**Court Detection (Nhận dạng cấu trúc sân)**

[*https://github.com/house3173/CourtDetectionOffical.git*](https://github.com/house3173/CourtDetectionOffical.git)

# Tổng quan

* Phân loại bài toán: Keypoints Detection
* Mục tiêu: Nhận diện cấu trúc sân cầu lông và sân tennis

|  |  |
| --- | --- |
|  |  |
| Hình 1.1 Cấu trúc sân cầu lông (22 keypoints) | Hình 1.2 Cấu trúc sân tennis (14 keypoints) |

# Dữ liệu và định dạng dữ liệu

* Tập dữ liệu hiện tại:
* Bộ dữ liệu sân cầu lông: 1508 ảnh – kích thước 1920 x 1080 – 22 keypoints
* Bộ dữ liệu sân tennis: 918 ảnh + 398 ảnh = 1316 ảnh – kích thước 1280 x 720 – 14 keypoints
* Định dạng dữ liệu
* Định dạng cho các mô hình YOLO-pose

dataset

|\_\_\_ train

|   |\_\_\_ images

|   |    |\_\_\_image\_train\_1.jpg

|   |    |\_\_\_image\_train\_2.jpg

|   |\_\_\_ labels

|   |    |\_\_\_image\_train\_1.txt --> class\_id(0) x\_min y\_min x\_max y\_max x1 y1 v1(0 or 1 or 2) x2 y2 v2 ...........

|   |    |\_\_\_image\_train\_2.txt

|\_\_\_ val

|   |\_\_\_ images

|   |\_\_\_ labels

|\_\_\_ test

|   |\_\_\_ images

|   |\_\_\_ labels

* Định dạng cho các mô hình hồi quy như Resnet:
* Input là ảnh, đầu ra là
* vector 28 chiều (ứng với 14 keypoints của sân tennis) hoặc vector 44 chiều (ứng với 22 keypoints của ssan cầu lông).
* Hoặc chỉ có 8 chiều (ứng với 4 keypoints góc) nếu muốn thử nghiệm việc dựa vào 4 góc kết hơn transform từ mô hình sân chuẩn
* Chiều vector đầu ra có thể thêm 1 giá trị cho mỗi keypoints nếu muốn thử nghiệm tỷ lệ hình ảnh chứa sân hay không
* Bộ dataset cần có folder chứa các ảnh (các folder train\_images, valid\_images, test\_images) và file .json ứng với từng folder ảnh chứa keypoints của các ảnh
* Convert dataset về định dạng phù hợp. Dưới đây là convert cho huấn luyện ResNet cho detect sân tennis (14 keypoints ~ 28 chiều)

class KeypointsDataset(Dataset):

    def \_\_init\_\_(self, img\_dir, data\_file):

        self.img\_dir = img\_dir

        with open(data\_file, "r") as f:

            self.data = json.load(f)

        self.transforms = transforms.Compose([

            transforms.ToPILImage(),

            transforms.Resize((224, 224)),

            transforms.ToTensor(),

            transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])

        ])   # transforms help in resizing and normalizing images

    def \_\_len\_\_(self):

        return len(self.data)

    def \_\_getitem\_\_(self, idx):

        item = self.data[idx]

        img = cv2.imread(f"{self.img\_dir}/{item['id']}.jpg")

        h,w = img.shape[:2]

        img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB) # BGR to RGB

        img = self.transforms(img)

        kps = np.array(item['kps']).flatten() # Conver 2D array to 1D array

        kps = kps.astype(np.float32)

        kps[0::2] = 224.0/w \* kps[0::2] # Adjust x coordinates

        kps[1::2] = 224.0/h \* kps[1::2] # Adjust y coordinates

        return img, kps

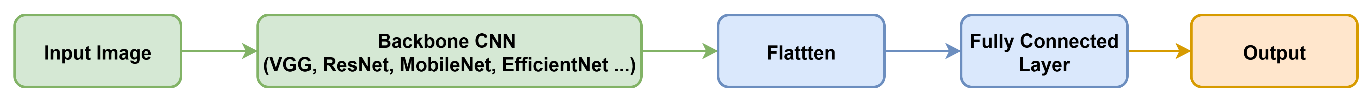
* Định dạng cho các mô hình dựa trên heatmap: sử dụng **Gaussian centered at keypoint** để xây dựng grouth truth heatmap

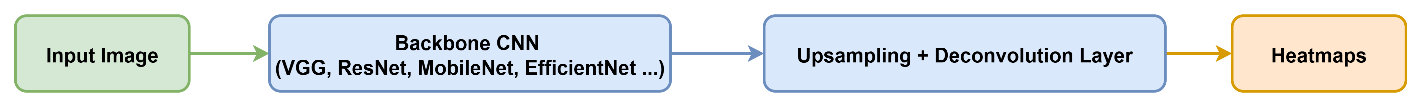
# Hướng giải quyết và các mô hình

A diagram of a model

AI-generated content may be incorrect.

Hình 3.1 Pipeline giải quyết bài toán detect sân

* Có hai hướng tiếp cận chính:
* **Two-stage (Top-down):** Dùng Object Detector (YOLO, Faster R-CNN…) để tìm bounding box của đối tượng. Sau đó, crop ảnh → dùng Keypoint Estimator để dự đoán keypoints. Ưu điểm là độ chính xác cao, nhược điểm là chậm hơn, phụ thuộc vào chất lượng detect ban đầu.
* **Single-stage (Bottom-up):** Dự đoán toàn bộ keypoints trực tiếp trên ảnh. Ưu điểm là nhanh hơn, xử lý nhiều đối tượng cùng lúc; nhược điểm là độ chính xác thấp hơn trong môi trường phức tạp.
* Có hai hướng mô hình tiếp cận chính:
* **Regression-based**: Mô hình trực tiếp dự đoán tọa độ (x, y) cho từng keypoint.
* **Heatmap-based**: Mô hình dự đoán một heatmap cho mỗi keypoint, trong đó giá trị pixel thể hiện xác suất điểm đó là vị trí keypoint. Sau đó lấy max hoặc soft-max để ra tọa độ.
* **Regression-based:** Đầu ra của mô hình là trực tiếp tọa độ (x, y) (và có thể thêm confidence) của từng keypoint. Nếu ảnh có *K keypoints*, mô hình thường trả về vector kích thước *2K (hoặc 3K nếu có confidence/visibility).*
* Cấu trúc mô hình chính:
* Họ mô hình Yolo-pose: Dự đoán bounding box + keypoints trực tiếp trong output vector
* Họ mô hình dựa trên backbone CNN kết hợp FC
* Quá trình huấn luyện:
* Loss function thường dùng: L1 Loss, L2 Loss, Smooth L1 Loss giữa tọa độ dự đoán và ground truth.
* Dữ liệu đầu vào có thể được chuẩn hóa tọa độ (0–1) hoặc ở pixel gốc.
* **Heatmap-based:** Mô hình dự đoán một heatmap 2D cho mỗi keypoint. Nếu bài toán có K keypoints, mô hình sẽ sinh ra K ảnh 2D (heatmap), mỗi ảnh có cùng kích thước với ảnh đầu vào hoặc giảm tỉ lệ (ví dụ 64×48 nếu ảnh gốc 256×192). Mỗi heatmap chỉ quan tâm đến một keypoint cụ thể. Heatmap ~ ma trận xác suất, giá trị cao nhất (peak) là vị trí keypoint.
* Cấu trúc mô hình chính:



A diagram of different types of bars

AI-generated content may be incorrect.

Hình 3.2 Minh họa heatmap-based model (khác là đầu vào có 1 ảnh, đầu ra là K heatmaps)

* Quá trình huấn luyện:
* Loss function thường dùng: MSELoss hoặc KL-Divergence (Kullback–Leibler) giữa heatmap dự đoán và heatmap ground truth.

# Chỉ số đánh giá

* ***PCK (Percentage of Correct Keypoints):*** tỉ lệ keypoint nằm trong bán kính cho phép so với ground truth.

*với True Keypoint If*

* ***OKS (Object Keypoint Similarity):*** tương tự IoU nhưng cho keypoints. ([Link bài viết](https://learnopencv.com/object-keypoint-similarity/))
* ​ ​ = khoảng cách Euclidean giữa keypoint dự đoán và ground truth thứ .
* ​ = visibility của keypoint thứ (0: không có, 1: ẩn, 2: hiển thị).
* = thước đo tỉ lệ đối tượng (ví dụ: ​).
* = hằng số scale cho từng keypoint.
* = hàm chỉ báo (1 nếu điều kiện đúng, 0 nếu sai).
* ***RMSE:*** sai số trung bình Euclidean.

***Chi tiết xây dựng hàm đánh giá:*** <https://github.com/house3173/CourtDetectionOffical/tree/master/evaluate>

* **Format dữ liệu đầu vào** cho đánh giá với 2 file json gồm ground\_truth.json và predited.json

***ground\_truth.json***

[

    {

        "id": "test\_image\_2",

        "bbox": [341, 251, 1247.143, 617.143],

        "area": 769665.386,

        "kps": [

            [ 615, 258 ],

            [ 1312, 258 ],

            [ 345, 860 ],

            [ 1584, 863 ],

            [ 703, 258 ],

            [ 502, 858 ],

            [ 1224, 259 ],

            [ 1425, 861 ],

            [ 675, 345 ],

            [ 1252, 346 ],

            [ 573, 649 ],

            [ 1354, 651 ],

            [ 963, 346 ],

            [ 963, 650 ]

        ]

    },

]

***predicted.json***

[

    {

        "id": "test\_image\_2",

        "kps": [

            [608, 259, 1],  # GIÁ TRỊ THỨ 3 LÀ CONFIDENCE HOẶC ĐỂ MẶC ĐỊNH LÀ 1

            [1288, 259, 1],

            [331, 851, 1],

            [1566, 851, 1],

            [692, 259, 1],

            [485, 851, 1],

            [1203, 259, 1],

            [1411, 851, 1],

            [663, 342, 1],

            [1232, 342, 1],

            [559, 641, 1],

            [1336, 641, 1],

            [947, 342, 1],

            [947, 641, 1 ]

        ]

    },

]

* **PCK**:
* Tham số được xác định bằng khoảng cách chéo giữa keypoints góc trên trái và keypoints góc dưới phải của sân ground\_truth.
* Giá trị hiện tại được đặt 0.02, đảm bảo hiệu khoảng cách Euclidean giữa predicted keypoints và ground\_truth keypoints trong khoảng 20-30 pixels.

A group of people on a tennis court

AI-generated content may be incorrect.

Hình 4.1 Minh họa True Keypoints (màu xanh lá) với PCK và False Keypoints (màu đỏ)

* **OKS**:
* Tham số được xác định dựa trên diện tích của đối tượng trong ground\_truth theo công thức: , trong đó giá trị 0.53 được recommend bởi COCO dataset
* Hằng số scale cho từng keypoint ban đầu được xác định trên độ lệch chuẩn của các anotator, sau đó điều chỉnh cho phù hợp với bài toán. Keypoints càng dễ nhận diện, độ sai lệch thấp thì giá trị phải càng nhỏ. Đối với 14 keypoints sân tennis hiện tại, như sau:

[0.075, 0.075, 0.060, 0.060, 0.0725, 0.0725, 0.0575, 0.0575, 0.07, 0.07, 0.055, 0.055, 0.055, 0.04]

A tennis court with people on it

AI-generated content may be incorrect.

Hình 4.2 Minh họa kết quả OKS giữa predicted keypoints và ground\_truth keypoints

* **RMSE**: Có 3 chỉ số gồm RMSE trung bình trên tất cả keypoints, RMSE trung bình trên tất cả ảnh, RMSE được chuẩn hóa với độ dài đường chéo sân.

# Thử nghiệm các mô hình và đánh giá

*Thử nghiệm ưu tiên theo thứ tự từ trên xuống:*

* Thử nghiệm thay đổi các backbone CNN trong các mô hình regression-based
* ResNet 18 – 34 – 50 – 101 …
* MobileNet v2 – v3 – small …
* EfficientNet
* VGG 16 – 19
* ….
* Thử nghiệm đầu ra của regression-based là 3K (x – y – confidence)
* Thử nghiệm chỉ xây mô hình detect 4 keypoints góc sân và kết hợp homopgraphy để suy ra các keypoints khác
* Thử nghiệm các mô hình Yolo-pose: ưu tiên các mô hình nhanh như Yolov8-pose, Yolov11n-pose
* Thử nghiệm thay đổi backbone trong các mô hình heatmap-based. Thử nghiệm các cấu trúc heatmap-based khác

*Đánh giá:*

* Đánh giá trên các chỉ số PCK, OKS, RMSE (3), Inference Time
* Plot các trường hợp và các keypoints hay sai để tìm cách điều chỉnh mô hình, xử lý

*Hậu xử lý:*

* Tìm các hậu xử lý kết quả để điều chỉnh keypoints chính xác hơn như Refine keypoints, Homography …
* Loại bỏ các detect nhầm khi không có sân, đặc biệt trong các mô hình regression-based

*Kết quả mục tiêu 🡪* **PCK: 75% – 85% và OKS: 80% – 90%**