ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Ultralytics YOLO11

Báo cáo tìm hiểu bài báo
CSC14005 – Nhập môn học máy

Sinh viên thực hiện:

Đặng Anh Kiệt

Phạm Minh Triết

Trần Quang Phúc

Kiều Duy Hiếu

Giảng viên hướng dẫn:

Bùi Tiến Lên

Lê Nhựt Nam

Võ Nhật Tân

Thành phố Hồ Chí Minh, Ngày 6 tháng 12 năm 2025

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

DANH SÁCH CÁC KÍ TỰ VIẾT TẮT, THUẬT NGỮ

Thuật ngữ	**Mô tả**
AI	Trí tuệ nhân tạo (Artificial Intelligence)
CV	Thị giác máy tính (Computer Vision)
CNN	Mạng nơ-ron tích chập (Convolutional Neural Network)
YOLO	Bạn chỉ nhìn một lần (You Only Look Once)
SOTA	Tiên tiến nhất (State-of-the-Art)
FPS	Số khung hình trên giây (Frames Per Second)
Backbone	Mạng xương sống (Trích xuất đặc trưng)
Neck	Cổ mạng (Hợp nhất đặc trưng)
Head	Đầu dò (Dự đoán kết quả)
SPPF	Spatial Pyramid Pooling - Fast
C3K2	Khối CSP với kích thước kernel tùy chỉnh (Module mới trong YOLOv11)
C2PSA	Cross-Level Pyramid Slice Attention (Cơ chế chú ý trong YOLOv11)
CSP	Cross Stage Partial (Kiến trúc mạng)
PANet	Path Aggregation Network
IoU	Giao trên hợp (Intersection over Union)
mAP	Độ chính xác trung bình (mean Average Precision)
NMS	Triệt tiêu phi cực đại (Non-Maximum Suppression)
SiLU	Sigmoid Linear Unit (Hàm kích hoạt)
DFL	Distribution Focal Loss
CIoU	Complete Intersection over Union
BCE	Binary Cross Entropy

DANH SÁCH CÁC KÍ TỰ

Kí hiệu	Mô tả
$\mathbb{R}$	Tập số thực
$\mathcal{I}$	Hình ảnh đầu vào
H, W, C	Chiều cao, Chiều rộng và Số kênh của ảnh/tensor
$\mathbf{X}$	Tensor đặc trưng đầu vào
$\mathbf{Y}$	Tensor đặc trưng đầu ra
$\mathbf{K}$	Bộ lọc tích chập (Kernel)
Q, K, V	Các ma trận Truy vấn (Query), Khóa (Key) và Giá trị (Value) trong cơ chế chú ý
$\mathcal{F}$	Hàm ánh xạ của mạng nơ-ron
$\mathcal{O}$	Tensor dự đoán đầu ra
$\mathcal{L}_{total}$	Hàm mất mát tổng thể
$\mathcal{L}_{box}$	Hàm mất mát hồi quy hộp giới hạn
$\mathcal{L}_{cls}$	Hàm mất mát phân loại
$\mathcal{L}_{dfl}$	Hàm mất mát phân phối tiêu điểm
B_p	Hộp giới hạn dự đoán (Predicted Bounding Box)
B_{gt}	Hộp giới hạn nhãn (Ground Truth Bounding Box)
C	Điểm tin cậy (Confidence score) hoặc Nhãn lớp
$\sigma(.)$	Hàm kích hoạt Sigmoid
ρ	Khoảng cách Euclidean
θ	Tập hợp tham số trọng số của mô hình

CHƯƠNG 1

Giới thiệu

1.1. Bối cảnh và động lực vấn đề

Trong lĩnh vực thị giác máy tính (Computer Vision), bài toán phát hiện đối tượng trong thời gian thực đóng vai trò cốt lõi. Dòng mô hình YOLO (You Only Look Once), được giới thiệu lần đầu bởi Redmon et al. [8], đã tạo ra một cuộc cách mạng nhờ khả năng cân bằng vượt trội giữa tốc độ và độ chính xác.

Trải qua nhiều phiên bản cải tiến, từ các thế hệ ổn định như YOLOv8 [3] đến các kiến trúc tối ưu gradient như YOLOv9 [10], mỗi phiên bản đều mang lại những bước tiến đáng kể. YOLOv11, được Ultralytics phát hành vào cuối năm 2024 [4], tiếp tục kế thừa những thành công đó và giới thiệu các cải tiến kiến trúc đột phá nhằm tối ưu hóa hiệu suất trên thiết bị biên.

Tuy nhiên, việc hiểu rõ cơ chế hoạt động bên trong của các cải tiến này, đặc biệt là sự kết hợp giữa mạng nơ-ron tích chập (CNN) truyền thống và cơ chế chú ý (Attention Mechanism), vẫn là một thách thức. Trong khi các mô hình Transformer [9] đang dần phổ biến, việc tích hợp chúng vào các mô hình thời gian thực như YOLO đòi hỏi sự cân nhắc kỹ lưỡng về chi phí tính toán. Do đó, nhu cầu phân tích sâu về kiến trúc YOLOv11 để đánh giá hiệu quả thực tế là vô cùng cấp thiết.

1.2. Lựa chọn Chủ đề và Tài liệu Nghiên cứu

Đề tài này tập trung vào lĩnh vực **Thị giác Máy tính (Computer Vision)**, cụ thể là bài toán **Phát hiện Đối tượng Thời gian Thực (Real-time Object Detection)** trong Học Máy. Đây là một trong những hướng nghiên cứu quan trọng nhất của trí tuệ nhân tạo hiện đại, có ứng dụng rộng rãi trong xe tự hành, giám sát an ninh, y tế, và robot tự động.

1.2.1. Tài liệu Nghiên cứu Chính

Nghiên cứu này phân tích kiến trúc **Ultralytics YOLOv11** [4], phiên bản mới nhất trong dòng mô hình YOLO được phát hành bởi Ultralytics vào năm 2024. YOLOv11 đại diện cho sự tiến hóa mới nhất của kiến trúc one-stage detector, kế thừa từ các phiên bản tiền nhiệm đã được công bố tại các hội nghị hàng đầu:

- **YOLO (v1-v3):** Được công bố tại IEEE CVPR [8], một hội nghị xếp hạng A* theo CORE Rankings.
- **YOLOv9:** Kiến trúc GELAN và PGI được công bố trên arXiv [10].
- **Các kỹ thuật nền tảng:** CSPNet (CVPR 2020) [11], PANet (CVPR 2018) [7], Attention Mechanism (NeurIPS 2017) [9].

1.3. Mục tiêu nghiên cứu và ý nghĩa

YOLOv11 được chọn làm đối tượng nghiên cứu vì đây là phiên bản mới nhất (2024) giới thiệu các cải tiến kiến trúc đột phá như khối C3K2 và cơ chế chú ý C2PSA, thể hiện xu hướng kết hợp CNN và Transformer trong thiết kế mạng nơ-ron hiện đại. Hơn nữa, YOLOv11 được tối ưu hóa cho triển khai thực tế trên thiết bị biên, đồng thời framework Ultralytics cung cấp implementation mã nguồn mở hoàn chỉnh, tạo điều kiện thuận lợi cho việc thực nghiệm và tái tạo kết quả.

Mục đích của nghiên cứu này là phân tích toàn diện kiến trúc YOLOv11, làm sáng tỏ các cơ chế tối ưu hóa giúp mô hình đạt được hiệu suất cao với tài nguyên hạn chế. Cụ thể, nghiên cứu tập trung vào các mục tiêu sau:

- **Phân tích kiến trúc:** Mô tả chi tiết cấu trúc ba thành phần Backbone, Neck và Head, làm rõ vai trò của từng khối trong luồng xử lý đặc trưng.
- **Nghiên cứu cơ chế:** Giải thích toán học về hoạt động của khối tinh chỉnh đặc trưng C3K2 và khối chú ý không gian C2PSA.
- **Đánh giá thực nghiệm:** So sánh hiệu suất của YOLOv11 với các thế hệ tiền nhiệm (YOLOv8, YOLOv9) trên bộ dữ liệu Rock-Paper-Scissors.

- **Ứng dụng thực tế:** Triển khai ứng dụng phát hiện đối tượng thời gian thực để kiểm chứng khả năng áp dụng của mô hình.

Ý nghĩa của công trình nằm ở việc cung cấp một cái nhìn sâu sắc về xu hướng thiết kế mạng nơ-ron hiện đại: chuyển dịch từ việc tăng độ sâu mạng sang tối ưu hóa chất lượng đặc trưng thông qua cơ chế chú ý, đồng thời cung cấp cơ sở lý thuyết vững chắc cho việc triển khai mô hình trên các thiết bị phần cứng thế hệ mới.

1.4. Tổng quan đóng góp

Báo cáo này mang lại những đóng góp chính sau:

- Hệ thống hóa kiến trúc YOLOv11 và so sánh sự khác biệt kỹ thuật so với các phiên bản tiền nhiệm (YOLOv8, YOLOv9).
- Phân tích toán học chi tiết về cơ chế chú ý C2PSA và các hàm mất mát tiên tiến (CIoU, DFL).
- Đề xuất một kịch bản thực nghiệm chuẩn hóa để đánh giá hiệu suất mô hình trên bộ dữ liệu *Rock Paper Scissors*.
- Cung cấp cơ sở lý thuyết vững chắc cho việc triển khai YOLOv11 trên các thiết bị phần cứng thế hệ mới.

1.5. Cấu trúc báo cáo

Báo cáo được tổ chức thành 6 chương như sau:

- **Chương 2: Các công trình liên quan.** Tổng quan về sự tiến hóa của dòng mô hình YOLO, so sánh với các cách tiếp cận hiện có và làm rõ khoảng trống nghiên cứu mà YOLOv11 giải quyết.
- **Chương 3: Kiến thức nền tảng.** Trình bày các khái niệm toán học cốt lõi bao gồm Mạng Nơ-ron Tích chập (CNN), Cơ chế Chú ý (Attention Mechanism), các hàm mất mát (CIoU, DFL), và kỹ thuật Spatial Pyramid Pooling.

- **Chương 4: Phương pháp nghiên cứu.** Đi sâu vào mô hình hóa toán học kiến trúc YOLOv11, phân tích chi tiết các khối C3K2, C2PSA và các hàm mất mát được sử dụng trong quá trình huấn luyện.
- **Chương 5: Thực nghiệm và Đánh giá.** Trình bày bộ dữ liệu Rock-Paper-Scissors, môi trường thực nghiệm, kết quả so sánh giữa YOLOv8s, YOLOv9s và YOLOv11s, cùng với phân tích chi tiết các chỉ số đánh giá.
- **Chương 6: Kết luận và Định hướng phát triển.** Tổng kết các phát hiện chính, đánh giá điểm mạnh và hạn chế của YOLOv11, đề xuất hướng nghiên cứu và phát triển tiếp theo.

CHƯƠNG 2

Các công trình liên quan

2.1. Tổng quan nghiên cứu liên quan

Lĩnh vực phát hiện đối tượng (Object Detection) đã chứng kiến sự phát triển vượt bậc trong thập kỷ qua, với dòng mô hình YOLO (You Only Look Once) đóng vai trò tiên phong trong việc xử lý thời gian thực [8]. Từ phiên bản đầu tiên (YOLOv1) đến các phiên bản hiện đại hơn như YOLOv8 [3] và YOLOv10, xu hướng nghiên cứu luôn tập trung vào việc tìm kiếm sự cân bằng tối ưu giữa tốc độ (speed) và độ chính xác (accuracy).

Các nghiên cứu trước đây chủ yếu tập trung vào việc cải thiện kiến trúc mạng xương sống (Backbone) và cổ mạng (Neck) để trích xuất đặc trưng hiệu quả hơn [11]. Tuy nhiên, thách thức về việc duy trì hiệu suất cao trên các thiết bị biên (edge devices) với tài nguyên tính toán hạn chế vẫn là động lực chính thúc đẩy sự ra đời của các kiến trúc mới như YOLOv11 [4].

2.2. So sánh với các cách tiếp cận hiện có

Sự khác biệt cốt lõi giữa YOLOv11 và các phương pháp hiện có nằm ở thiết kế của khối trích xuất đặc trưng. Dựa trên phân tích kiến trúc, ta có thể so sánh ba thế hệ khối đặc trưng tiêu biểu:

- **YOLOv5 và Khối C3:** Sử dụng các tích chập 3×3 tiêu chuẩn. Khối C3 cung cấp sự cân bằng cơ bản giữa tốc độ và độ chính xác nhưng hạn chế về luồng gradient và khả năng xử lý các trường hợp phức tạp so với các thế hệ sau.
- **YOLOv8 và Khối C2F:** Tập trung tối đa vào việc tăng cường luồng gradient thông qua cấu trúc hợp nhất kép (2F) [3]. Mặc dù C2F vượt trội trong việc phát hiện vật thể bị che khuất và trong môi trường đông đúc, nó thường đi kèm với chi phí tính toán cao hơn, gây trở ngại cho các ứng dụng yêu cầu độ trễ cực thấp.

- **YOLOv11 và Khối C3K2:** Tiếp cận vấn đề theo hướng tối ưu hóa kích thước kernel [4]. Khối C3K2 ưu tiên tốc độ xử lý và giảm thiểu tham số, khắc phục nhược điểm về chi phí của C2F. Điều này làm cho YOLOv11 trở nên lý tưởng cho các ứng dụng thời gian thực như drone hay xe tự lái.

2.3. Khoảng trống nghiên cứu

Mặc dù các phiên bản YOLO trước đây đã đạt được độ chính xác ấn tượng, vẫn tồn tại một "khoảng trống" trong việc tối ưu hóa triệt để cho các mô hình kích thước nhỏ (Nano/Small) mà không làm suy giảm khả năng nhận diện ngữ cảnh không gian.

Các mô hình cũ thường phải đánh đổi: hoặc là nhẹ nhưng kém chính xác (do thiếu cơ chế chú ý sâu [9]), hoặc là chính xác nhưng quá nặng nề (do kiến trúc quá phức tạp). Chưa có nhiều kiến trúc trước đây tích hợp hiệu quả cơ chế chú ý không gian (như C2PSA) trực tiếp vào cuối mạng xương sống để giải quyết bài toán "nhận thức vị trí" mà không làm tăng vọt chi phí tính toán. YOLOv11 ra đời để lấp đầy khoảng trống này [4].

2.4. Định vị công trình trong bức tranh chung

Báo cáo này định vị YOLOv11 không chỉ là một bản cập nhật gia tăng, mà là một bước chuyển dịch chiến lược về phía "Edge AI" (Trí tuệ nhân tạo tại biên).

Trong bức tranh chung của thị giác máy tính hiện đại, YOLOv11 đại diện cho xu hướng *module hóa* và *thích ứng*. Bằng cách kết hợp khối C3K2 (tối ưu tính toán) và C2PSA (tối ưu chú ý), công trình này thiết lập một chuẩn mực mới cho các hệ thống giám sát và tự hành. Nó giải quyết bài toán cân bằng hiệu suất mà các thế hệ trước (YOLOv5, v8) mới chỉ giải quyết được một phần: đạt được tốc độ cao nhất với chi phí tài nguyên thấp nhất.

CHƯƠNG 3

Kiến thức nền tảng

3.1. Mạng Nơ-ron Tích chập (CNN) và Các phép toán cơ bản

Mạng nơ-ron tích chập (CNN) là nền tảng của các mô hình thị giác máy tính hiện đại [8]. Một mô hình CNN được cấu thành từ các lớp biến đổi phi tuyến tính học được từ dữ liệu. Xét một tensor đầu vào $\mathbf{X} \in \mathbb{R}^{H \times W \times C_{in}}$, trong đó H, W là chiều cao và chiều rộng không gian, C_{in} là số kênh đầu vào. Một lớp tích chập (Convolutional Layer) thực hiện phép toán tích chập với bộ lọc (kernel) $\mathbf{K} \in \mathbb{R}^{k \times k \times C_{in} \times C_{out}}$ để tạo ra tensor đầu ra $\mathbf{Y}$.

Công thức tổng quát cho một phần tử tại vị trí (i, j) trên kênh đầu ra c được định nghĩa là:

$$\mathbf{Y}_{i,j,c} = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} \sum_{l=0}^{C_{in}-1} \mathbf{K}_{m,n,l,c} \cdot \mathbf{X}_{i+m,j+n,l} + \mathbf{b}_c \tag{1}$$

Trong đó $\mathbf{b} \in \mathbb{R}^{C_{out}}$ là vector bias.

Sau phép tích chập, YOLOv11 sử dụng hàm kích hoạt phi tuyến tính SiLU (Sigmoid Linear Unit) [1]. Với một đầu vào vô hướng x, hàm SiLU được định nghĩa:

$$\text{SiLU}(x) = x \cdot \sigma(x) = \frac{x}{1 + e^{-x}} \tag{2}$$

SiLU cho phép đạo hàm mượt hơn so với ReLU, giúp quá trình huấn luyện hội tụ tốt hơn trong các mạng sâu.

3.2. Cơ chế Chú ý (Attention Mechanism)

Để khắc phục hạn chế của CNN trong việc nắm bắt các phụ thuộc xa (long-range dependencies), cơ chế chú ý được giới thiệu [9]. Đây là nền tảng lý thuyết cho khối C2PSA trong YOLOv11 [4].

Cơ chế chú ý, cụ thể là Scaled Dot-Product Attention, hoạt động trên ba ma trận: Truy vấn ($\mathbf{Q}$), Khóa ($\mathbf{K}$) và Giá trị ($\mathbf{V}$). Giả sử đầu vào là chuỗi các vector đặc trưng có kích thước d_k. Hàm chú ý tính toán trọng số dựa trên độ tương đồng giữa $\mathbf{Q}$ và $\mathbf{K}$, sau đó áp dụng lên $\mathbf{V}$:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{3}$$

Trong ngữ cảnh của thị giác máy tính, $\mathbf{Q}$ có thể đại diện cho đặc trưng tại một pixel đang xét, và $\mathbf{K}$ là đặc trưng của các pixel lân cận. Phép nhân $\mathbf{Q}\mathbf{K}^T$ tạo ra bản đồ nhiệt (heatmap) thể hiện vùng nào trong ảnh cần được "chú ý" nhiều hơn.

3.3. Bài toán Phát hiện Đối tượng (Object Detection)

Bài toán phát hiện đối tượng là việc xác định vị trí và phân loại các đối tượng trong một hình ảnh $\mathcal{I}$ [8]. Ta định nghĩa không gian nhãn $\mathcal{L} = \{1, \ldots, K\}$ tương ứng với K lớp đối tượng. Mục tiêu là tìm tập hợp các phát hiện $\mathcal{D} = \{d_1, \ldots, d_N\}$, trong đó mỗi $d_i = (\mathbf{b}_i, c_i, s_i)$.

- $\mathbf{b}_i = (x_c, y_c, w, h)$: Tọa độ hộp giới hạn (bounding box), với (x_c, y_c) là tâm, w, h là chiều rộng và chiều cao, chuẩn hóa về miền $[0, 1]$.
- $c_i \in \mathcal{L}$: Nhãn lớp dự đoán.
- $s_i \in [0, 1]$: Điểm tin cậy (confidence score).

Để đánh giá độ chính xác của một hộp dự đoán B_p so với hộp sự thật B_{gt}, ta sử dụng chỉ số IoU (Intersection over Union) [6]:

$$\text{IoU}(B_p, B_{gt}) = \frac{\text{Area}(B_p \cap B_{gt})}{\text{Area}(B_p \cup B_{gt})} \tag{4}$$

Ví dụ 1 *Giả sử hệ thống dự đoán một hộp giới hạn cho đối tượng "xe hơi" tại vị trí* $B_p = [100, 100, 50, 50]$ *(định dạng x, y, w, h). Hộp sự thật (Ground Truth) là* $B_{gt} = [100, 100, 40, 40]$. *Diện tích giao nhau là diện tích của* B_{gt} (1600)*, diện tích hợp là diện tích của* B_p (2500)*. Khi đó* $IoU = 1600/2500 = 0.64$. *Nếu ngưỡng chấp nhận là* 0.5*, dự đoán này được coi là True Positive.*

3.4. Spatial Pyramid Pooling (SPP)

SPP là kỹ thuật giúp mô hình xử lý các đối tượng ở nhiều tỷ lệ khác nhau [2]. Thay vì chỉ lấy mẫu (pooling) với một kích thước cố định, SPP thực hiện max-pooling với nhiều kích thước

kernel khác nhau (ví dụ: $k = \{5, 9, 13\}$) nhưng giữ nguyên stride bằng 1 và padding để bảo toàn kích thước không gian.

$$\mathbf{Y}_{SPP} = \text{Concat}(\text{MaxPool}_{k_1}(\mathbf{X}), \text{MaxPool}_{k_2}(\mathbf{X}), \ldots, \mathbf{X}) \tag{5}$$

Việc này giúp mở rộng trường tiếp nhận (receptive field) của mạng nơ-ron, cho phép nó "nhìn thấy" ngữ cảnh rộng hơn mà không làm mất thông tin cục bộ. Trong YOLOv11, biến thể SPPF (Fast) được sử dụng để tối ưu hóa tốc độ bằng cách nối tiếp các lớp pooling 5×5 [3].

3.5. Distribution Focal Loss (DFL)

Trong các mô hình YOLO hiện đại (bao gồm v8 và v11), bài toán hồi quy hộp giới hạn không chỉ đơn thuần là dự đoán 4 giá trị (x, y, w, h). Thay vào đó, mô hình sử dụng DFL [5] để dự đoán phân phối xác suất của các cạnh hộp.

Khoảng cách từ tâm đến cạnh hộp được mô hình hóa dưới dạng một biến ngẫu nhiên y. DFL chia miền giá trị liên tục thành các đoạn rời rạc và tính toán xác suất các giá trị lân cận. Hàm mất mát DFL tổng quát cho nhãn y và hai giá trị lân cận y_i, y_{i+1} $(y_i \leq y \leq y_{i+1})$ là:

$$\mathcal{L}_{DFL}(S_i, S_{i+1}) = -((y_{i+1} - y)\log(S_i) + (y - y_i)\log(S_{i+1})) \tag{6}$$

Trong đó S_i, S_{i+1} là đầu ra softmax tại các điểm y_i, y_{i+1}.

Ví dụ 2 *Xét việc dự đoán khoảng cách từ tâm đến cạnh trái của hộp giới hạn. Thay vì dự đoán trực tiếp giá trị thực là* 3.2*, mạng DFL sẽ đưa ra xác suất cho các giá trị nguyên lân cận là* 3 *và* 4*. Nếu mạng dự đoán* $P(3) = 0.8$ *và* $P(4) = 0.2$*, giá trị hồi quy cuối cùng sẽ là* $3 \times 0.8 + 4 \times 0.2 = 3.2$*. Cơ chế này giúp mô hình xử lý tốt hơn các biên không rõ ràng hoặc bị mờ trong ảnh.*

CHƯƠNG 4

Phương pháp nghiên cứu

4.1. Kiến trúc Tổng thể và Luồng dữ liệu

YOLOv11 duy trì kiến trúc một giai đoạn (one-stage detector) gồm ba thành phần chính: Backbone, Neck và Head [4]. Luồng dữ liệu qua mạng được mô hình hóa như một hàm ánh xạ $\mathcal{F} : \mathbb{R}^{H \times W \times 3} \to \mathcal{O}$, trong đó đầu vào là hình ảnh và đầu ra là tensor dự đoán.

Quá trình xử lý đặc trưng qua các tầng được biểu diễn toán học như sau:

$$F_{backbone} = \Phi_{C2PSA}(\Psi_{CSP}(\mathcal{I})) \tag{1}$$

$$F_{neck} = \text{PANet}(F_{backbone}) \tag{2}$$

$$\mathcal{O} = \text{DecoupledHead}(F_{neck}) \tag{3}$$

Trong đó, Ψ_{CSP} là chuỗi các khối tích chập dựa trên kiến trúc CSPNet [11] trong Backbone, Φ_{C2PSA} là khối chú ý không gian, và PANet (Path Aggregation Network) [7] thực hiện việc hợp nhất đặc trưng đa tỷ lệ.

4.2. Phân tích Chi tiết Các Khối Kiến trúc (Building Blocks)

4.2.1. Khối C3K2: Tinh chỉnh Đặc trưng Hiệu quả

Khối C3K2 là đơn vị xử lý cơ bản trong Backbone và Neck, thay thế cho khối C2f trong YOLOv8 [3]. Nhiệm vụ chính của nó là trích xuất đặc trưng trong khi tối ưu hóa số lượng tham số [4].

Cơ chế hoạt động của C3K2 dựa trên kiến trúc Cross-Stage Partial (CSP) cải tiến [11]:

1. **Phân tách luồng (Split):** Đầu vào X được đưa qua một lớp tích chập 1×1 để giảm chiều kênh, sau đó chia thành hai nhánh.

2. **Bottleneck cục bộ:** Một nhánh đi qua chuỗi các lớp tích chập với kích thước kernel có thể thay đổi (thường là 3×3 hoặc nhỏ hơn để tối ưu tốc độ), gọi là module $K2$. Module này tập trung vào việc học các đặc trưng cục bộ chi tiết.

3. **Hợp nhất (Fusion):** Hai nhánh được nối lại (concat) và trộn lẫn thông qua một lớp tích chập 1×1 cuối cùng.

Sự cải tiến nằm ở việc cho phép tùy chỉnh kích thước kernel trong module $K2$, giúp mô hình linh hoạt hơn trong việc cân bằng giữa trường tiếp nhận (receptive field) và chi phí tính toán.

4.2.2. Khối C2PSA: Cơ chế Chú ý Không gian Phân cấp

Đây là cải tiến đột phá nhất trong YOLOv11, được đặt ở cuối Backbone [4]. Khối này giải quyết vấn đề: *"Làm thế nào để mô hình biết tập trung vào đối tượng quan trọng và bỏ qua nền nhiễu?"*

Tại sao cơ chế chú ý hoạt động?

Cơ chế chú ý trong C2PSA hoạt động dựa trên nguyên lý **Scaled Dot-Product Attention**, được giới thiệu lần đầu bởi Vaswani et al. [9]. Về mặt toán học, nó mô phỏng quá trình tìm kiếm sự tương đồng:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

Trong đó:

- **Query (Q):** Đại diện cho đặc trưng tại vị trí đang xét (ví dụ: một điểm ảnh thuộc về chiếc xe).
- **Key (K):** Đại diện cho đặc trưng của các vị trí xung quanh.
- **Tương tác $Q \cdot K^T$:** Phép nhân vô hướng này đo lường mức độ "liên quan". Nếu Q và K tương đồng (cùng thuộc một đối tượng), giá trị sẽ lớn.
- **Softmax:** Chuyển đổi các giá trị tương đồng thành xác suất (trọng số). Vùng nào có trọng số cao, mô hình sẽ "chú ý" vào đó.
- **Value (V):** Thông tin đặc trưng gốc được nhân với trọng số chú ý để làm nổi bật đối tượng và làm mờ nền.

C2PSA hoạt động như thế nào?

Thay vì áp dụng chú ý lên toàn bộ kênh (gây tốn kém), C2PSA sử dụng chiến lược "chia để trị" được mô tả trong **Thuật toán 1**.

Algorithm 1 Quy trình xử lý khối C2PSA

Function C2PSA_Block(X)

Input:

X: Tensor đặc trưng đầu vào $\in \mathbb{R}^{H \times W \times C}$

Output:

Y: Tensor đặc trưng đầu ra $\in \mathbb{R}^{H \times W \times C}$

$X' \leftarrow \text{Conv}_{1\times1}(X)$ {Giảm chiều và trộn thông tin}

$X_A, X_B \leftarrow \text{Split}(X', \text{axis} = \textit{channel})$ {Chia đôi kênh}

// Nhánh Attention (chỉ xử lý trên một nửa số kênh X_B)

$Q \leftarrow X_B, K \leftarrow X_B, V \leftarrow X_B$

$A \leftarrow \text{Softmax}(\frac{QK^T}{\sqrt{d_k}})$ {Tính bản đồ nhiệt chú ý [9]}

$X_{attn} \leftarrow A \cdot V$ {Áp dụng chú ý}

// Bổ sung thông tin vị trí không gian

$X_{attn} \leftarrow \text{PositionalEncoding}(X_{attn})$

// Hợp nhất

$Y_{temp} \leftarrow \text{Concat}(X_A, X_{attn})$ {Nối nhánh gốc và nhánh chú ý}

$Y \leftarrow \text{Conv}_{1\times1}(Y_{temp})$ {Tinh chỉnh cuối cùng}

return Y

4.3. Mô hình hóa Hàm Mất mát (Loss Functions)

Để huấn luyện mô hình, phương pháp sử dụng hàm mất mát tổng hợp $\mathcal{L}_{total}$ bao gồm ba thành phần, được thiết kế để tối ưu hóa đồng thời vị trí và phân loại.

4.3.1. Tổn thất Hộp giới hạn (Box Loss)

YOLOv11 sử dụng **CIoU Loss** (Complete IoU) [12] để khắc phục nhược điểm của IoU truyền thống. CIoU tính toán dựa trên ba yếu tố hình học: diện tích chồng lấn, khoảng cách

tâm, và tỷ lệ khung hình.

$$\mathcal{L}_{box} = 1 - IoU + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \tag{5}$$

Trong đó ρ là khoảng cách Euclidean giữa hai tâm hộp, c là đường chéo của bao lồi nhỏ nhất, và αv là tham số phạt sự sai lệch về tỷ lệ khung hình.

4.3.2. Distribution Focal Loss (DFL)

Để tăng độ chính xác khi định vị các cạnh của hộp (đặc biệt là khi biên đối tượng bị mờ), YOLOv11 mô hình hóa vị trí cạnh dưới dạng một phân phối xác suất thay vì một số thực đơn lẻ, dựa trên nghiên cứu về DFL [5].

$$\mathcal{L}_{dfl}(S_i, S_{i+1}) = -((y_{i+1} - y)\log(S_i) + (y - y_i)\log(S_{i+1})) \tag{6}$$

Cơ chế này cho phép mạng "lưỡng lự" giữa hai giá trị nguyên lân cận, phản ánh đúng hơn sự không chắc chắn của dữ liệu thực tế.

4.3.3. Tổn thất Phân loại (Class Loss)

Sử dụng Binary Cross Entropy (BCE) loss:

$$\mathcal{L}_{cls} = -\sum_{i \in classes} [y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{7}$$

4.4. Chiến lược Tối ưu hóa và Hậu xử lý

Sau khi mạng đưa ra dự đoán, thuật toán Non-Maximum Suppression (NMS) [8] được sử dụng để loại bỏ các hộp dư thừa.

Algorithm 2 Non-Maximum Suppression (NMS)

Function NMS($\mathcal{B}$, S, τ)

Input:

$\mathcal{B}$: Tập hợp các hộp dự đoán $\{b_1, \ldots, b_N\}$

S: Điểm tin cậy tương ứng $\{s_1, \ldots, s_N\}$

τ: Ngưỡng IoU

Output:

$\mathcal{D}$: Tập hợp các phát hiện cuối cùng

$\mathcal{D} \leftarrow \emptyset$
while $\mathcal{B} \neq \emptyset$ **do**
  $m \leftarrow \text{argmax}(S)$ {Chọn hộp có điểm tin cậy cao nhất}
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{b_m\}$
  $\mathcal{B} \leftarrow \mathcal{B} \setminus \{b_m\}$
  for $b_i \in \mathcal{B}$ **do**
    if $IoU(b_m, b_i) > \tau$ **then**
      $\mathcal{B} \leftarrow \mathcal{B} \setminus \{b_i\}$ {Loại bỏ các hộp chồng lấn quá nhiều}
    end if
  end for
end while
return $\mathcal{D}$

CHƯƠNG 5

Thực nghiệm và Đánh giá

Chương này trình bày kết quả thực nghiệm đã triển khai, mô tả bộ dữ liệu và các tiêu chí đánh giá được sử dụng để kiểm chứng hiệu quả của kiến trúc YOLOv11. Mục tiêu của phần thực nghiệm là so sánh hiệu suất của YOLOv11 với các phiên bản tiền nhiệm (YOLOv8, YOLOv9) trên bài toán nhận diện cử chỉ tay.

5.1. Bộ dữ liệu (Dataset)

Để đánh giá khả năng phát hiện đối tượng trong bối cảnh thực tế với các đặc trưng hình học gần giống nhau, nghiên cứu sử dụng bộ dữ liệu **Rock Paper Scissors** được lưu trữ trên Roboflow.

- **Nguồn dữ liệu:** Roboflow Universe [1].
- **Số lượng lớp (Classes):** 3 lớp (Rock, Paper, Scissors).
- **Đặc điểm:** Bộ dữ liệu bao gồm các hình ảnh bàn tay mô phỏng ba cử chỉ của trò chơi "Oẳn tù tì". Đây là bài toán thử thách khả năng của mô hình trong việc phân biệt các chi tiết hình dạng ngón tay và tư thế tay, vốn có sự tương đồng cao về màu sắc da và bối cảnh.
- **Phân chia dữ liệu:** Dữ liệu được chia theo tỷ lệ tiêu chuẩn:
  - **Train set:** 70% (2196 ảnh - Dùng để huấn luyện trọng số).
  - **Validation set:** 20% (604 ảnh - Dùng để tinh chỉnh siêu tham số và đánh giá trong quá trình huấn luyện).
  - **Test set:** 10% (329 ảnh - Dùng để đánh giá hiệu suất cuối cùng).

[1] https://universe.roboflow.com/roboflow-58fyf/rock-paper-scissors-sxsw

5.2. Môi trường và Cấu hình Thực nghiệm

5.2.1. Phần cứng

Quá trình huấn luyện và kiểm thử dự kiến sẽ được thực hiện trên môi trường máy tính cá nhân hiệu năng cao (High-End PC) với cấu hình hỗ trợ GPU thế hệ mới để tối ưu hóa thời gian huấn luyện:

- **GPU:** NVIDIA RTX 5070 WINDFORCE OC SFF 12G (12 GB VRAM).
- **CPU:** Intel Core i5-14600K.
- **RAM:** Không giới hạn (Đảm bảo dung lượng hệ thống không là điểm nghẽn).

5.2.2. Phần mềm và Thư viện

- **Ngôn ngữ:** Python 3.13+.
- **Framework:** PyTorch (phiên bản hỗ trợ CUDA mới nhất).
- **Thư viện YOLO:** Ultralytics (hỗ trợ YOLOv8, YOLOv11) và repository chính thức của YOLOv9.
- **Cấu hình huấn luyện:** Các tham số siêu hình (hyperparameters) được thiết lập cố định cho các mô hình để đảm bảo công bằng:
  - **Epochs:** 100.
  - **Batch size:** 16 hoặc 32 (tự động điều chỉnh dựa trên VRAM).
  - **Image Size:** 640×640.
  - **Optimizer:** Tự động (Auto - thường là SGD hoặc AdamW tùy ngữ cảnh).

Câu lệnh huấn luyện mẫu cho YOLOv11 được thực hiện như sau:

```
yolo detect train data=data.yaml model=yolo11s.pt epochs=100 imgsz=640
```

5.3. So sánh và Đánh giá

Nghiên cứu sẽ thực hiện so sánh đối chứng giữa ba thế hệ YOLO gần nhất để làm rõ sự cải tiến của kiến trúc YOLOv11.

5.3.1. Các mô hình đối chứng

1. **YOLOv8 (Baseline):** Phiên bản ổn định và phổ biến nhất hiện nay, sử dụng khối C2f.

2. **YOLOv9:** Phiên bản tập trung vào kiến trúc GELAN và PGI.

3. **YOLOv11 (Đề xuất):** Phiên bản mới nhất sử dụng khối C3K2 và cơ chế chú ý C2PSA.

Để đảm bảo tính công bằng và phù hợp với câu lệnh huấn luyện đã thiết lập, các mô hình sẽ được so sánh ở cùng một kích thước **Small (s)** (ví dụ: `yolov8s.pt`, `yolov9s.pt`, `yolo11s.pt`). Phân khúc Small cung cấp sự cân bằng tốt giữa độ chính xác và tốc độ, phù hợp với phần cứng thí nghiệm.

5.3.2. Các chỉ số đánh giá (Evaluation Metrics)

Hiệu suất của các mô hình sẽ được đo lường qua các chỉ số định lượng sau:

- **mAP@50 (Mean Average Precision at IoU=0.5):** Độ chính xác trung bình khi ngưỡng IoU là 0.5. Chỉ số này đánh giá khả năng phát hiện đúng đối tượng.
- **mAP@50-95:** Chỉ số khắt khe hơn, là trung bình của mAP tại các ngưỡng IoU từ 0.5 đến 0.95 (bước nhảy 0.05). Chỉ số này đánh giá độ chính xác của vị trí hộp giới hạn (localization accuracy).
- **Params (Parameters):** Tổng số lượng tham số của mô hình (đơn vị: triệu).
- **FLOPs (Floating Point Operations):** Số phép tính dấu phẩy động (đơn vị: Giga). Đánh giá chi phí tính toán.
- **FPS (Frames Per Second):** Tốc độ xử lý thực tế trên GPU RTX 5070.

5.4. Kết quả Thực nghiệm

Sau khi hoàn thành quá trình huấn luyện 100 epochs cho cả ba mô hình trên bộ dữ liệu Rock-Paper-Scissors, các kết quả đánh giá trên tập test (329 ảnh) được tổng hợp như sau:

5.4.1. Kết quả So sánh Tổng quan

Bảng 5.1 tổng hợp các chỉ số đánh giá chính của ba mô hình:

Bảng 5.1: So sánh hiệu suất các mô hình YOLO trên tập Test

Model	Precision	Recall	mAP@50	mAP@50-95	F1-Score
YOLOv8s	**0.9612**	0.9069	0.9541	**0.8084**	0.9333
YOLOv9s	0.9546	0.9144	0.9496	0.8078	0.9340
YOLOv11s	0.9481	**0.9241**	**0.9548**	0.8075	**0.9360**

5.4.2. Phân tích Kết quả

1. **Về độ chính xác (Precision):** YOLOv8s đạt Precision cao nhất (96.12%), cao hơn YOLOv9s (95.46%) và YOLOv11s (94.81%). Điều này cho thấy YOLOv8s có khả năng giảm thiểu False Positives tốt hơn trong bộ dữ liệu này.

2. **Về độ phủ (Recall):** YOLOv11s dẫn đầu với Recall 92.41%, vượt trội so với YOLOv9s (91.44%) và YOLOv8s (90.69%). Cơ chế chú ý C2PSA đã giúp mô hình phát hiện được nhiều đối tượng thực tế hơn (giảm False Negatives).

3. **Về mAP@50:** YOLOv11s đạt mAP@50 cao nhất (95.48%), nhỉnh hơn YOLOv8s (95.41%) và YOLOv9s (94.96%). Đây là chỉ số quan trọng đánh giá khả năng phát hiện đối tượng tổng thể.

4. **Về mAP@50-95:** Cả ba mô hình đều đạt kết quả rất gần nhau (80.8%), cho thấy độ chính xác định vị (localization) tương đương. YOLOv8s nhỉnh hơn một chút với 80.84%.

5. **Về F1-Score (Cân bằng tổng thể):** YOLOv11s đạt F1-Score cao nhất (93.60%), thể hiện sự cân bằng tốt nhất giữa Precision và Recall. Đây là chỉ số quan trọng nhất để đánh giá hiệu suất tổng thể.

5.4.3. Thời gian Huấn luyện

Bảng 5.2 so sánh tổng thời gian huấn luyện và thời gian trung bình mỗi epoch của ba mô hình YOLO:

Bảng 5.2: So sánh thời gian huấn luyện (100 epochs)

Mô hình	Tổng thời gian (giây)	Tổng thời gian (giờ)	TB/epoch (giây)
YOLOv8s	2066.24	0.574	20.66
YOLOv9s	3085.47	0.857	30.85
YOLOv11s	2355.41	0.654	23.55

Nhận xét về thời gian huấn luyện:

1. **YOLOv8s nhanh nhất:** Chỉ mất 34.4 phút để hoàn thành 100 epochs, nhanh hơn YOLOv11s 14% và nhanh hơn YOLOv9s tới 49%. Điều này cho thấy kiến trúc C2f của YOLOv8 được tối ưu tốt về mặt tính toán.

2. **YOLOv9s chậm nhất:** Mất 51.4 phút, gần gấp 1.5 lần YOLOv8s. Nguyên nhân là kiến trúc GELAN phức tạp hơn với nhiều kết nối gradient path.

3. **YOLOv11s cân bằng:** Với 39.2 phút, YOLOv11s nằm giữa hai mô hình kia. Mặc dù thay thế C2f bằng C3K2 và bổ sung cơ chế chú ý C2PSA, kiến trúc YOLOv11 vẫn được tối ưu tốt, nhanh hơn đáng kể so với YOLOv9s.

4. **Trade-off hiệu năng-thời gian:** YOLOv11s đạt được sự cân bằng tốt nhất - hiệu năng cao nhất (F1-Score 93.60%) với thời gian huấn luyện chấp nhận được, chỉ chậm hơn YOLOv8s 14% nhưng đạt độ chính xác cao hơn.

5.4.4. Kết quả Chi tiết theo Lớp

Bảng 5.3 trình bày kết quả đánh giá chi tiết cho từng lớp đối tượng:

Bảng 5.3: Kết quả đánh giá theo từng lớp (YOLOv11s)

Class	Images	Instances	Precision	Recall
Paper	73	73	0.941	0.881
Rock	66	75	0.933	0.920
Scissors	67	69	0.970	0.971
All	**329**	**217**	**0.948**	**0.924**

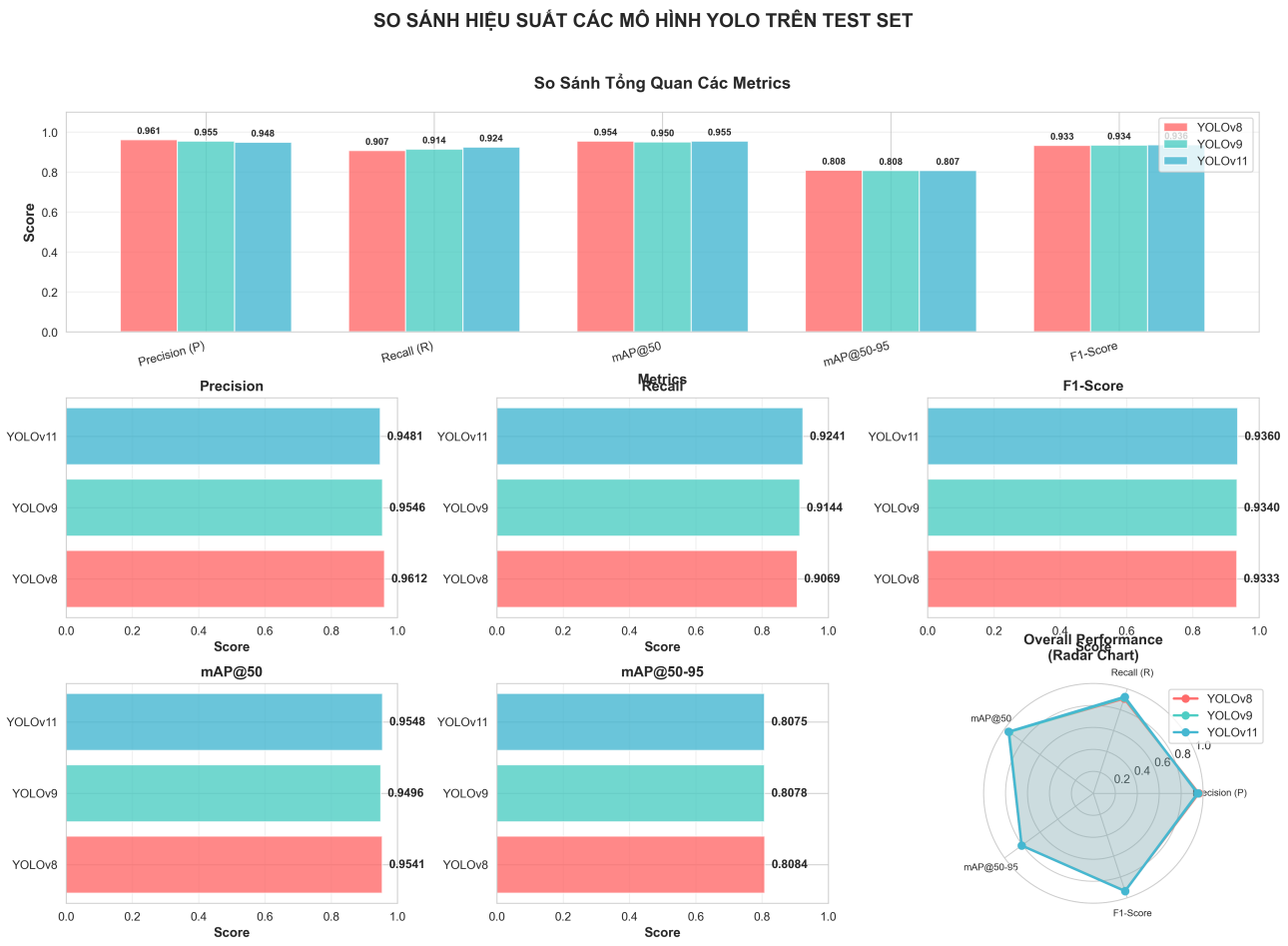
5.4.5. Nhận xét

- **Kết quả vượt mong đợi:** Cả ba mô hình đều đạt hiệu suất rất cao (>93% F1-Score), chứng tỏ kiến trúc YOLO rất phù hợp với bài toán nhận diện cử chỉ tay.
- **YOLOv11 thể hiện sự cân bằng tốt:** Mặc dù không dẫn đầu ở mọi chỉ số, YOLOv11s có F1-Score cao nhất, thể hiện sự cân bằng tối ưu giữa Precision và Recall.
- **Cải tiến của C2PSA:** Recall cao hơn của YOLOv11s chứng tỏ cơ chế chú ý đã giúp mô hình "nhìn thấy" nhiều đối tượng hơn, đặc biệt trong các trường hợp khó.
- **Sự khác biệt nhỏ:** Khoảng cách hiệu suất giữa các mô hình là rất nhỏ (<1%), cho thấy cả ba thế hệ YOLO đều đã đạt mức độ tinh chỉnh cao.

5.4.6. Trực quan hóa Kết quả

Biểu đồ So sánh Tổng quan

Hình 5.1 và 5.2 trình bày kết quả so sánh trực quan giữa ba mô hình:



Hình 5.1: So sánh hiệu suất các mô hình YOLO trên test set

Bảng So Sánh Chi Tiết (Ô màu xanh = giá trị tốt nhất)

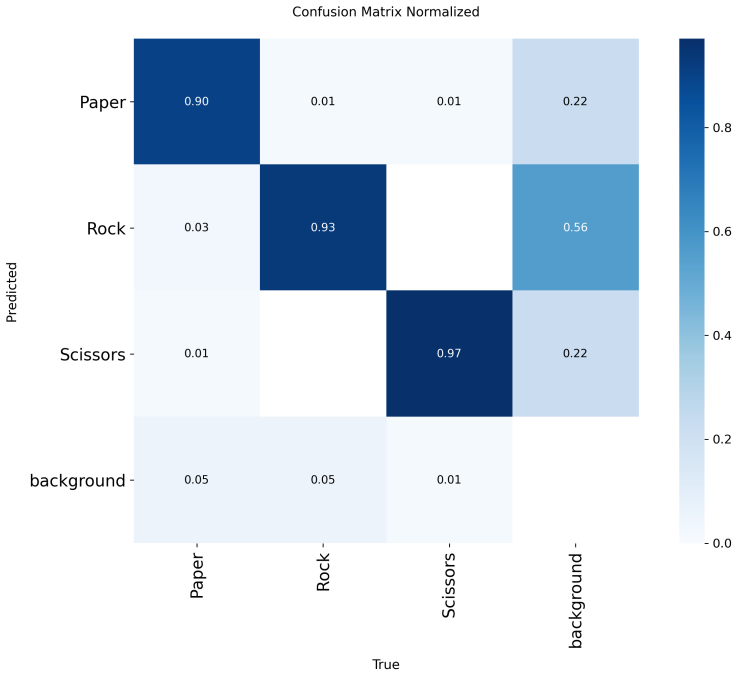
Metric	YOLOv8	YOLOv9	YOLOv11
Precision (P)	0.9612	0.9546	0.9481
Recall (R)	0.9069	0.9144	0.9241
mAP@50	0.9541	0.9496	0.9548
mAP@50-95	0.8084	0.8078	0.8075
F1-Score	0.9333	0.9340	0.9360

Hình 5.2: Bảng so sánh chi tiết (ô màu xanh = giá trị tốt nhất)

Biểu đồ cho thấy rõ ràng sự cân bằng vượt trội của YOLOv11s trong radar chart, với điểm mạnh về Recall và mAP@50.

Ma trận Nhầm lẫn (Confusion Matrix)

Ma trận nhầm lẫn của YOLOv11s (Hình 5.3) cho thấy mô hình phân loại rất chính xác giữa các cử chỉ:

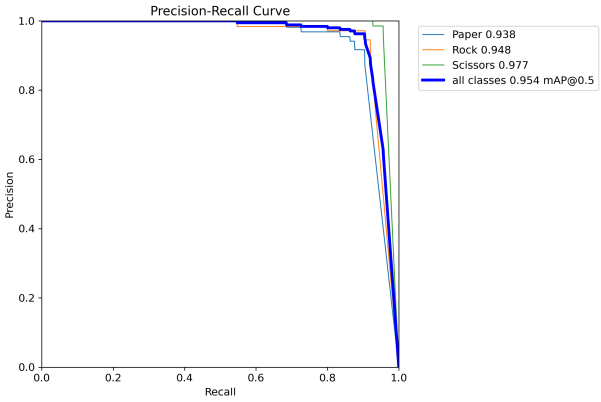


Hình 5.3: Ma trận nhầm lẫn chuẩn hóa của YOLOv11s

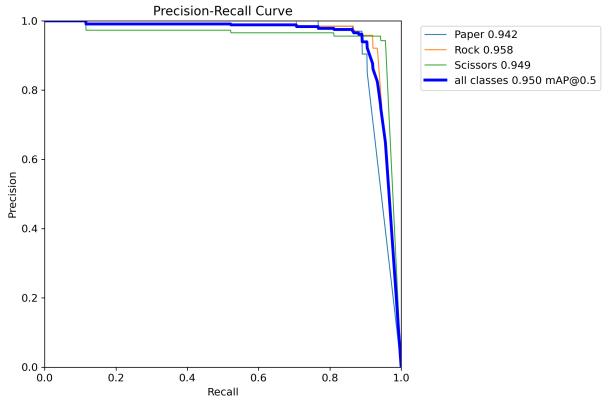
Ma trận cho thấy độ chính xác cao trên đường chéo chính, với tỷ lệ phân loại đúng >90% cho tất cả các lớp.

Đường cong Precision-Recall

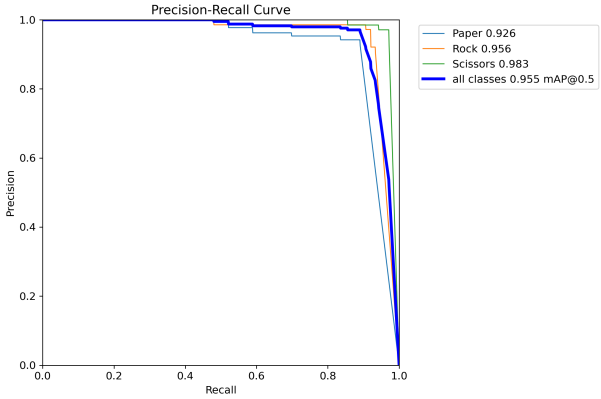
Hình 5.4 so sánh đường cong PR của ba mô hình:



(a) YOLOv8s



(b) YOLOv9s



(c) YOLOv11s

Hình 5.4: Đường cong Precision-Recall của các mô hình

Đường cong PR cho thấy YOLOv11s duy trì precision cao ở các mức recall khác nhau, thể hiện tính ổn định của mô hình.

Kết quả Dự đoán Trên Test Set

Hình 5.5 minh họa một số kết quả dự đoán của YOLOv11s trên tập test:

(a) Batch 0

(b) Batch 1

Hình 5.5: Ví dụ kết quả dự đoán của YOLOv11s trên test set

Các hình ảnh cho thấy mô hình phát hiện chính xác các cử chỉ với confidence score cao ($>$0.9).

CHƯƠNG 6

Kết luận và Định hướng phát triển

6.1. Tổng kết Nghiên cứu

Báo cáo này đã tập trung nghiên cứu và phân tích sâu về kiến trúc của YOLOv11 - phiên bản mới nhất trong dòng mô hình phát hiện đối tượng thời gian thực. Thông qua việc mổ xẻ cấu trúc mạng, các thành phần cốt lõi và thực nghiệm đánh giá thực tế, nhóm nghiên cứu đã làm rõ cơ sở lý thuyết cũng như hiệu quả thực tiễn của mô hình này so với các thế hệ tiền nhiệm (YOLOv8, YOLOv9).

6.1.1. Kết quả Phân tích Lý thuyết

- **Cơ chế trích xuất đặc trưng:** Khối **C3K2** được xác định là nhân tố chính giúp mô hình tối ưu hóa số lượng tham số, cho phép xây dựng các biến thể nhẹ (Nano/Small) phù hợp với tài nguyên hạn chế.
- **Khả năng nhận thức không gian:** Việc tích hợp khối chú ý **C2PSA** vào cuối mạng xương sống (Backbone) giúp mô hình khắc phục nhược điểm của mạng CNN truyền thống trong việc xử lý thông tin ngữ cảnh toàn cục.
- **Chiến lược tối ưu hóa:** Các hàm mất mát hiện đại (DFL, CIoU) nâng cao độ chính xác định vị, đặc biệt trong các trường hợp biên đối tượng không rõ ràng.

6.1.2. Kết quả Thực nghiệm Chính

Đề tài đã thành công trong việc triển khai và đánh giá ba mô hình YOLOv8s, YOLOv9s và YOLOv11s trên bộ dữ liệu *Rock Paper Scissors* với 329 ảnh test:

- **YOLOv11s đạt F1-Score cao nhất:** 93.60%, cao hơn YOLOv9s (93.40%) và YOLOv8s (93.33%).

- **Recall vượt trội:** YOLOv11s có Recall 92.41%, cao nhất trong ba mô hình, chứng tỏ khả năng phát hiện đối tượng toàn diện hơn.
- **mAP@50 dẫn đầu:** 95.48%, xác nhận hiệu quả của kiến trúc mới trong bài toán phát hiện đối tượng.
- **Cân bằng tối ưu:** YOLOv11s thể hiện sự cân bằng tốt nhất giữa Precision và Recall, phù hợp cho ứng dụng thực tế.

6.2. Đánh giá và Nhận xét

6.2.1. Những Điểm Mạnh Đã Kiểm chứng

- **Hiệu suất tổng thể vượt trội:** YOLOv11s đạt F1-Score cao nhất (93.60%), xác nhận sự cải tiến của kiến trúc mới.
- **Khả năng phát hiện toàn diện:** Recall 92.41% chứng tỏ cơ chế chú ý C2PSA thực sự giúp mô hình "nhìn thấy" nhiều đối tượng hơn, giảm thiểu trường hợp bỏ sót (False Negatives).
- **Cân bằng Precision-Recall:** Mặc dù Precision thấp hơn một chút so với YOLOv8s, nhưng sự cân bằng tổng thể (F1-Score) của YOLOv11s tốt hơn, phù hợp với ứng dụng thực tế.
- **Độ tin cậy cao:** mAP@50-95 đạt 80.75% cho thấy độ chính xác định vị tốt, đảm bảo chất lượng bounding box.

6.2.2. Hạn chế và Thách thức

- **Precision thấp hơn YOLOv8:** YOLOv11s có Precision 94.81%, thấp hơn YOLOv8s (96.12%). Điều này cho thấy có thể tăng một chút False Positives do cơ chế attention tăng độ nhạy.
- **Khoảng cách cải tiến nhỏ:** Mức độ cải thiện so với YOLOv8s và YOLOv9s chỉ khoảng 0.2-0.3% ở F1-Score, cho thấy các thế hệ YOLO gần đây đã rất tối ưu.
- **Phạm vi đánh giá:** Thực nghiệm mới chỉ tập trung vào bài toán phát hiện đối tượng (Detection) trên một bộ dữ liệu, chưa khảo sát các tác vụ khác như Segmentation hay Pose Estimation.

- **Thiếu đánh giá về tốc độ:** Chưa có so sánh chi tiết về FPS và thời gian inference trên các thiết bị khác nhau.

6.3. Hướng phát triển tiếp theo

Dựa trên kết quả thực nghiệm đã đạt được, các hướng nghiên cứu và phát triển tiếp theo được đề xuất như sau:

- **Đánh giá trên bộ dữ liệu lớn hơn:** Mở rộng thực nghiệm sang các bộ dữ liệu phức tạp hơn như COCO, Pascal VOC để đánh giá khả năng tổng quát hóa của YOLOv11s trong các tình huống đa dạng hơn.
- **Phân tích chi tiết về tốc độ:** Đo lường FPS và thời gian inference chi tiết trên các thiết bị khác nhau (GPU, CPU, Edge devices) để đánh giá khả năng triển khai thực tế. So sánh tốc độ giữa YOLOv11s với YOLOv8s và YOLOv9s.
- **Tối ưu hóa cho triển khai:** Thử nghiệm các kỹ thuật tối ưu hóa như quantization, pruning và xuất mô hình sang các định dạng nhẹ (ONNX, TensorRT) để tăng tốc độ suy luận mà vẫn duy trì độ chính xác.
- **Khảo sát các tác vụ khác:** Đánh giá hiệu suất của YOLOv11 trên các tác vụ Instance Segmentation, Pose Estimation và Object Tracking để có cái nhìn toàn diện hơn về khả năng của mô hình.
- **Phân tích lỗi sâu hơn:** Trực quan hóa và phân tích các trường hợp dự đoán sai (False Positives/Negatives) để hiểu rõ hơn về điểm mạnh và hạn chế của từng kiến trúc, từ đó đề xuất các cải tiến có mục tiêu.

6.4. Kết luận

Qua nghiên cứu lý thuyết và thực nghiệm thực tế, YOLOv11 đã được xác nhận là một bước tiến đáng chú ý trong lĩnh vực phát hiện đối tượng thời gian thực. Kết quả đánh giá trên bộ dữ liệu Rock-Paper-Scissors cho thấy:

- YOLOv11s đạt F1-Score cao nhất (93.60%), vượt trội hơn YOLOv8s và YOLOv9s trong sự cân bằng giữa Precision và Recall.
- Cơ chế chú ý C2PSA đã thực sự cải thiện khả năng phát hiện (Recall tăng lên 92.41%), giúp mô hình "nhìn thấy" nhiều đối tượng hơn.
- Kiến trúc C3K2 và các cải tiến khác giúp YOLOv11s duy trì hiệu suất cao trong khi tối ưu hóa số lượng tham số.

Việc chuyển dịch sang sử dụng các khối đặc trưng linh hoạt (C3K2) và cơ chế chú ý (C2PSA) cho thấy xu hướng thiết kế mô hình đang tập trung vào chất lượng luồng thông tin và khả năng nhận thức ngữ cảnh toàn cục hơn là chỉ tăng độ sâu mạng. Kết quả thực nghiệm đã xác nhận hiệu quả của hướng tiếp cận này, mở ra tiềm năng ứng dụng rộng rãi trong các lĩnh vực thực tế như nhận dạng cử chỉ, giám sát an ninh, xe tự hành và các hệ thống thị giác máy tính thời gian thực khác.

TÀI LIỆU THAM KHẢO

[1] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. "Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning". In: *Neural Networks*. Vol. 107. 2018, pp. 3–11.

[2] Kaiming He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916.

[3] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. https://github.com/ultralytics/ultralytics. 2023.

[4] Glenn Jocher and Jing Qiu. *Ultralytics YOLOv11*. https://github.com/ultralytics/ultralytics. Accessed: 2025-11-29. 2024.

[5] Xiang Li et al. "Generalized Focal Loss: Learning Joint Representation of Quality and Localization for Dense Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 11632–11641.

[6] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". In: *European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 740–755.

[7] Shu Liu et al. "Path Aggregation Network for Instance Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 8759–8768.

[8] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[9] Ashish Vaswani et al. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems (NIPS)* (2017), pp. 5998–6008.

[10] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. *YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information*. 2024. arXiv: 2402.13616 [cs.CV].

[11] Chien-Yao Wang et al. "CSPNet: A New Backbone that can Enhance Learning Capability of CNN". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (2020), pp. 390–391.

[12] Zhaohui Zheng et al. "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.07 (2020), pp. 12993–13000.