỰC**

**MAT3508 - NHẬP MÔN TRÍ TUỆ NHÂN TẠO**

**Học kỳ 1, Năm học 2025-2026**

**Giảng viên: TS. Hoàng Anh Đức**

Nhóm thực hiện:

Đặng Hải Bình	23001502
Chu Thị Mai Duyên	23001510
Đỗ Thị Mây	23001536
Nguyễn Trọng Đức	23001961
Nguyễn Quốc Hiệu	23001520

**Hà Nội - 2025**

## Thông tin Dự án

<b>Học phần:</b>	MAT3508 – Nhập môn Trí tuệ Nhân tạo
<b>Học kỳ:</b>	Học kỳ 1, Năm học 2025-2026
<b>Trường:</b>	Trường Đại học Khoa học Tự nhiên – Đại học Quốc gia Hà Nội
<b>Tên dự án:</b>	Phân loại cá trong thời gian thực và tích hợp tra cứu thông tin loài cá
<b>Ngày nộp:</b>	30/11/2025
<b>Báo cáo PDF:</b>	Liên kết tới báo cáo PDF trong kho GitHub
<b>Slide thuyết trình:</b>	Liên kết tới slide thuyết trình trong kho GitHub
<b>GitHub:</b>	<a href="https://github.com/wokhyu/real-time-fish-classification">https://github.com/wokhyu/real-time-fish-classification</a>
<b>Kho lưu trữ:</b>	<a href="https://drive.google.com/drive/folders/10DZXecvxZj9Ys18drgafsNuKt_pSRCJA?usp=drive_link">https://drive.google.com/drive/folders/10DZXecvxZj9Ys18drgafsNuKt_pSRCJA?usp=drive_link</a>

## Thành viên nhóm

Họ tên	Mã sinh viên	Tên GitHub
Đặng Hải Bình	23001502	chaotolabin
Chu Thị Mai Duyên	23001510	maiduyen05
Đỗ Thị Mây	23001536	sharonmyoui37
Nguyễn Trọng Đức	23001961	trognduck
Nguyễn Quốc Hiệu	23001520	wokhyu

# MỤC LỤC

<b>Phân công nhiệm vụ</b>	<b>4</b>
<b>Lời nói đầu</b>	<b>7</b>
<b>1 Giới thiệu</b>	<b>8</b>
1.1 Tóm tắt dự án . . . . .	8
1.2 Bài toán đặt ra . . . . .	8
1.2.1 Vấn đề . . . . .	9
1.2.2 Động lực nghiên cứu . . . . .	9
1.2.3 Ý nghĩa thực tiễn . . . . .	9
<b>2 Tổng quan và trực quan hóa dữ liệu</b>	<b>10</b>
2.1 Tổng quan dữ liệu . . . . .	10
2.2 Trực quan hóa dữ liệu . . . . .	12
2.2.1 Số lượng ảnh các họ cá . . . . .	12
2.2.2 Kích thước ảnh và Bounding box trong bộ dữ liệu . . . . .	14
<b>3 Phương pháp và Triển khai</b>	<b>18</b>
3.1 Phương pháp . . . . .	18
3.1.1 Mô hình phân loại ảnh CNN . . . . .	18
3.1.2 Mô hình phát hiện đối tượng YOLOv8 . . . . .	23
3.2 Triển khai hệ thống . . . . .	25
3.2.1 Kiến trúc tổng thể hệ thống . . . . .	25
3.2.2 Môi trường và công cụ triển khai . . . . .	26
3.2.3 Tiền xử lý và chuẩn hoá dữ liệu . . . . .	26
3.2.4 Huấn luyện mô hình . . . . .	27
3.2.5 Tổ chức mã nguồn hệ thống . . . . .	27
3.2.6 Pipeline xử lý trong thời gian thực . . . . .	29
<b>4 Kết quả và Phân tích</b>	<b>30</b>
4.1 Đánh giá mô hình Yolov8 . . . . .	30
4.1.1 Kết quả tổng thể . . . . .	30
4.1.2 Đánh giá theo các nhóm độ phổ biến . . . . .	31
4.1.3 Độ chính xác theo ngưỡng tin cậy . . . . .	32
4.2 Đánh giá mô hình CNN . . . . .	33

---

4.2.1 Thiết lập các trọng số cho mô hình . . . . .	33
4.2.2 Kết quả thực nghiệm với mô hình CNN . . . . .	35
<b>5 Demo nhận dạng cá từ webcam và tra cứu Wikipedia</b>	<b>37</b>
5.1 Khởi Tạo và Cấu Hình Hệ Thống . . . . .	37
5.2 Vòng Lặp Xử Lý Thời Gian Thực . . . . .	38
5.2.1 Nhận dạng và Theo dõi YOLO + ByteTrack . . . . .	38
5.2.2 Ổn định Nhãn và Bounding Box . . . . .	39
5.2.3 Cơ Chế Auto-Freeze . . . . .	39
5.2.4 Tra cứu Wikipedia và Hiển Thị Thông Tin . . . . .	40
5.2.5 Hiển thị thông tin lên màn hình . . . . .	41
<b>6 Kết luận và Hướng phát triển</b>	<b>42</b>
6.1 Kết luận . . . . .	42
6.2 Hướng phát triển . . . . .	43
<b>Tài liệu tham khảo</b>	<b>44</b>

## **PHÂN CÔNG NHIỆM VỤ**

<b>Công việc</b>	<b>Người phụ trách</b>
Phân chia nhiệm vụ	Đặng Hải Bình
Tìm tài liệu tham khảo	Tất cả các thành viên
Tìm hiểu tổng quan và trực quan hóa bộ dữ liệu	Đặng Hải Bình
Nghiên cứu, xây dựng và huấn luyện mô hình YOLO	Đỗ Thị Mây, Nguyễn Quốc Hiệu
Tìm hiểu, xây dựng và huấn luyện mạng nơ-ron tích chập (CNN)	Chu Thị Mai Duyên
Demo nhận dạng cá từ webcam và tra cứu Wikipedia	Nguyễn Trọng Đức
Hoàn thiện, chỉnh sửa báo cáo	Tất cả các thành viên
Hoàn thiện chỉnh sửa slide	Tất cả các thành viên

# Mục lục cho hình ảnh

Hình 2.1 Mẫu ảnh từ các họ cá khác nhau . . . . .	10
Hình 2.2 Tetraodontidae (Cá nóc) và Diodontidae (Cá nóc gai) . . . . .	11
Hình 2.3 Từ trái qua phải, Cá bơn Petrale (eojor), Cá bơn phù thủy (Glcyn), Cá bơn sao (plste), đều thuộc họ Pleuronectidae . . . . .	12
Hình 2.4 Trực quan hóa dữ liệu số lượng ảnh các họ cá . . . . .	13
Hình 2.5 80% số lượng ảnh đến từ 76 họ cá . . . . .	14
Hình 2.6 Biểu đồ 2D Histogram của kích thước ảnh các họ cá . . . . .	15
Hình 2.7 Phân phối tỷ lệ khung hình (width/height) . . . . .	15
Hình 2.8 Phân phối độ phân giải ảnh . . . . .	16
Hình 2.9 Mẫu ảnh cá có kèm Bounding box . . . . .	17
 Hình 4.1 Đường cong Precision-Confidence (Độ chính xác theo Ngưỡng tin cậy) trên tập kiểm thử . . . . .	32
Hình 4.2 Sự biến thiên độ chính xác và hàm mất mát trên tập train theo từng epoch.	35
Hình 4.3 Top 15 cặp loài cá bị phân loại sai nhiều nhất trên tập validation. . . . .	36
 Hình 5.1 Chế độ Auto-Freeze giữ nguyên frame . . . . .	40
Hình 5.2 Tra cứu thông tin loài cá từ Wikipedia. . . . .	40

# Mục lục cho bảng

Bảng 2.1 Mô tả các trường thông tin trong file fishnet_annotations.csv	11
Bảng 2.2 5 Họ Phổ Biến Nhất . . . . .	12
Bảng 2.3 5 Họ Hiếm Gặp Nhất . . . . .	12
Bảng 2.4 Thống kê phân bố dữ liệu theo mức độ phổ biến của họ cá . . . . .	14
Bảng 4.1 Kết quả đánh giá mô hình YOLO trên tập test và tập valid . . . . .	30
Bảng 4.2 Hiệu suất mô hình YOLOv8 theo độ phổ biến của các họ cá . . . . .	31
Bảng 4.3 Cấu trúc và vai trò các lớp trong phần head của mô hình CNN . . . . .	34
Bảng 4.4 Chiến lược huấn luyện hai giai đoạn cho mô hình CNN . . . . .	34
Bảng 4.5 Kết quả đánh giá tổng thể của mô hình CNN trên tập validation . . . . .	35

## LỜI NÓI ĐẦU

Việc nhận dạng và phân loại loài cá hiện nay đòi hỏi chuyên môn cao và thường dựa vào kinh nghiệm cá nhân, dễ dẫn đến sai sót trong cả nghiên cứu lẫn hoạt động đánh bắt thủy sản. Để giải quyết bài toán này, nhóm nghiên cứu đã phát triển hệ thống nhận diện tự động dựa trên các mô hình tiên tiến như YOLO và CNN. Mục tiêu của dự án là cung cấp giải pháp nhanh chóng và chính xác, hỗ trợ đa dạng đối tượng từ nhà sinh học, các trại nuôi trồng đến ngư dân đánh bắt xa bờ, giúp giảm thiểu rủi ro khai thác nhầm/khai thác quá mức loài quý hiếm. Hướng tới tương lai, chúng tôi đặt mục tiêu đưa công nghệ này hoạt động offline trên nền tảng di động, mang lại giá trị thực tiễn cao nhất cho người dùng trong mọi điều kiện môi trường.

Báo cáo này, thuộc khuôn khổ môn học Nhập môn Trí tuệ Nhân tạo (MAT3508), tập trung làm rõ về quy trình khai phá và phân tích dữ liệu (EDA), phương pháp xây dựng và tối ưu hóa các mô hình nhận diện họ cá (CNN, YOLO), cũng như kỹ thuật xử lý ảnh với OpenCV để hoàn thiện hệ thống hỗ trợ tra cứu thông tin loài cá nhanh chóng và chính xác.

Do nhiều yếu tố hạn chế cũng như các nguyên nhân khách quan nên những phần tìm hiểu của nhóm có thể có nhiều thiếu sót. Nhóm luôn cố gắng tiếp thu những nhận xét, sự đóng góp ý kiến để từ đó có thể tiến bộ hơn.

Xin chân thành cảm ơn!

Hà Nội, Ngày 30 tháng 11 năm 2025

## Phần 1

### GIỚI THIỆU

#### 1.1 Tóm tắt dự án

Trong những năm gần đây, nhu cầu giám sát và quản lý quần thể cá trong các hệ sinh thái tự nhiên, ao nuôi công nghệ cao và môi trường nghiên cứu khoa học ngày càng tăng. Việc thu thập, phân loại và đánh giá thông tin về các loài cá truyền thống dựa vào quan sát thủ công tốn nhiều thời gian, chi phí nhân lực lớn, dễ sai lệch và không đáp ứng được yêu cầu xử lý dữ liệu lớn hay theo dõi thời gian thực.

Đề tài này hướng tới xây dựng một hệ thống nhận dạng và phân loại cá tự động với khả năng xử lý ảnh tĩnh theo thời gian thực, nhằm tối ưu hóa giám sát, nghiên cứu và giáo dục. Hệ thống bao gồm hai thành phần chính:

1. **Phân loại loài cá từ ảnh tĩnh:** Áp dụng YOLOv8 để phát hiện vị trí và phân loại cá trong từng khung hình và mô hình CNN, giúp xác định chính xác loài cá trong từng ảnh đầu vào, từ đó hỗ trợ thống kê và quản lý quần thể.
2. **Tích hợp tri thức mở từ Wikipedia:** Khi xác định loài cá, hệ thống tự động truy vấn dữ liệu sinh học, đặc điểm hình thái và hình ảnh minh họa từ Wikipedia API, cung cấp thông tin phong phú cho người dùng.

Nhờ kết hợp cả nhận dạng chính xác và cung cấp dữ liệu sinh học tức thời, hệ thống mang lại giá trị cao cho nghiên cứu, quản lý thủy sản, bảo tồn đa dạng sinh học và giáo dục trực quan.

#### 1.2 Bài toán đặt ra

Hiện nay, việc giám sát và phân loại các loài cá chủ yếu dựa vào quan sát thủ công. Phương pháp này tồn tại nhiều hạn chế: tốn thời gian, không đồng nhất, dễ sai lệch, khó xử lý lượng dữ liệu lớn và không đáp ứng được nhu cầu giám sát liên tục theo thời gian thực. Trong bối cảnh số lượng dữ liệu hình ảnh và video tăng nhanh, nhu cầu có một giải pháp tự

động, chính xác và hiệu quả trở nên cấp thiết.

### 1.2.1 Vấn đề

Vấn đề đặt ra là làm thế nào để xây dựng một hệ thống tự động có khả năng thực hiện được các yêu cầu sau:

1. Nhận diện và phân loại chính xác các loài cá từ ảnh và video.
2. Hoạt động hiệu quả trong môi trường dưới nước phức tạp, nơi ánh sáng, góc nhìn và vị trí cá thay đổi liên tục.
3. Kết hợp thông tin sinh học, đặc điểm hình thái và hình ảnh minh họa, phục vụ nghiên cứu, quản lý và giáo dục.

### 1.2.2 Động lực nghiên cứu

**Trong thủy sản và nông nghiệp công nghệ cao:** Tự động giám sát quần thể cá, phát hiện sớm thay đổi bất thường, tiết kiệm chi phí nhân công và nâng cao hiệu quả quản lý ao nuôi.

**Trong bảo tồn và nghiên cứu sinh học:** Hỗ trợ phân loại loài cá trong môi trường tự nhiên, xử lý dữ liệu dài hạn phục vụ nghiên cứu hành vi, phân bố và mô hình quần thể.

**Trong giáo dục và ứng dụng công cộng:** Xây dựng hệ thống hướng dẫn thông minh trong thủy cung, viện nghiên cứu, giúp người dùng tra cứu thông tin trực quan thông qua camera.

### 1.2.3 Ý nghĩa thực tiễn

- Khẳng định khả năng ứng dụng của mô hình Deep Learning hiện đại trong môi trường dưới nước vốn rất phức tạp.
- Đặt nền tảng cho các nghiên cứu nâng cao: theo dõi cá (tracking), phân tích hành vi hoặc giám sát sinh thái quy mô lớn.
- Cung cấp một giải pháp tích hợp, từ nhận dạng đến truy xuất tri thức, góp phần thúc đẩy nghiên cứu đa ngành và các ứng dụng giáo dục thông minh.

## Phần 2

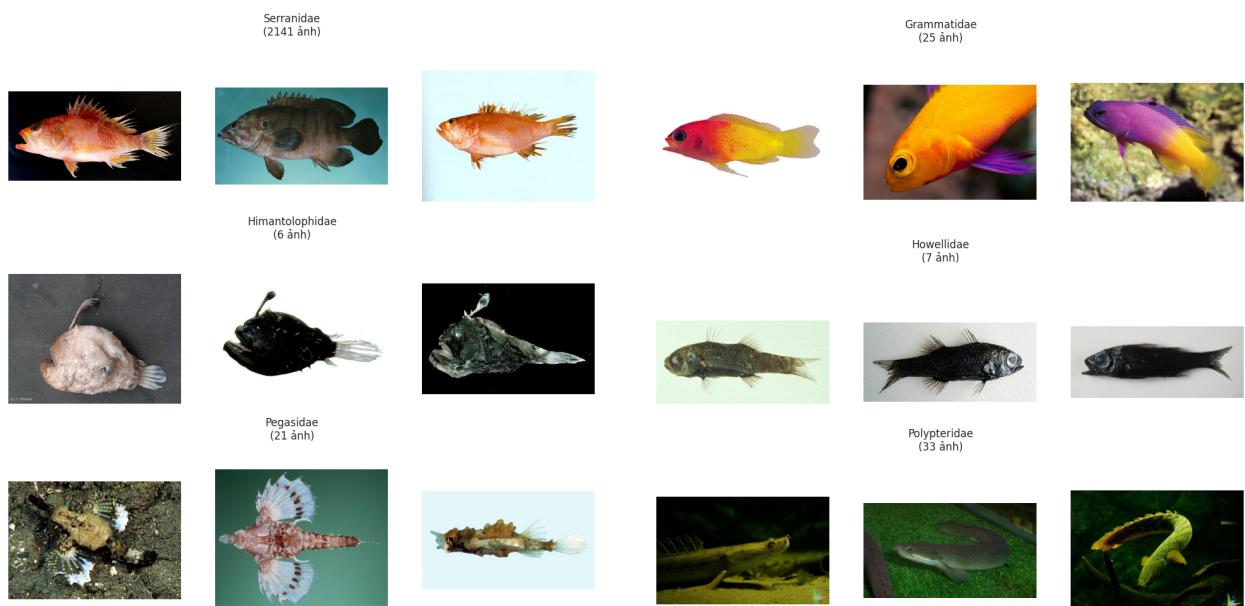
### TỔNG QUAN VÀ TRỰC QUAN HÓA DỮ LIỆU

#### 2.1 Tổng quan dữ liệu

Dữ liệu nhóm sử dụng trong nghiên cứu gồm 2 tập chính:

- Tập các file .txt về vị trí Bounding box trong từng ảnh, được phân theo họ cá.
- Ảnh các họ cá của bộ dữ liệu Fishnet, được thu thập chính từ hai cơ sở dữ liệu trực tuyến lớn là *FishBase* và *iNaturalist*.

**Hình 2.1: Mẫu ảnh từ các họ cá khác nhau**



Bộ dữ liệu ảnh ban đầu gồm 95032 ảnh từ 570 họ cá, sau quy trình lọc thủ công để loại bỏ các hình ảnh không hợp lệ (*anh không chụp cá, bản phác thảo xương cá, cá ở giai đoạn ấu trùng, hình ảnh tiền xu hoặc tem có hình cá*) và xác định Bounding box trong từng ảnh, bộ dữ liệu Fishnet cuối cùng nhóm sử dụng gồm 84680 ảnh (loại bỏ 10352 ảnh không có Bounding box), tương ứng với 84680 file Bounding box từ 463 họ cá.

Để tiện cho việc phân tích khám phá dữ liệu, nhóm thực hiện đọc và lưu thông tin về

vị trí Bounding box thành file `fishnet_annotations.csv`, gồm các trường:

Tên trường	Mô tả
<code>bbox_id</code>	Khóa chính - ID duy nhất cho mỗi bbox
<code>file_path</code>	Đường dẫn đến file .txt annotation
<code>image_name</code>	Tên file (không có đuôi .txt) - dùng để liên kết với ảnh gốc
<code>family_name</code>	Tên họ cá (bao gồm 463 họ khác nhau)
<code>bbox_index</code>	Thứ tự bbox trong file (0 = đầu tiên, 1 = thứ hai...)
<code>x_min, y_min, x_max, y_max</code>	Tọa độ của bounding box trên ảnh

**Bảng 2.1:** Mô tả các trường thông tin trong file `fishnet_annotations.csv`

File `fishnet_annotations.csv` gồm 93399 bản ghi chú về vị trí Bounding box của từng ảnh, các bản ghi đều đầy đủ và không chứa bất kỳ giá trị rỗng (non-null) nào, đảm bảo tính toàn vẹn cho các bước phân tích tiếp theo.

#### Một số ảnh họ cá dễ nhầm lẫn

Khi thực hiện xem kỹ hơn ảnh của từng họ cá, nhóm nhận thấy, trong bộ dữ liệu có các họ cá khác nhau nhưng chúng lại có ngoại hình khá giống nhau, hoặc, các loài cá trong cùng một họ nhưng chúng lại có hình dạng khác nhau. Điều này có thể ảnh hưởng đến khả năng học của mô hình trong quá trình huấn luyện.

**Hình 2.2:** Tetraodontidae (Cá nóc) và Diodontidae (Cá nóc gai)



Mặc dù cả hai họ cá nóc trên có hình thái bên ngoài khá giống nhau, đều phồng lên khi gặp nguy hiểm, chúng lại có điểm khác biệt là Cá nóc nhím có gai lớn rõ rệt, trong khi họ Cá nóc thường da trơn hoặc gai rất nhỏ.

**Hình 2.3:** Từ trái qua phải, Cá bơn Petrale (*eojo*), Cá bơn phù thủy (*Glcyn*), Cá bơn sao (*plste*), đều thuộc họ Pleuronectidae



Trong họ Cá bơn mắt phải (*Pleuronectidae*), ba loài được lựa chọn khảo sát là **eojo** (*Eopsetta jordani*), **Glcyn** (*Glyptocephalus cynoglossus*), và **plste** (*Platichthys stellatus*) thể hiện sự đa dạng hình thái, mặc dù cùng chia sẻ đặc điểm bất đối xứng cơ thể.

Về mặt cấu trúc sọ và hàm, *G. cynoglossus* (*Glcyn*) sở hữu khoang miệng rất nhỏ cùng phần đầu thon gọn. Ngược lại, *E. jordani* (*eojo*) có cấu trúc hàm rộng, khỏe với hàm dưới kéo dài. Sự khác biệt về kích thước miệng là đặc trưng quan trọng nhất để phân loại hai loài này trên dữ liệu hình ảnh.

Riêng đối với **plste** (*P. stellatus*), khác với quy luật chung của họ Pleuronectidae (mắt nằm bên phải), *P. stellatus* thể hiện tính đa hình cao về hướng mắt (có thể nằm bên trái hoặc phải). Tuy nhiên, loài này lại có các dải màu đen-trắng tương phản cao trên vây lưng và vây hậu môn, cùng bề mặt da thô ráp phủ gai hình sao. Đây là các dấu hiệu giúp tách biệt **plste** khỏi hai loài còn lại bất kể biến thể về hướng mắt.

## 2.2 Trực quan hóa dữ liệu

### 2.2.1 Số lượng ảnh các họ cá

Phân tích trên bộ dữ liệu FishNet cho thấy một sự chênh lệch rất lớn về số lượng ảnh giữa các họ cá, với số lượng outliers phát hiện được bằng IQR là 58 họ.

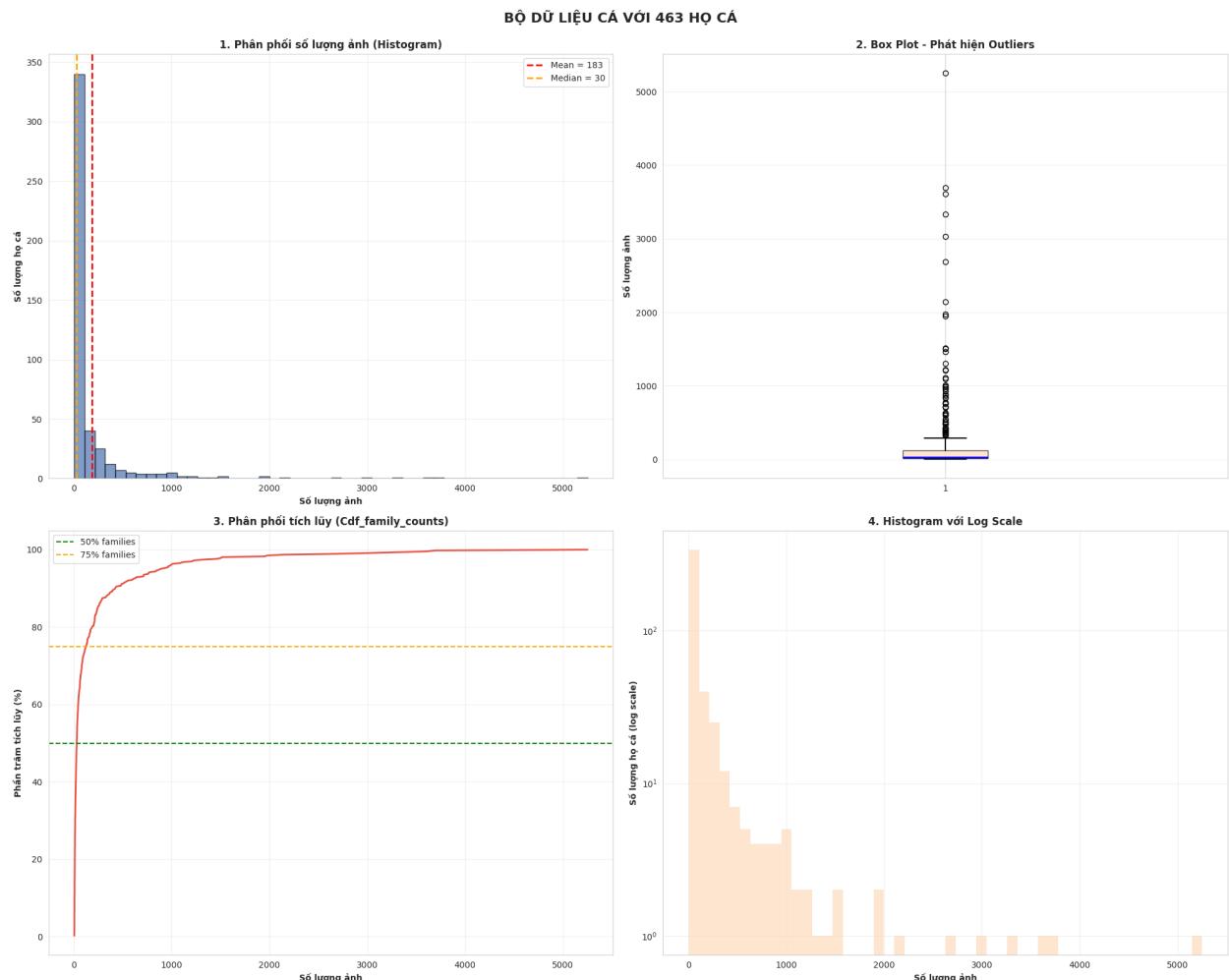
**Bảng 2.2: 5 Họ Phổ Biến Nhất**

Họ cá	Số lượng ảnh	Bounding box
Labridae	5251	5701
Cyprinidae	3694	4109
Pomacentridae	3609	4518
Centrarchidae	3333	3421
Gobiidae	3027	3127

**Bảng 2.3: 5 Họ Hiếm Gặp Nhất**

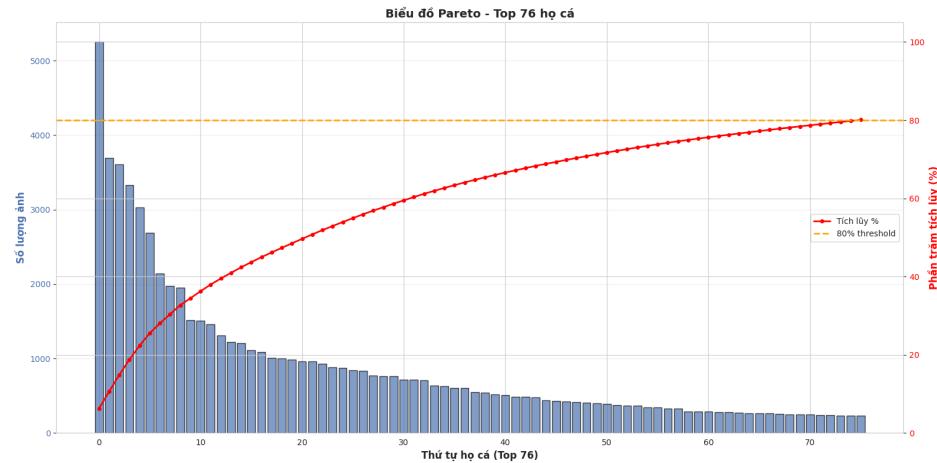
Họ cá	Số lượng ảnh	Bounding box
Banjosidae	4	4
Diplomystidae	4	4
Spatelloididae	4	7
Indostomidae	4	4
Gonorynchidae	4	5

Sau khi thực hiện tính toán các chỉ số thống kê cơ bản về số lượng ảnh trên các họ cá, nhóm nhận thấy bộ dữ liệu có sự chênh lệch rất lớn giữa giá trị trung bình (182.89) và trung vị (30), cho thấy rằng một vài họ cá có số lượng ảnh cực lớn đang kéo giá trị trung bình lên cao, trong khi đa số các loài khác đều có dữ liệu không nhiều. Độ lệch dương (5.76) xác nhận rằng phân bố dữ liệu bị lệch phải nghiêm trọng, hay còn gọi là "phân phối đuôi dài" (long-tail distribution), nơi một số ít lớp chiếm phần lớn dữ liệu ảnh, trong khi đa số các lớp khác có rất ít mẫu.



**Hình 2.4:** Trực quan hóa dữ liệu số lượng ảnh các họ cá

Áp dụng nguyên tắc Pareto (80/20) cho thấy 80% tổng số ảnh của bộ dữ liệu chỉ đến từ 76 họ cá, chiếm vỏn vẹn 16.4% tổng số họ. Điều này cho thấy sự tập trung dữ liệu rất cao vào một nhóm nhỏ các lớp.



**Hình 2.5:** 80% số lượng ảnh đến từ 76 họ cá

Để định lượng rõ hơn và xác định ảnh hưởng của long-tailed, nhóm phân loại các họ cá thành ba nhóm dựa trên đóng góp của chúng vào tổng số lượng ảnh: Common (Phổ biến), Medium (Trung bình), và Rare (Hiếm) để đánh giá hiệu suất của mô hình sau khi huấn luyện.

Từ bảng 2.4 ta có thể thấy Chỉ vỏn vẹn 5 họ cá đã đóng góp tới 25% dữ liệu hình ảnh. Trong khi đó, 404 họ cá thuộc nhóm "Rare" (chiếm tới 87.26% tổng số họ) lại phải chia nhau 25% lượng dữ liệu còn lại.

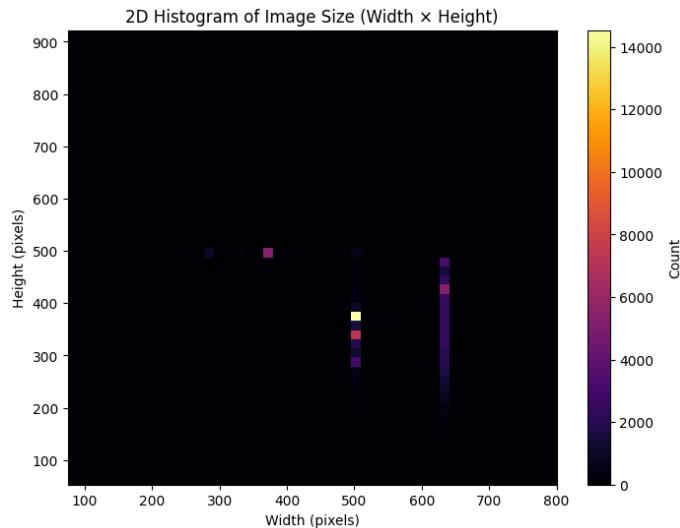
Nhóm	Số lượng họ cá	Tỷ lệ (%)	Mô tả đóng góp
<b>Common</b>	5	1.08%	Các họ cá hàng đầu chiếm 25% tổng số ảnh.
<b>Medium</b>	54	11.66%	Các họ cá tiếp theo chiếm 50% tổng số ảnh.
<b>Rare</b>	404	87.26%	Các họ cá còn lại chỉ chiếm 25% tổng số ảnh.

**Bảng 2.4:** Thống kê phân bố dữ liệu theo mức độ phổ biến của họ cá

## 2.2.2 Kích thước ảnh và Bounding box trong bộ dữ liệu

### Kích thước ảnh

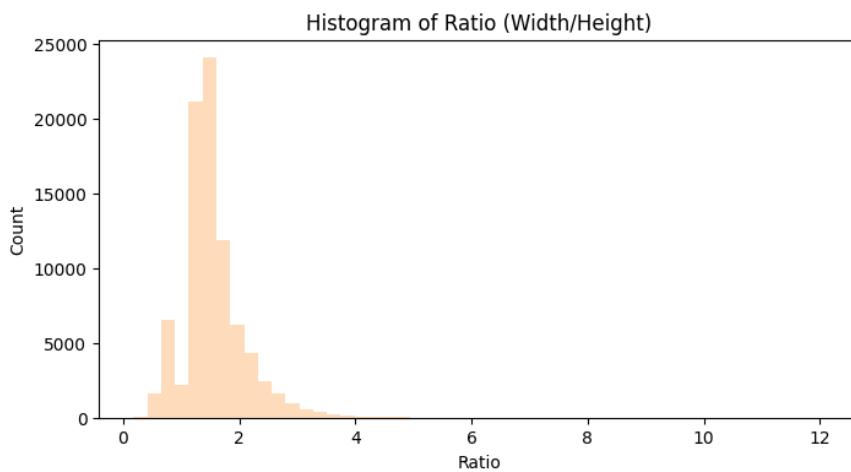
Kích thước của ảnh trong bộ dữ liệu có xu hướng tập trung vào một số kích thước nhất định. Trong đó, ba kích thước phổ biến nhất chiếm một tỷ lệ đáng kể trong bộ dữ liệu là: 500x375 (15.32%), 375x500 (5.96%) và 500x333 (5.16%).



**Hình 2.6:** Biểu đồ 2D Histogram của kích thước ảnh các họ cá

Biểu đồ 2D Histogram cho thấy kích thước ảnh (*width x height*) không phân bố ngẫu nhiên, mà chủ yếu tập trung vào 1 số kích thước chính. Điểm vàng sáng trong biểu đồ thể hiện đa số ảnh có kích thước khoảng 500x375.

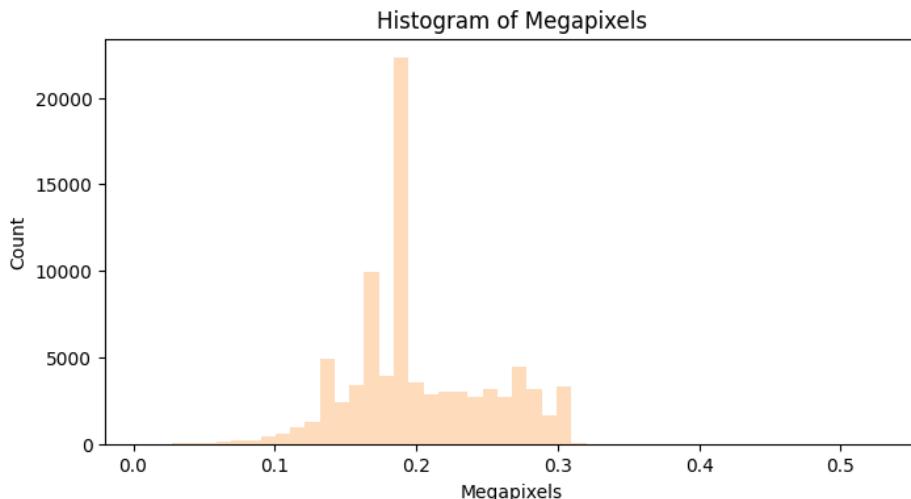
Bên cạnh đó, có 2 vệt dọc trong biểu đồ, với width bằng 500px và 640px, cho thấy các ảnh có chiều rộng cố định, trong khi chiều cao lại biến đổi linh hoạt. Điều này có thể do dữ liệu được thu thập từ 2 nguồn chính đã đề cập ở trên. Từ việc phân tích trên, nhóm quyết định sử dụng kích thước 512x512 cho các ảnh làm đầu vào của ảnh trong YOLO, sử dụng các phép biến đổi hình học để tăng cường dữ liệu, cũng như điều chỉnh hàm mất mát để gán trọng số cao hơn cho các lỗi dự đoán trên các lớp hiếm trong quá trình huấn luyện mô hình CNN.



**Hình 2.7:** Phân phối tỷ lệ khung hình (width/height)

Tuy nhiên, về tỷ lệ khung hình, các tỷ lệ phổ biến nhất là  $4/3$  và  $3/4$ , nên khi thay đổi kích thước tất cả ảnh thành một định dạng vuông chung, có thể làm méo hình dáng của cá, ảnh hưởng tới khả năng nhận dạng chính xác của mô hình.

#### Độ phân giải hình ảnh (Megapixels)



**Hình 2.8:** Phân phối độ phân giải ảnh

Dựa trên kết quả thống kê mô tả từ tập dữ liệu (với 84680 ảnh), nhóm rút ra các nhận xét như sau về độ phân giải của các ảnh trên toàn bộ dữ liệu:

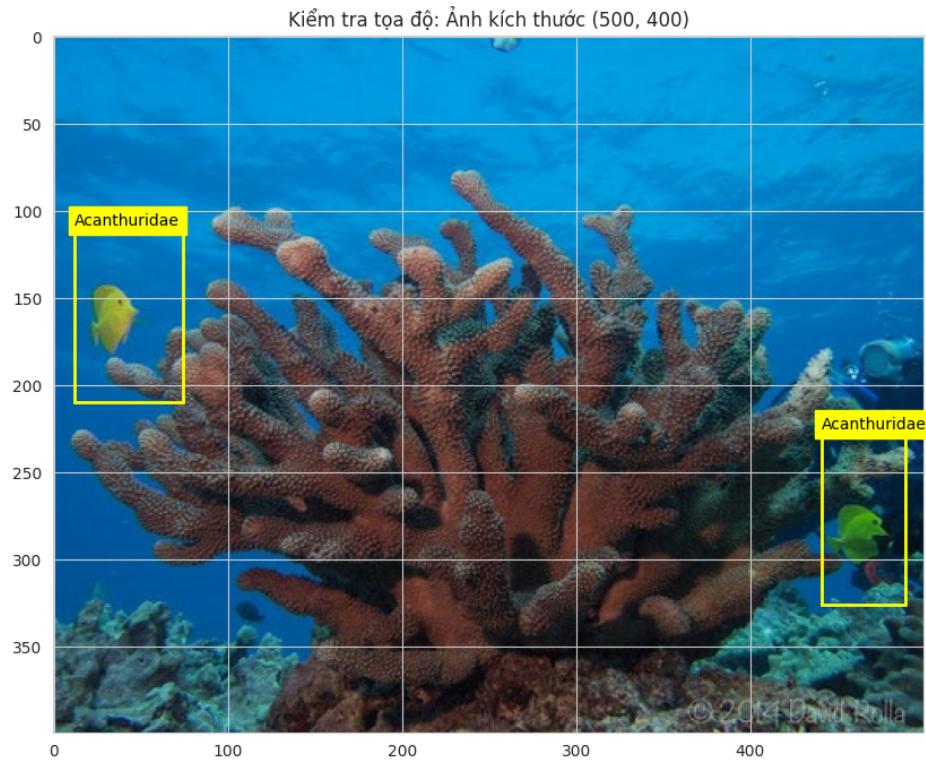
- Độ phân giải tổng thể thấp: Giá trị Megapixels trung bình của bộ dữ liệu chỉ đạt mức 0.20 MP. Ngay cả các ảnh có chất lượng cao nhất cũng không vượt quá 0.53 MP. Điều này cho thấy dữ liệu nguồn không phải là ảnh chất lượng cao (HD/Full HD).
- Sự đồng nhất về kích thước: Độ lệch chuẩn rất nhỏ ( $\approx 0.05$ ) và khoảng từ phân vị (Q1-Q3) nằm trong vùng hẹp từ 0.1665 MP đến 0.2368 MP. Giá trị trung vị (0.1875 MP) tương ứng với kích thước phổ biến 500 x 375 pixels.

Với độ phân giải thấp, khối lượng tính toán mô hình sẽ giảm đáng kể. Tuy nhiên, độ phân giải thấp là một rào cản lớn cho bài toán Fine-grained Classification (Phân loại chi tiết). Các đặc trưng quan trọng để phân biệt các loài cá có ngoại hình giống nhau (như vân vảy, gai vây, màu mắt) có thể bị mờ hoặc mất đi, làm giảm độ chính xác của mô hình.

#### Kích thước và vị trí của Bounding box

Thống kê cho thấy, tỷ lệ trung bình diện tích của các Bounding box so với diện tích của toàn bộ ảnh là 46.7%, có nghĩa là, trung bình, con cá chiếm khoảng 46.7% diện tích của ảnh. Con số này cho thấy các đối tượng thường chiếm một phần đáng kể trong khung

hình, giúp mô hình dễ dàng "nhìn thấy" hơn. Mặc dù vậy, sự thay đổi lớn trong tỷ lệ này (từ gần 0% đến 100%) đòi hỏi mô hình phải có khả năng xử lý các đối tượng ở nhiều tỷ lệ khác nhau một cách hiệu quả.



**Hình 2.9:** Mẫu ảnh cá có kèm Bounding box

## Phần 3

### PHƯƠNG PHÁP VÀ TRIỂN KHAI

#### 3.1 Phương pháp

Phần này trình bày chi tiết các phương pháp, thuật toán và dữ liệu được sử dụng trong quá trình xây dựng hệ thống nhận diện và phân loại cá theo thời gian thực. Việc lựa chọn từng mô hình và kỹ thuật đều dựa trên đặc thù của bài toán thị giác máy tính dưới nước, nơi ánh sáng thay đổi liên tục, độ nhiễu cao và hình dạng cá đa dạng.

##### 3.1.1 Mô hình phân loại ảnh CNN

###### a. Tổng quan

**Convolutional Neural Network (CNN)** là một loại mạng nơ-ron nhân tạo chuyên biệt, được thiết kế để xử lý dữ liệu dạng lưới, chẳng hạn như hình ảnh. CNN được sử dụng rộng rãi trong nhận diện hình ảnh, phân loại ảnh, nhận dạng đối tượng, và nhiều bài toán thị giác máy tính khác.

###### b. Các lớp chính trong CNN

CNN gồm một chuỗi các lớp với các chức năng khác nhau:

###### **Convolutional Layer (Lớp tích chập) :**

- **Chức năng:** Tích chập là quá trình trích xuất các đặc trưng từ dữ liệu đầu vào bằng cách áp dụng các bộ lọc.

- **Cách hoạt động :**

- Bộ lọc là một ma trận nhỏ trượt qua dữ liệu đầu vào (thường là hình ảnh)

- Mỗi bước trượt, nó tính một phép nhân điểm giữa bộ lọc và phần tương ứng của dữ liệu đầu vào

- **Đầu ra:** Một ma trận đặc trưng chứa thông tin trích xuất từ dữ liệu.

###### **Pooling Layer (Lớp gộp) :**

- **Chức năng:** Giảm kích thước của ma trận đặc trưng, giúp giảm số lượng tham số và tăng tính bất biến với dịch chuyển và biến dạng

- **Loại phổ biến :**

- Max Pooling: Lấy giá trị lớn nhất trong vùng được chọn
- Average Pooling: Lấy giá trị trung bình trong vùng được chọn.

**Fully Connected Layer (Lớp kết nối đầy đủ):**

- **Chức năng:** Kết nối toàn bộ các neuron từ lớp trước vào lớp hiện tại.
- Đây là phần cuối của CNN, nơi các đặc trưng đã được trích xuất được sử dụng để dự đoán nhãn đầu ra (classification hoặc regression).

**Dự đoán :**

- Đầu vào: Một ảnh cần phân loại.
- Ảnh được truyền qua toàn bộ mạng CNN.
- Lớp cuối cùng trả về xác suất ảnh thuộc từng lớp.
- Lớp có xác suất cao nhất được chọn làm kết quả dự đoán.

**c. Các tham số tối ưu hóa**

**Các tham số tối ưu**

- Batch size:

- Định nghĩa: Là số lượng mẫu được sử dụng trong một lần cập nhật trọng số
- Batch size lớn:

- \* Ưu điểm: Ổn định hơn, tốc độ huấn luyện nhanh hơn
- \* Nhược điểm: Yêu cầu bộ nhớ lớn, khả năng tổng quát hóa kém hơn

- Batch size nhỏ:

- \* Ưu điểm: Ít bộ nhớ, khả năng tổng quát tốt hơn
- \* Nhược điểm: Training không ổn định, thời gian lâu hơn

- Thông thường chọn batch size từ 32-256 tùy vào bài toán

- Epochs:

- Định nghĩa : Một epoch là khi toàn bộ dữ liệu huấn luyện được truyền qua mô hình một lần.
- Epochs quá ít : Mô hình chưa học đủ (underfitting)
- Epochs quá nhiều : có thể dẫn đến overfitting (mô hình ghi nhớ thay vì học khái quát).
- Lựa chọn phổ biến: Thường kiểm soát epochs dựa trên early stopping hoặc theo dõi độ chính xác (accuracy) và hàm mất mát (loss).
- Optimizer:
  - Định nghĩa: Là thuật toán tối ưu hóa trọng số trong mạng, giúp giảm giá trị của hàm mất mát (loss function).
  - Vai trò :
    - \* Quyết định tốc độ và hiệu quả của việc học.
    - \* Các optimizer phổ biến:
      - + SGD (Stochastic Gradient Descent): Cơ bản nhưng chậm, có thể cải thiện bằng Momentum hoặc Nesterov.
      - + Adam (Adaptive Moment Estimation): Tối ưu hóa hiện đại, tự động điều chỉnh learning rate, hiệu quả trên nhiều loại dữ liệu.
      - + RMSProp: Phù hợp cho dữ liệu chuỗi thời gian hoặc không gian.
  - Lựa chọn phổ biến: Adam thường là lựa chọn mặc định vì tốc độ hội tụ nhanh và ổn định.

#### c. Vai trò CNN

Convolutional Neural Networks (CNNs) đóng vai trò cốt lõi trong bài toán phân loại cá do khả năng học và trích xuất các đặc trưng thị giác quan trọng giúp phân biệt giữa các loài. Cụ thể, CNN hỗ trợ hệ thống theo các hướng sau:

- **Trích xuất đặc trưng hình thái tự động:** CNN có khả năng tự học các đặc trưng quan trọng như:
  - \* hình dạng tổng thể của thân cá,
  - \* kích thước và vị trí các vây,

- \* màu sắc đặc trưng và hoa văn trên thân,
- \* các sọc, đốm hoặc kết cấu bề mặt.

Những đặc trưng này đặc biệt quan trọng vì nhiều loài cá có ngoại hình gần giống nhau, chỉ khác biệt ở các chi tiết tinh tế mà các phương pháp thủ công khó trích xuất.

- **Giảm số lượng tham số so với Fully Connected Networks:** Việc sử dụng các bộ lọc tích chập giúp mô hình học hiệu quả hơn, giảm số tham số cần tối ưu, đồng thời hạn chế overfitting khi dữ liệu không quá lớn—một ưu điểm quan trọng trong bài toán phân loại cá với số lượng ảnh theo từng loài thường không đồng đều.
- **Khả năng xử lý dữ liệu 2D mạnh mẽ:** CNN đặc biệt phù hợp với hình ảnh cá vì chúng nắm bắt tốt mối quan hệ không gian giữa các pixel—ví dụ:
  - \* rìa vây cá thường có dạng cong,
  - \* mắt và miệng có vị trí cố định tương đối,
  - \* hoa văn trải dài dọc thân.

Điều này giúp CNN phân biệt tốt các loài cá trong điều kiện thực tế.

- **Tính ổn định với nhiều và biến dạng môi trường nước:** Nhờ cơ chế dịch chuyển bất biến (translation invariance) của các lớp pooling, CNN vẫn giữ được độ chính xác trong môi trường nước vốn có:
  - \* ánh sáng thay đổi,
  - \* chuyển động nhanh,
  - \* độ mờ và nhiễu do nước gây ra.

- **Hỗ trợ phân loại bổ sung khi YOLOv8 chưa đủ chi tiết:** Trong một số trường hợp YOLOv8 chỉ phát hiện được vị trí cá nhưng phân loại chưa chính xác, CNN có thể đóng vai trò bộ phân loại hai bước (two-stage classification) để tăng độ tin cậy của hệ thống.

#### d. Lý do lựa chọn CNN

Việc lựa chọn Convolutional Neural Networks (CNN) cho bài toán phân loại cá xuất phát từ những đặc điểm đặc thù của dữ liệu và yêu cầu nhận dạng loài. Những lý do chính bao gồm:

- **Khả năng học đặc trưng thị giác phức tạp:** Các loài cá thường có sự khác biệt về hoa văn, màu sắc, hình dạng vây, cấu trúc thân và các chi tiết nhỏ khác. CNN có khả năng tự động học các đặc trưng này thông qua nhiều tầng tích chập, giúp phân biệt tốt giữa các loài cá có hình thái tương đồng.
- **Tính bền vững trước biến dạng môi trường:** Hình ảnh cá trong môi trường tự nhiên hoặc bể nuôi thường chịu ảnh hưởng của:
  - \* ánh sáng không đồng đều,
  - \* nhiễu do bụi nước,
  - \* chuyển động nhanh của cá,
  - \* phản xạ và khúc xạ ánh sáng.
 Nhờ translation invariance và pooling layers, CNN vẫn giữ được độ chính xác cao trong các điều kiện trên.
- **Giảm số lượng tham số so với mạng Fully Connected:** Việc chia sẻ trọng số giữa các kernel giúp CNN học hiệu quả hơn, yêu cầu ít dữ liệu hơn và giảm nguy cơ overfitting. Đây là ưu điểm quan trọng vì số lượng ảnh của từng loài cá thường không đồng đều.
- **Tối ưu cho bài toán phân loại fine-grained:** Nhiều loài cá chỉ khác nhau ở những điểm rất nhỏ như:
  - \* độ cong của vây,
  - \* tỉ lệ chiều dài thân,
  - \* màu sắc nhạt hoặc các đốm nhỏ.
 CNN đặc biệt phù hợp cho các dạng phân loại tinh vi như vậy nhờ khả năng học đặc trưng đa tầng (từ low-level đến high-level).
- **Tích hợp tốt với YOLOv8 trong hệ thống hai giai đoạn:** Khi YOLOv8 phát hiện được vùng chứa cá nhưng phân loại chưa chắc chắn (đặc biệt với cá nhỏ hoặc bị che khuất), CNN có thể đảm nhận bước phân loại bổ sung, tăng độ tin cậy tổng thể của hệ thống.
- **Hỗ trợ triển khai trên nhiều nền tảng:** Các mô hình CNN có thể được tối ưu và triển khai trên:

- \* máy tính cấu hình trung bình,
  - \* GPU mạnh,
  - \* thiết bị IoT hoặc Edge (như Jetson Nano, Raspberry Pi),
- nhờ các phiên bản nhẹ như MobileNetV3 hoặc EfficientNet-lite.

Những đặc điểm trên cho thấy CNN là lựa chọn phù hợp và hiệu quả cho bài toán phân loại cá trong hệ thống nhận diện thời gian thực.

### 3.1.2 Mô hình phát hiện đối tượng YOLOv8

#### a. Tổng quan

YOLOv8 (You Only Look Once version 8) là mô hình phát hiện đối tượng dạng one-stage, tức toàn bộ ảnh được xử lý trong một lần suy diễn. Mô hình này cân bằng tốt giữa độ chính xác và tốc độ, đạt khoảng 50–150 FPS trên GPU hiện đại, phù hợp cho các ứng dụng nhận diện cá theo thời gian thực trong ao nuôi hoặc môi trường tự nhiên.

#### b. Kiến trúc YOLOv8

- **Backbone – C2f Block:** Backbone sử dụng các khối C2f kết hợp Bottleneck và Cross-Stage Partial để trích xuất các đặc trưng hình dạng, màu sắc, hoa văn của cá, đồng thời giữ chi phí tính toán ở mức thấp.
  - + **Chức năng:** Trích xuất đặc trưng từ ảnh đầu vào để phục vụ phát hiện và phân loại.
  - + **Cách hoạt động:** Các khối C2f cho phép thông tin từ nhiều tầng được kết hợp, giữ các đặc trưng chi tiết và giảm mất mát thông tin.
  - + **Đầu ra:** Feature map chứa các đặc trưng quan trọng của cá để truyền tới Neck.
- **Neck – FPN + PAN:** Neck kết hợp FPN và PAN nhằm tổng hợp thông tin từ nhiều tầng feature map.
  - + **FPN:** Hỗ trợ phát hiện cá có kích thước lớn và trung bình.
  - + **PAN:** Cải thiện khả năng phát hiện cá nhỏ hoặc bị che khuất.
  - + **Đầu ra:** Feature map được tổng hợp từ nhiều tầng, sẵn sàng cho Head dự đoán.
- **Head – Anchor-free:** Head không sử dụng anchor box cố định, giúp mô hình linh hoạt phát hiện các loài cá có kích thước và hình dạng đa dạng.
  - + **Chức năng:** Dự đoán bounding box, nhãn lớp và độ tin cậy.

- + **Cách hoạt động:** Head dự đoán trực tiếp các tọa độ và nhãn mà không cần anchor, giảm sai số cho cá nhỏ và đa dạng hình dạng.
- + **Đầu ra:** Danh sách bounding box với nhãn và confidence cho từng cá thể trong ảnh.

#### c. Quy trình xử lý ảnh

Quy trình suy diễn YOLOv8 diễn ra theo các bước chính:

- **Chuẩn hóa ảnh:** Resize ảnh đầu vào về  $512 \times 512$  để đồng nhất dữ liệu, đảm bảo mô hình hoạt động ổn định.
- **Trích xuất đặc trưng:** Mạng Backbone xử lý ảnh để lấy các đặc trưng quan trọng, bao gồm hình dạng, màu sắc và hoa văn của cá.
- **Tổng hợp đặc trưng:** Neck kết hợp các feature map từ nhiều tầng, giúp phát hiện cá với kích thước và vị trí đa dạng.
- **Dự đoán cuối cùng:** Head dự đoán bounding box và nhãn loài cá, đồng thời tính confidence cho từng đối tượng.

#### d. Hàm mất mát

YOLOv8 tối ưu hóa thông qua ba thành phần chính trong hàm mất mát. **Bounding Box Loss** sử dụng CIoU Loss để đo lường sai số giữa bounding box dự đoán và ground truth, tính toán không chỉ IoU mà còn bao gồm khoảng cách giữa tâm hai khung, tỷ lệ khung và độ chênh lệch hình dạng, được biểu diễn bởi công thức:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v$$

Nhờ đó, mô hình học được cách dự đoán các bounding box chính xác hơn.

**Class Loss** được tính bằng Binary Cross-Entropy, đánh giá độ chính xác trong việc phân loại loài cá, đảm bảo mỗi đối tượng được nhận diện đúng loại. Đồng thời, **Objectness Loss** xác định xem một vùng dự đoán cụ thể có chứa cá hay không, giúp mô hình phân biệt các vùng nền với các cá thể thực tế.

Sự kết hợp của ba thành phần này cho phép YOLOv8 tối ưu đồng thời vị trí, nhãn và xác suất tồn tại của đối tượng trong mỗi khung ảnh, nâng cao hiệu quả nhận diện cá trong các môi trường phức tạp.

### e. Lý do lựa chọn YOLOv8

Mô hình YOLOv8 được lựa chọn chủ yếu nhờ khả năng xử lý nhanh, đáp ứng tốt yêu cầu thời gian thực (real-time). Với kiến trúc tối ưu và hiệu suất suy diễn cao, mô hình có thể phát hiện các cá thể trong luồng video trực tiếp mà không làm gián đoạn quá trình quan sát. Điều này đặc biệt quan trọng trong các ứng dụng giám sát ao nuôi hoặc môi trường tự nhiên, nơi tốc độ nhận diện ảnh hưởng trực tiếp đến khả năng phản ứng và phân loại cá.

Ngoài ra, YOLOv8 có khả năng nhận diện hiệu quả các cá thể với nhiều kích thước khác nhau, bao gồm cả cá nhỏ hoặc bị che khuất một phần. Nhờ cơ chế kết hợp các tầng feature map trong Neck (FPN + PAN) và head anchor-free, mô hình duy trì độ chính xác cao ngay cả trong điều kiện ánh sáng yếu hoặc nước đục, giúp cải thiện độ tin cậy của hệ thống.

## 3.2 Triển khai hệ thống

Phần này trình bày chi tiết quá trình triển khai hệ thống phân loại cá thời gian thực, bao gồm kiến trúc tổng thể, công cụ và thư viện sử dụng, quy trình xây dựng mô hình, tổ chức mã nguồn, và cơ chế vận hành trong môi trường thực tế. Hệ thống được thiết kế nhằm đáp ứng ba mục tiêu chính: (1) phát hiện cá từ ảnh, video hoặc camera trực tiếp, (2) phân loại chính xác loài cá bằng mô hình học sâu, và (3) liên kết nhanh đến trang thông tin sinh học tương ứng trên Wikipedia.

### 3.2.1 Kiến trúc tổng thể hệ thống

Hệ thống được triển khai theo mô hình pipeline hai giai đoạn, bao gồm:

- **Giai đoạn 1: YOLOv8 Detection** Xác định vị trí của cá trong ảnh hoặc từng khung hình của video. YOLOv8 cho phép phát hiện đa đối tượng trong thời gian thực với tốc độ >30 FPS.
- **Giai đoạn 2: CNN Classification** Vùng chứa cá (crop) được đưa vào mô hình CNN để phân loại loài. CNN giúp phân biệt những loài có hình thái tương đồng.
- **Giai đoạn 3: Truy xuất thông tin** Từ kết quả phân loại, hệ thống ánh xạ tên loài sang URL thông tin sinh học trên Wikipedia và hiển thị cho người dùng.

Kiến trúc này đáp ứng được yêu cầu nhận diện trong điều kiện môi trường phức tạp, đồng thời giữ được tính mở rộng nếu bổ sung loài mới.

### 3.2.2 Môi trường và công cụ triển khai

Hệ thống được xây dựng dựa trên các công cụ và thư viện sau:

- **Ngôn ngữ lập trình:** Python 3.10
- **Thư viện học sâu:**
  - \* PyTorch 2.0 — xây dựng và huấn luyện CNN
  - \* Ultralytics YOLOv8 — mô hình phát hiện đối tượng
- **Xử lý dữ liệu:** OpenCV, NumPy, Pandas
- **Trực quan hóa và đánh giá:** Matplotlib, Seaborn, TensorBoard
- **Giao diện người dùng (tùy chọn):** Flask hoặc Streamlit
- **Công cụ anotate dữ liệu:** LabelImg, Roboflow
- **Nguồn tài nguyên phần cứng:**
  - \* GPU: NVIDIA RTX 3060 / Tesla T4 (Google Colab)
  - \* CPU: Intel Core i7

### 3.2.3 Tiền xử lý và chuẩn hoá dữ liệu

Bộ dữ liệu gồm hình ảnh của nhiều loài cá trong môi trường tự nhiên và môi trường nuôi. Các bước tiền xử lý bao gồm:

- **Chuyển đổi kích thước:** ảnh được resize về  $224 \times 224$  (CNN) và  $512 \times 512$  (YOLOv8).
- **Chuẩn hoá pixel:** sử dụng chuẩn hoá mean–std của ImageNet.
- **Tăng cường dữ liệu (Data Augmentation):**
  - \* rotation  $0\text{--}30^\circ$
  - \* horizontal flip
  - \* color jitter
  - \* Gaussian noise
  - \* blur nhẹ

Mục tiêu là làm mô hình bền vững với biến đổi ánh sáng, màu nước và chuyển động của cá.

### 3.2.4 Huấn luyện mô hình

#### Huấn luyện YOLOv8

Mô hình YOLOv8m được lựa chọn do cân bằng tốt giữa tốc độ và độ chính xác.

- **Batch size:** 32
- **Epochs:** 50
- **Optimizer:** SGD (momentum 0.937)
- **Learning rate:** 0.01
- **Loss:** combination of box loss + cls loss + DFL loss

Tập dữ liệu được anotate theo chuẩn YOLO: mỗi ảnh có file TXT đi kèm chứa {class, x, y, w, h}.

#### Huấn luyện CNN

CNN có nhiệm vụ phân loại chính xác loài cá sau khi YOLO crop vùng chứa cá. Mô hình được sử dụng có thể là:

- ResNet-50
- EfficientNet-B0
- Custom CNN 5–10 layers

#### Thông số huấn luyện:

- Batch size: 64
- Epochs: 50–120
- Optimizer: Adam
- Loss: Cross-entropy

### 3.2.5 Tổ chức mã nguồn hệ thống

Cấu trúc mã nguồn được tổ chức theo chuẩn lập trình mô-đun để dễ mở rộng:

```
project_root/
  cnn/
    CNN.ipynb
```

```
  docs/
```

```

eda/
    README.md
    create_fish_annotation.ipynb
    eda_fish_data.ipynb
    read_original_image.ipynb

```

```

utils/
    class_map.txt
    resize_224x224.ipynb

```

```

webcam/
    bytetrack.yaml
    live_classifier.py

```

```

yolo/
    best_new.pt
    yolo_train.ipynb

```

README.md

### **Ý nghĩa:**

- `cnn/`: chứa notebook huấn luyện mô hình CNN phân loại cá.
- `docs/`: thư mục tài liệu dự án.
- `eda/`: chứa các notebook phân tích dữ liệu, đọc ảnh gốc và tạo nhãn dữ liệu cho cá.
- `utils/`: các tiện ích hỗ trợ như ánh xạ nhãn lớp và resize ảnh về kích thước 224x224.
- `webcam/`: xử lý real-time với webcam, track đối tượng bằng ByteTrack và phân loại trực tiếp.
- `yolo/`: huấn luyện YOLO và lưu trữ số mô hình nhận diện cá.
- `README.md`: mô tả tổng quan về toàn bộ dự án.

### 3.2.6 Pipeline xử lý trong thời gian thực

Quy trình xử lý từng khung hình được mô tả như sau:

- 1) Nhận đầu vào từ camera hoặc video.
- 2) YOLOv8 phát hiện tất cả vị trí cá trong khung hình.
- 3) Crop từng vùng chứa cá.
- 4) Đưa từng vùng vào CNN để phân loại loài.
- 5) Trả về:
  - Tên loài cá
  - Độ tin cậy (confidence)
  - Đường dẫn Wikipedia
- 6) Hiển thị kết quả ngay trên khung hình.

Cơ chế hiển thị thông tin Wikipedia được tích hợp bằng cách ánh xạ:

species → URL\_wikipedia

## Phần 4

### KẾT QUẢ VÀ PHÂN TÍCH

#### 4.1 Đánh giá mô hình Yolov8

##### 4.1.1 Kết quả tổng thể

Bảng 4.1 trình bày các chỉ số đánh giá chính của mô hình YOLO trên tập kiểm thử. Chỉ số mAP@0.5 và mAP@0.5:0.95 thể hiện chất lượng phát hiện đối tượng, Precision và Recall phản ánh độ chính xác và khả năng tìm đúng các cá thể, với F1-score là chỉ số tổng hợp cân bằng giữa Precision và Recall.

*Bảng 4.1: Kết quả đánh giá mô hình YOLO trên tập test và tập valid*

Chỉ số	Test	Valid	Ý nghĩa
mAP@0.5	0.447	0.454	Độ chính xác trung bình tại IoU = 0.5
mAP@0.5:0.95	0.308	0.315	Chỉ số mAP tổng quát (IoU từ 0.5 đến 0.95)
Precision	0.587	0.614	Tỷ lệ dự đoán đúng trên tổng số dự đoán
Recall	0.411	0.405	Tỷ lệ tìm đúng trên tổng số đối tượng thật
F1-score	0.495	0.495	Trung hòa giữa Precision và Recall

Dựa vào bảng 4.1, mô hình YOLOv8 cho thấy hiệu năng ổn định giữa tập test và tập valid. Các chỉ số giữa hai tập gần bằng nhau, cho thấy mô hình không bị overfit, mặc dù dataset gồm 80.000 ảnh với số lượng và phân bố các lớp cá không đồng đều.

##### Đánh giá mAP

Chỉ số **mAP@0.5** đạt 0.447 trên tập test và 0.454 trên tập valid, chứng tỏ mô hình có khả năng phát hiện tổng quan các đối tượng ở mức IoU = 0.5. Khi yêu cầu độ chính xác chặt hơn với **mAP@0.5:0.95**, giá trị giảm còn 0.308 trên train và 0.315 trên valid, phản ánh rằng việc định vị chính xác các bounding box trên tất cả các lớp cá, đặc biệt là những lớp ít ảnh, vẫn là thách thức. Đây là kết quả hợp lý trong bối cảnh dataset mất cân bằng và nhiều lớp, với sự đa dạng về hình dạng, kích thước và môi trường nước.

### Đánh giá Precision và Recall

Chỉ số **Precision** đạt 0.587 trên test và 0.614 trên valid, cho thấy mô hình dự đoán chính xác khi xác định một đối tượng thuộc một lớp nào đó. Chỉ số **Recall** lần lượt là 0.411 và 0.405, phản ánh rằng một số đối tượng trong ảnh vẫn bị bỏ sót. Điều này là hợp lý trong dataset nhiều lớp và mất cân bằng, khi các lớp ít ảnh khó học hơn và một số cá thể bị che khuất hoặc có kích thước nhỏ.

### Đánh giá F1-score

Chỉ số **F1-score** giữ nguyên ở mức 0.495 cho cả test và valid, thể hiện sự cân bằng giữa Precision và Recall. Điều này chứng tỏ mô hình hoạt động ổn định và nhất quán giữa hai tập, đồng thời phù hợp với mức độ khó của bài toán nhận diện nhiều lớp cá trong môi trường nước phức tạp.

### Kết luận tổng thể

Nhìn chung, mô hình YOLOv8 đã thể hiện hiệu năng ổn định, với khả năng dự đoán chính xác các đối tượng phổ biến và duy trì tính nhất quán giữa test và valid. Mức độ Recall và mAP tổng quát phản ánh đặc điểm khó của dataset, bao gồm phân bố lớp mất cân bằng, nhiều cá thể bị che khuất hoặc có kích thước nhỏ, chứ không phải do lỗi huấn luyện hay thiết kế mô hình.

#### 4.1.2 Đánh giá theo các nhóm độ phổ biến

*Bảng 4.2: Hiệu suất mô hình YOLOv8 theo độ phổ biến của các họ cá*

Nhóm	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Common	0.721	0.518	0.580	0.269
Medium	0.581	0.461	0.480	0.219
Rare	0.558	0.369	0.399	0.224

#### Nhận xét:

Bảng 4.2 tóm tắt hiệu suất mô hình YOLOv8 theo độ phổ biến của các họ cá. Nhóm *Common* gồm các họ phổ biến, *Medium* gồm các họ trung bình, và *Rare* là các họ hiếm.

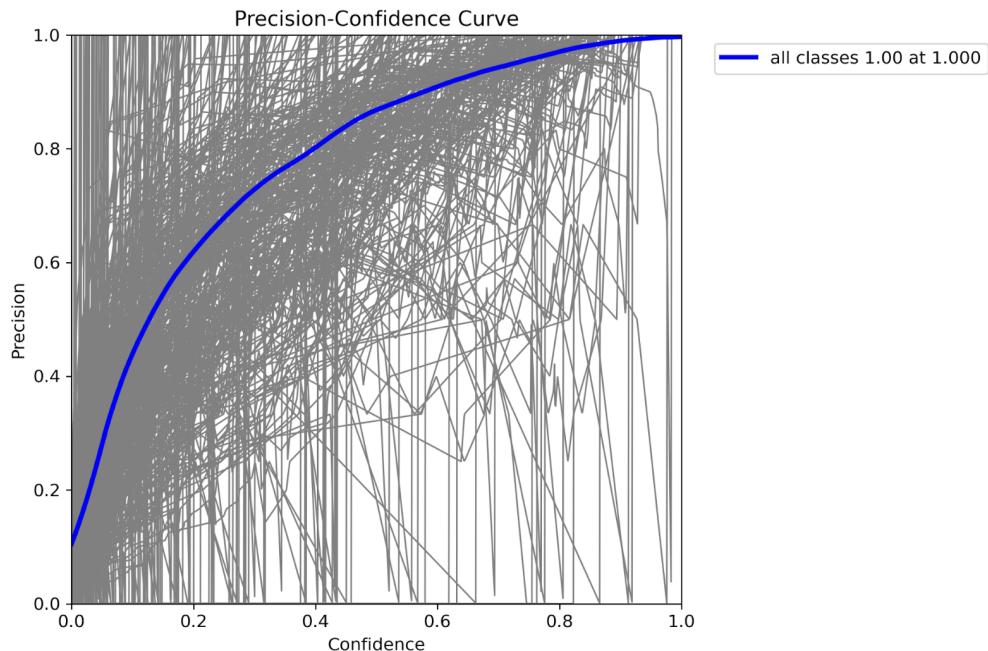
Có thể thấy, mô hình đạt **Precision và mAP cao nhất ở nhóm Common** (0.721 và 0.580), chứng tỏ các lớp phổ biến được mô hình học tốt, dự đoán chính xác và định vị bounding box hiệu quả. Ngược lại, nhóm Rare có chỉ số thấp nhất (Precision 0.558, mAP50 0.399), phản ánh rằng các lớp hiếm khó học hơn do số lượng ảnh ít, đa dạng

về hình dạng, kích thước và điều kiện môi trường (như ánh sáng, cá bị che khuất hoặc chuyển động).

**Recall** cũng giảm theo độ phổ biến: từ 0.518 (Common) xuống 0.369 (Rare), cho thấy mô hình bỏ sót nhiều đối tượng trong các lớp ít dữ liệu. Điều này là bình thường trong các dataset mất cân bằng mạnh, khi các lớp phổ biến chiếm ưu thế trong quá trình huấn luyện, còn các lớp hiếm chưa được học đủ để nhận diện tất cả cá thể.

Nhìn chung, việc phân tích theo nhóm giúp đánh giá chi tiết khả năng mô hình với các lớp hiếm và phổ biến trong dataset không cân bằng. Kết quả cho thấy mô hình YOLOv8 có hiệu năng tốt với lớp phổ biến và vẫn duy trì dự đoán có ý nghĩa ở các lớp ít xuất hiện, phù hợp với đặc thù dữ liệu đa dạng.

#### 4.1.3 Độ chính xác theo ngưỡng tin cậy



**Hình 4.1:** Đường cong Precision-Confidence (Độ chính xác theo Ngưỡng tin cậy) trên tập kiểm thử

#### Nhận xét về Đường cong Precision-Confidence

Hình 4.1 minh họa mối quan hệ giữa độ chính xác (Precision) và ngưỡng tin cậy (Confidence) của các dự đoán. Đường cong màu xanh lam thể hiện độ chính xác trung bình tổng hợp cho tất cả các lớp, trong khi các đường màu xám đại diện cho các đường cong Precision theo từng lớp hoặc từng ngưỡng IoU cụ thể, cho thấy sự biến động cục bộ.

Đường cong trung bình cho thấy mối quan hệ tỷ lệ thuận rõ rệt: với ngưỡng tin cậy thấp, Precision cũng thấp do nhiều dự đoán sai; khi ngưỡng tin cậy tăng lên (khoảng 0.8 - 0.9), Precision đạt gần tuyệt đối, chứng tỏ mô hình đánh giá các dự đoán tin cậy rất chính xác.

Hình dạng tổng thể cho thấy mô hình có khả năng phân biệt tốt giữa dự đoán đúng và sai, mặc dù sự dao động của các đường màu xám phản ánh độ chính xác cục bộ có thể không ổn định với một số đối tượng hoặc tình huống cụ thể.

**Kết luận:** Mô hình đạt hiệu suất cao khi áp dụng ngưỡng tin cậy lớn, khuyến nghị chọn Confidence 0.7 để đảm bảo phần lớn các phát hiện là chính xác, giảm thiểu đáng kể các False Positives.

## 4.2 Đánh giá mô hình CNN

### 4.2.1 Thiết lập các trọng số cho mô hình

Trong quá trình xây dựng mô hình CNN cho bài toán phân loại cá, một đặc điểm đáng chú ý của dữ liệu là sự mất cân bằng giữa các lớp: một số loài cá phổ biến có số lượng ảnh rất lớn, trong khi các loài hiếm hoặc khó thu thập chỉ có rất ít mẫu. Để hạn chế hiện tượng mô hình “thiên lệch” về các lớp đa số, trong quá trình huấn luyện, nhóm đã sử dụng cơ chế trọng số lớp (class\_weight) xây dựng trực tiếp từ phân bố dữ liệu. Cụ thể, hàm `compute_class_weights_from_generator` duyệt qua các thư mục tương ứng với từng loài cá trong `train_generator` để đếm số lượng ảnh của mỗi lớp, sau đó tính toán trọng số dựa trên tỉ lệ giữa số ảnh lớn nhất và số ảnh của từng lớp, có áp dụng phép biến đổi log để làm “dịu” sự chênh lệch. Kết quả là một từ điển `class_weight_dict` ảnh xạ từ chỉ số lớp sang trọng số tương ứng; cấu trúc này được truyền vào `model.fit` trong cả hai giai đoạn huấn luyện, giúp tăng mức phạt đối với lỗi trên các lớp ít mẫu mà không cần thay đổi kiến trúc mô hình.

Về kiến trúc, mô hình CNN được xây dựng theo hướng học chuyển giao với backbone MobileNetV2 đã tiền huấn luyện trên ImageNet. Mô hình được khởi tạo với `include_top=False` và `weights="imagenet"` để sử dụng phần trích xuất đặc trưng, bỏ đi tầng phân loại gốc. Kích thước ảnh đầu vào được chuẩn hóa về  $224 \times 224$  pixel, 3 kênh màu (RGB), đồng bộ với kiến trúc **MobileNetV2**. Trong giai đoạn huấn luyện đầu tiên, toàn bộ backbone được cố định và chỉ phần đầu phân loại (classifier head) được huấn luyện. Đầu ra feature map của backbone được đưa vào một lớp pooling toàn

cục, sau đó đi qua các lớp Dense, BatchNormalization và Dropout, cuối cùng là lớp softmax cho phân loại đa lớp. Cấu trúc phần head và lý do lựa chọn các tham số chính được tóm tắt như sau:

Chi tiết các lớp trong phần head của mô hình được tóm tắt trong Bảng 4.3.

**Bảng 4.3:** Cấu trúc và vai trò các lớp trong phần head của mô hình CNN

Thành phần	Mô tả và lý do lựa chọn
GlobalAveragePooling2D	Pooling trung bình toàn cục trên không gian (spatial), chuyển tensor đặc trưng 2D thành vector 1 chiều. Giảm mạnh số tham số so với Flatten, tăng khả năng khai quát hóa và giảm nhẹ cảm với dịch chuyển, nhiễu cục bộ trong ảnh.
Dense(512, relu) + BatchNormalization + Dropout(0.5)	Tầng Dense 512 nút cung cấp năng lực biểu diễn phi tuyến đủ lớn trên đặc trưng do MobileNetV2 trích xuất. ReLU đơn giản, hiệu quả và giảm hiện tượng mất gradient. BatchNormalization ổn định phân phối kích hoạt, cho phép dùng learning rate tương đối cao. Dropout 0.5 đóng vai trò regularization mạnh, hạn chế overfitting khi dữ liệu không quá lớn.
Dense(256, relu) + BatchNormalization + Dropout(0.3)	Tầng Dense 256 nút tiếp tục tinh chế đặc trưng ở mức trừu tượng cao hơn; giảm số nút từ 512 xuống 256 để cân bằng giữa năng lực mô hình và nguy cơ overfitting. BatchNormalization duy trì ổn định quá trình huấn luyện, Dropout 0.3 cung cấp regularization nhẹ hơn, phù hợp với giai đoạn sâu trong head.
Dense(num_classes, softmax)	Lớp đầu ra với số nút bằng số loài cá, dùng softmax để ánh xạ vector đặc trưng cuối thành phân phối xác suất trên không gian nhãn. Tương thích trực tiếp với hàm mất mát categorical_crossentropy và cho phép diễn giải đầu ra như xác suất dự đoán cho từng loài.

Chiến lược huấn luyện được tổ chức thành hai giai đoạn như trình bày trong Bảng 4.4.

**Bảng 4.4:** Chiến lược huấn luyện hai giai đoạn cho mô hình CNN

Nội dung	Phase 1: Huấn luyện head	Phase 2: Fine-tune backbone
Backbone	MobileNetV2 bị đóng băng hoàn toàn, chỉ dùng để trích xuất đặc trưng.	Mở trainable một phần (các lớp cuối), các lớp thấp vẫn giữ cố định.
Phần head	Khởi tạo mới, được huấn luyện đầy đủ với dữ liệu cá.	Tiếp tục được cập nhật cùng với phần backbone được mở.
Bộ tối ưu, LR	Adam, learning rate tương đối cao $10^{-3}$ để học nhanh các lớp mới.	Adam, learning rate nhỏ hơn ( $\sim 10^{-4}$ ) để tinh chỉnh nhẹ trọng số.
Class weight	Sử dụng class_weight_dict trong model.fit để xử lý mất cân bằng lớp.	Tiếp tục dùng class_weight_dict nhằm ưu tiên các lớp ít mẫu khi fine-tune.
Callbacks	EarlyStopping, ReduceLROn-Plateau để tránh overfitting và điều chỉnh LR.	Giữ nguyên các callback, patience nhỏ hơn vì mô hình đã gần vùng tối ưu.
Mục tiêu	Học nhanh tầng phân loại mới trên đặc trưng tiền huấn luyện.	Điều chỉnh một phần backbone cho phù hợp hơn với dữ liệu cá.

#### 4.2.2 Kết quả thực nghiệm với mô hình CNN

Sau khi hoàn tất huấn luyện hai giai đoạn, mô hình CNN dựa trên backbone MobileNetV2 được đánh giá trên tập *validation* gồm 18 510 mẫu, với kích thước batch là 64. Kết quả đánh giá tổng thể cho thấy mô hình đạt độ chính xác tương đối cao, đồng thời các thước đo Precision, Recall và F1-Score cũng ở mức cân bằng, phản ánh khả năng phân loại ổn định trên tập dữ liệu chưa từng thấy trong quá trình huấn luyện.

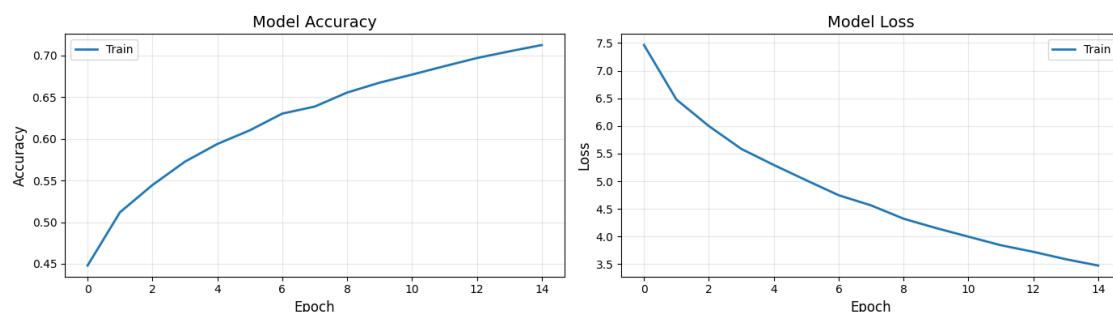
**Bảng 4.5:** Kết quả đánh giá tổng thể của mô hình CNN trên tập validation

Chỉ số	Giá trị
Accuracy	0.8269
Precision	0.8360
Recall	0.8269
F1-Score	0.8261
Số mẫu (validation)	18 510
Batch size	64

Bảng 4.5 cho thấy độ chính xác tổng thể đạt khoảng 82.7%, trong khi Precision, Recall và F1-Score đều xấp xỉ nhau (chênh lệch nhỏ hơn 1%). Điều này cho thấy mô hình không chỉ dự đoán đúng với tỉ lệ cao, mà còn duy trì được sự cân bằng giữa hai loại lỗi *false positive* và *false negative*, phù hợp với mục tiêu giảm thiểu nhầm lẫn giữa các loài cá khác nhau trong bối cảnh dữ liệu không đồng đều.

#### Điễn biến quá trình huấn luyện

Điễn biến độ chính xác và hàm mất mát trên tập huấn luyện theo từng epoch được minh họa trong Hình 4.3. Có thể quan sát thấy độ chính xác tăng đều từ khoảng 0.45 lên hơn 0.70 sau 15 epoch, trong khi giá trị loss giảm đơn điệu từ khoảng 7.5 xuống còn xấp xỉ 3.5.

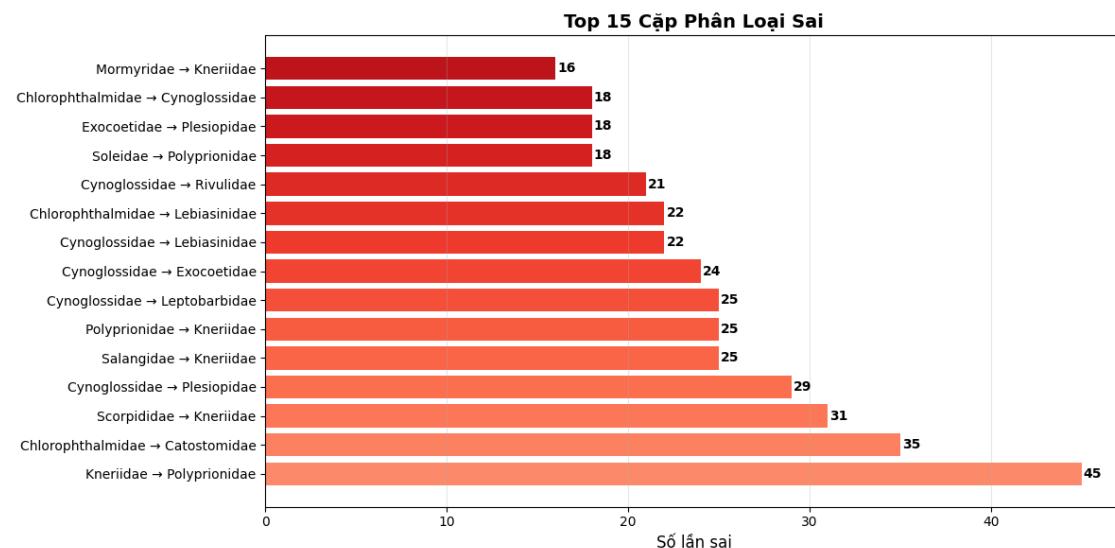


**Hình 4.2:** Sự biến thiên độ chính xác và hàm mất mát trên tập train theo từng epoch.

Xu hướng tăng dần của độ chính xác kèm theo sự giảm ổn định của loss cho thấy quá trình tối ưu hội tụ tốt, không xuất hiện dao động mạnh hay dấu hiệu học quá đà trong giai đoạn đầu. Khi so sánh với các chỉ số trên tập validation (Bảng 4.5), có thể nhận thấy mô hình vẫn giữ được khả năng khai quát hóa tốt, cho thấy các kỹ thuật tăng cường dữ liệu và đánh trọng số lớp phát huy hiệu quả trong bối cảnh dữ liệu mất cân bằng.

#### Phân tích lỗi và các cặp loài dễ nhầm lẫn

Để hiểu rõ hơn hành vi của mô hình, nghiên cứu tiến hành thống kê các cặp (*loài thật → loài dự đoán*) bị nhầm lẫn nhiều nhất trên tập validation. Hình 4.3 thể hiện *Top 15* cặp phân loại sai cùng số lần xuất hiện tương ứng.



**Hình 4.3:** Top 15 cặp loài cá bị phân loại sai nhiều nhất trên tập validation.

## Phần 5

### DEMO NHẬN DẠNG CÁ TỪ WEBCAM VÀ TRA CỨU WIKIPEDIA

Chương này trình bày quy trình chạy (demo) của hệ thống nhận dạng cá thời gian thực, sử dụng webcam làm nguồn dữ liệu đầu vào. Hệ thống được xây dựng bằng Python với các thành phần chính: YOLOv8 cho nhận dạng đối tượng, ByteTrack cho theo dõi mục tiêu, mô-đun ổn định nhãn và bounding box, cơ chế tự đóng băng (Auto-Freeze), và tích hợp tra cứu thông tin loài cá từ Wikipedia.

Hệ thống hỗ trợ tương tác người dùng thông qua các phím sau:

- **q**: thoát chương trình
- **SPACE**: bật/tắt chế độ đóng băng
- **L**: mở trang Wikipedia của loài cá nếu có

Tóm tắt chức năng chính của hệ thống demo:

- Kết hợp các module nhận dạng, theo dõi, ổn định nhãn, Auto-Freeze và tra cứu Wikipedia.
- Cơ chế Auto-Freeze giúp hiển thị mượt mà và tiện lợi cho người dùng.
- Tra cứu Wikipedia cung cấp thông tin loài cá kịp thời mà không làm chậm hiển thị frame từ webcam.

#### 5.1 Khởi Tạo và Cấu Hình Hệ Thống

Hệ thống khởi tạo các thư viện cần thiết:

- **OpenCV** để xử lý ảnh từ webcam và hiển thị kết quả.
- **Ultralytics YOLO** để nhận dạng đối tượng trong từng frame.
- **ByteTrack** để theo dõi các đối tượng qua các frame liên tiếp, duy trì định danh `track_id`.

- **Wikipedia API** để tra cứu thông tin loài cá khi nhãn ổn định được xác định.
- **NumPy** để xử lý các phép toán số học trên bounding box.
- **threading** để thực hiện tra cứu Wikipedia song song, tránh làm chậm hiển thị frame hiện tại.

Mô hình YOLOv8 đã được huấn luyện trước được tải từ đường dẫn cục bộ, cùng với tệp cấu hình ByteTrack. Các tham số cấu hình chính:

- YOLO\_CONF\_THRESHOLD = 0.35: ngưỡng nhận dạng tối thiểu.
- CONF\_FREEZE\_LIMIT = 0.60: độ tin cậy tối thiểu để kích hoạt cơ chế Auto-Freeze.
- STABILITY\_THRESHOLD = 5: số frame liên tiếp để xác định nhãn ổn định.
- AUTO\_FREEZE\_THRESHOLD = 8: số frame liên tiếp với độ tin cậy cao để tự động đóng băng.
- MAX\_MISSES = 10: số frame tối đa mà một đối tượng mất dấu trước khi bị loại khỏi theo dõi.

Các biến toàn cục được sử dụng để theo dõi trạng thái hệ thống bao gồm nhãn ổn định hiện tại, độ tin cậy cao nhất, bounding box trung bình, bộ nhớ cache Wikipedia, trạng thái đóng băng, và các luồng tra cứu đang chạy.

## 5.2 Vòng Lặp Xử Lý Thời Gian Thực

Sau khi mở webcam bằng lệnh `VideoCapture(0)`, hệ thống bắt đầu vòng lặp xử lý từng frame. Quy trình cơ bản gồm các bước:

### 5.2.1 Nhận dạng và Theo dõi YOLO + ByteTrack

Mỗi frame được gửi vào YOLOv8 để nhận dạng và ByteTrack để theo dõi các đối tượng:

- YOLOv8 trả về bounding box, nhãn, độ tin cậy, và `track_id`.
- ByteTrack dự đoán vị trí đối tượng trong frame tiếp theo dựa trên Kalman Filter và IoU, duy trì `track_id` ổn định.
- Nếu một `track_id` không xuất hiện trong `MAX_MISSES` frame liên tiếp, nó sẽ bị loại khỏi danh sách theo dõi.

Mỗi đối tượng được lưu trong cấu trúc tracked\_fishes, gồm:

- Lịch sử nhãn và bounding box qua các frame.
- Độ tin cậy cao nhất từng quan sát.
- Số frame liên tiếp mà đối tượng không xuất hiện.
- Số frame liên tiếp đạt độ tin cậy cao để kích hoạt Auto-Freeze.

### 5.2.2 Ổn định Nhãn và Bounding Box

Hệ thống duy trì lịch sử nhãn và bounding box gần nhất để giảm dao động. Nhãn ổn định được xác định bằng cách chọn nhãn xuất hiện nhiều nhất trong lịch sử:

`stable_label = most_common(history) nếu số lần xuất hiện  $\geq \frac{1}{2} \times \text{STABILITY\_THRESHOLD}$`

Bounding box được làm mượt bằng trung bình của các bounding box trong lịch sử gần nhất:

$$\text{box}_{stable} = \frac{1}{n} \sum_{i=1}^n \text{box}_i$$

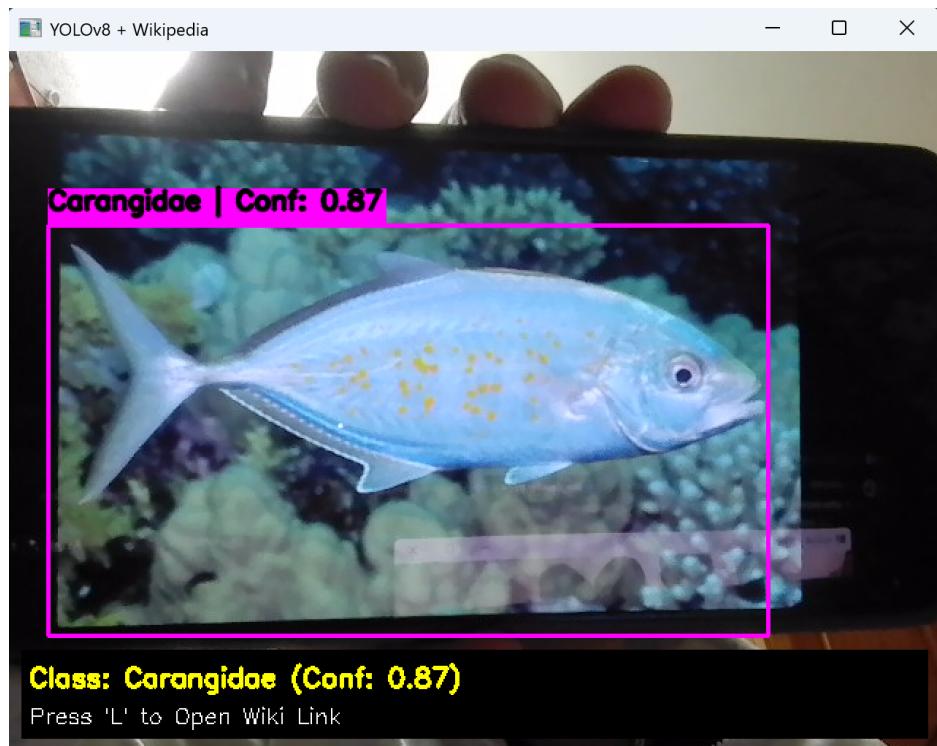
hoặc có thể sử dụng trung bình trọng số theo độ tin cậy để giảm ảnh hưởng của các dự đoán sai tạm thời.

### 5.2.3 Cơ Chế Auto-Freeze

Nếu một đối tượng xuất hiện liên tục với độ tin cậy cao hơn CONF\_FREEZE\_LIMIT trong AUTO\_FREEZE\_THRESHOLD frame liên tiếp, hệ thống sẽ tự động đóng băng frame:

$$\text{high\_conf\_frames} \geq \text{AUTO\_FREEZE\_THRESHOLD} \Rightarrow \text{freeze}$$

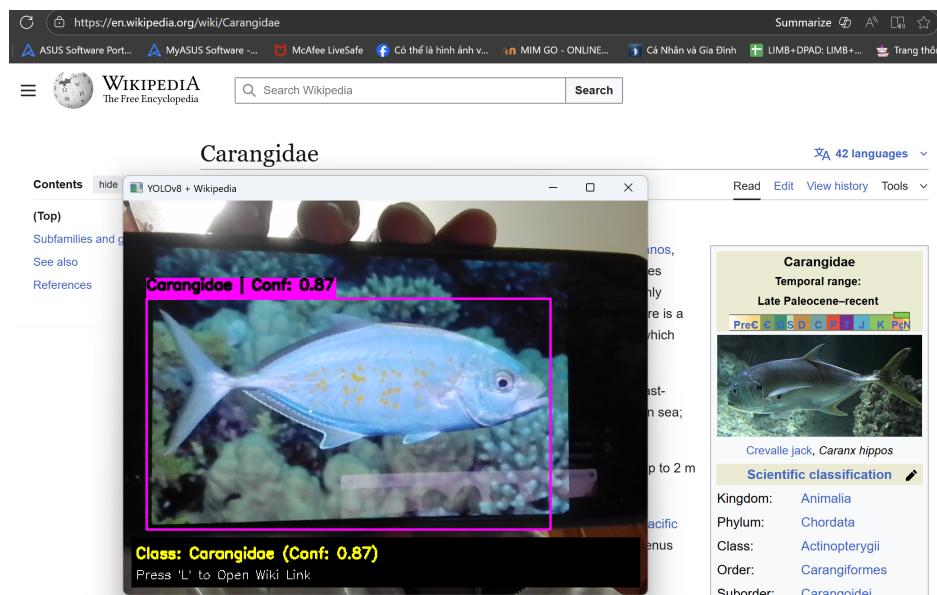
Cơ chế này giúp người dùng có thời gian đọc thông tin nhãn và tra cứu Wikipedia mà không bị thay đổi frame. Người dùng cũng có thể kích hoạt đóng băng thủ công bằng phím SPACE.



**Hình 5.1:** Chế độ Auto-Freeze giữ nguyên frame

#### 5.2.4 Tra cứu Wikipedia và Hiển Thị Thông Tin

Khi nhãn ổn định của đối tượng được xác định hoặc khi người dùng kích hoạt chế độ đóng băng, hệ thống sẽ tra cứu thông tin loài cá từ Wikipedia. Việc tra cứu được thực hiện trong một luồng riêng, tách biệt với quá trình xử lý frame từ webcam. Cách làm này đảm bảo rằng việc tra cứu không làm chậm hoặc gián đoạn hiển thị frame hiện tại.



**Hình 5.2:** Tra cứu thông tin loài cá từ Wikipedia.

Một luồng riêng được khởi tạo bằng lệnh `thread(target=fetch_wiki, args=(label_search,), daemon=True).start()`, cho phép tra cứu Wikipedia song song với quá trình nhận dạng frame từ webcam.

Thông tin tra cứu được lưu trong bộ nhớ cache, ngăn việc gửi nhiều yêu cầu trùng lặp và cải thiện hiệu năng hệ thống. Hệ thống xử lý các ngoại lệ phổ biến từ API Wikipedia như `DisambiguationError`, `PageError` hoặc các lỗi tổng quát, đảm bảo không làm gián đoạn quá trình nhận dạng hoặc hiển thị frame hiện tại. Khi tra cứu thành công, người dùng có thể mở trang Wikipedia tương ứng bằng phím L.

### 5.2.5 Hiển thị thông tin lên màn hình

Kết quả nhận dạng, bao gồm bounding box, nhãn ổn định và độ tin cậy, được vẽ trực tiếp lên frame hiện tại. Màu sắc được sử dụng để biểu diễn trạng thái:

- Màu xanh lá: dự đoán bình thường
- Màu hồng: chế độ đóng băng (tự động hoặc thủ công)

Ngoài bounding box, hệ thống hiển thị thông tin bổ sung ở cuối khung hình, bao gồm nhãn loài cá, độ tin cậy dự đoán, trạng thái tra cứu Wikipedia (đang tải, lỗi hoặc có sẵn), và hướng dẫn mở liên kết nếu thông tin hợp lệ. Cách hiển thị này giúp người dùng nắm được thông tin loài cá kịp thời mà không làm gián đoạn quá trình quan sát frame từ webcam.

Việc kết hợp tra cứu đa luồng, bộ nhớ cache và hiển thị trực quan giúp hệ thống trở nên linh hoạt, mượt mà và dễ sử dụng trong các tình huống quan sát thực tế.

## Phần 6

### KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

#### 6.1 Kết luận

Trong khuôn khổ đề tài, nhóm đã nghiên cứu, xây dựng và triển khai một hệ thống nhận diện cá theo thời gian thực dựa trên các kỹ thuật học sâu hiện đại, kết hợp giữa mô hình phân loại ảnh CNN và mô hình phát hiện đối tượng YOLOv8. Hệ thống cho phép nhận diện chính xác loài cá trong hình ảnh, video và luồng camera trực tiếp, đồng thời tự động liên kết đến trang Wikipedia tương ứng nhằm cung cấp thông tin sinh học, phân loại và đặc điểm nhận diện của từng loài.

Kết quả thực nghiệm cho thấy:

- YOLOv8 đạt hiệu suất nhận diện (mAP) cao trên bộ dữ liệu thử nghiệm, đồng thời đảm bảo tốc độ xử lý thời gian thực (từ 25–60 FPS tùy phần cứng).
- Các mô hình CNN như ResNet, EfficientNet và MobileNetV3 cho độ chính xác tốt trên bài toán phân loại hình ảnh tĩnh, hỗ trợ nhận diện chi tiết khi cần phân biệt những loài có hình thái gần giống nhau.
- Pipeline tích hợp Wikipedia API hoạt động ổn định, cho phép hiển thị nhanh thông tin sinh học liên quan đến loài cá vừa được phát hiện.
- Hệ thống đáp ứng mục tiêu ban đầu: hoạt động ổn định trong thời gian thực, độ chính xác cao, giao diện trực quan, có khả năng mở rộng trong tương lai.

Bên cạnh những kết quả tích cực, hệ thống vẫn tồn tại một số hạn chế:

- Độ chính xác giảm trong điều kiện ánh sáng yếu, độ tương phản thấp hoặc nhiều lớp phản chiếu dưới nước.
- Một số loài cá có hình dạng và hoa văn gần giống nhau khiến mô hình khó phân biệt, đặc biệt khi kích thước cá nhỏ hoặc bị che khuất.

- Bộ dữ liệu chưa bao phủ đầy đủ tất cả các loài cá bản địa hoặc các loài trong môi trường tự nhiên đặc thù.

Tuy nhiên, các hạn chế này hoàn toàn có thể khắc phục bằng việc mở rộng dataset, cải thiện kỹ thuật augmentation, sử dụng các mô hình thị giác hiện đại hơn hoặc triển khai tối ưu phần cứng.

## 6.2 Hướng phát triển

Trong tương lai, hệ thống có thể được mở rộng theo các hướng sau:

- **Mở rộng tập dữ liệu:** Thu thập thêm dữ liệu từ nhiều môi trường khác nhau (nước trong, đục, biển sâu, hồ tự nhiên) nhằm tăng độ chính robust.
- **Tích hợp các mô hình thị giác mới:** Áp dụng Vision Transformer (ViT), YOLOv9, hay các mô hình multi-modal để cải thiện kết quả nhận diện.
- **Tối ưu triển khai trên Edge/IoT:** Chuyển đổi sang TensorRT, OpenVINO hoặc ONNX Runtime nhằm chạy mô hình trên Jetson Nano, Raspberry Pi hoặc thiết bị nhúng.
- **Nhận diện hành vi và theo dõi chuyển động (tracking):** Kết hợp DeepSORT hoặc ByteTrack để theo dõi cá theo thời gian.
- **Ứng dụng vào nuôi trồng thủy sản thông minh:** Đếm số lượng cá, ước lượng kích thước, phát hiện cá bệnh hoặc hành vi bất thường.
- **Xây dựng cơ sở dữ liệu loài cá chuyên sâu:** Tự động lưu trữ thông tin đã thu thập, hỗ trợ công tác nghiên cứu thuỷ sinh, giáo dục và bảo tồn.

Nhìn chung, đề tài không chỉ mang ý nghĩa thực tiễn trong ngành thuỷ sản và sinh học mà còn mở ra nhiều tiềm năng ứng dụng của thị giác máy tính trong các lĩnh vực nghiên cứu, giáo dục, bảo tồn và công nghiệp.

## TÀI LIỆU THAM KHẢO

- [1] Khan, Faizan Farooq and Li, Xiang and Temple, Andrew J. and Elhoseiny, Mohamed, *FishNet: A Large-scale Dataset and Benchmark for Fish Recognition, Detection, and Functional Trait Prediction*, in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20496–20506, Oct. 2023.  
(available at <https://fishnet-2023.github.io/>).
- [2] He, K., Zhang, X., Ren, S., & Sun, J. (2016), *Deep Residual Learning for Image Recognition*, CVPR (available at <https://arxiv.org/abs/1512.03385>).
- [3] Tan, M. & Le, Q. V. (2019), *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*, ICML (at <https://arxiv.org/abs/1905.11946>).
- [4] Howard, A. et al. (2019), *MobileNetV3: Searching for Mobile Architecture*, ICCV  
(available at <https://arxiv.org/abs/1905.02244>).
- [5] Bradski, G. (2000), *The OpenCV Library*, Dr. Dobb's Journal of Software Tools  
(available at <https://opencv.org>).
- [6] Paszke, A. et al. (2019), *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, NeurIPS (available at <https://pytorch.org>).
- [7] Wikipedia (n.d.), *MediaWiki Action API Guide*, Wikimedia Foundation (available at [https://www.mediawiki.org/wiki/API:Main\\_page](https://www.mediawiki.org/wiki/API:Main_page)).
- [8] Shorten, C. & Khoshgoftaar, T. (2019), *A Survey on Image Data Augmentation for Deep Learning*, Journal of Big Data (available at <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0>).
- [9] Goodfellow, I., Bengio, Y., & Courville, A. (2016), *Deep Learning*, MIT Press  
(available at <https://www.deeplearningbook.org>).
- [10] Jocher, G. (2023), *YOLOv8: Next-Generation Real-Time Object Detection*, Ultralytics (available at <https://docs.ultralytics.com>).