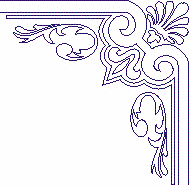
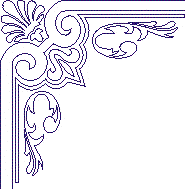
**Duong vien 2** **TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

|  |  |  |
| --- | --- | --- |
|  |  | **Duong vien** |

**Duong vien**

****

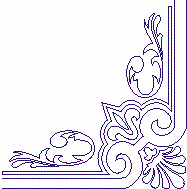
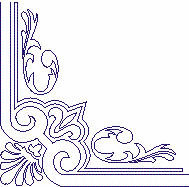
**DEEP LEARNING**

**XÂY DỰNG MÔ HÌNH NHẬN DIỆN BIỂN BÁO GIAO THÔNG**

Người thực hiện: Nguyễn Văn Huy MSSV: 20170080

Nguyễn Ngọc Thịnh MSSV: 20173386

Nguyễn Thanh Nhã MSSV: 20170103

**Duong vien 2**

***Hà Nội, 2020***

**Tóm tắt kết quả**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Câu hỏi** | **Trả lời** | **Chi tiết (nếu có)** |
| 1 | Ý tưởng đề tài mới không? Nhiều nhóm khác làm chưa? | Ý tưởng không còn mới. Duy nhất nhóm em làm | Bài toán nhận dạng biển báo đã xuất hiện từ lâu. Cuộc thi này là lúc thể hiện. |
| 2 | Có thu thập và gán nhãn dữ liệu không? Mô tả thông tin chi tiết nếu có. | Không. | Dữ liệu đã được Zalo gán nhãn sẵn theo COCO format. Tập public gồm 5086 ảnh với hơn 11000 box |
| 3 | Dùng tập dữ liệu benchmark có sẵn không? | Có | Dữ liệu được lấy từ Zalo AI Challenge (<https://challenge.zalo.ai/portal/traffic-sign-detection>).  Kết quả của nhóm được submit lên đây. |
| 4 | Có tiền xử lý dữ liệu không? | Có | Chia đôi ảnh theo chiều ngang, sau đó scale ảnh theo các options (1, 1.25, 1.5) |
| 5 | Giải quyết vấn đề bằng nhiều phương pháp khác nhau không? | Có | 1.Preprocess: Cắt đôi ảnh, scale ảnh lên để detect tốt hơn các object bé  2.Model: Thay các backbone khác nhau |
| 6 | Có đề xuất mô hình mới hay chỉnh sửa/cải tiến mô hình có sẵn nào để giải quyết vấn đề không? | Có | Data có nhiều nhiễu, một số box được đánh label nhiều lần. Team top 1 có đề xuất sử dụng những box được đánh label 1 lần để training, và label lại cho những box bị đánh label nhiều lần.  Quy định của btc là không được đánh nhãn bằng tay. |
| 7 | Huấn luyện mô hình mới hay chỉ dùng mô hình sẵn có cùng bộ weight sẵn có (pre-trained model)? | Có cả 2 | Mô hình mới chỉ đạt 0.39504 khi submiss lên cuộc thi trong khi train với weight sẵn có (tập COCO) thì đạt 0.49502 |
| 8 | Nếu huấn luyện lại thì huấn luyện từ đầu (from scratch) hay dùng pre-trained weights? | Huấn luyện dùng pretrain weights | Nhóm em đã thử 1 số lần huấn luyện với pretrain / huấn luyện từ đầu thì thấy huấn luyện với pretrain cho kết quả tốt hơn |
| 9 | Có cài đặt mô hình không? Hay chỉ clone từ github? | Clone từ github và chỉnh sửa | Repo chính là CenterNet (), ngoài ra nhóm có code thêm để phù hợp với bài toán (chủ yếu là về dữ liệu) |
| 10 | Nhóm tự lập trình bao nhiêu phần trăm code trong tổng số những phần code chính (không tính những code phụ trợ) | 30% | Lập trình xử lý dữ liệu là chính |
| 11 | Có deploy mô hình lên đâu không (ví dụ web, mobile app…)? | Không | Nhóm đã thử deploy lên docker tuy nhiên thất bại |
| 12 | Có backend, CSDL gì không? | Không |  |
| 13 | Công việc từng thành viên |  | Nguyễn Văn Huy: trưởng nhóm, Lập trình mô hình, huấn luyện mô hình, deploy mô hình  Nguyễn Ngọc Thịnh: Nghiên cứu các mô hình, làm báo cáo  Nguyễn Thanh Nhã: Nghiên cứu và triển khai các phương pháp tiền xử lý dữ liệu, làm slide |
| 14 | Phần trăm đóng góp của các thành viên | Nguyễn Văn Huy: 33%  Nguyễn Ngọc Thịnh: 33%  Nguyễn Thanh Nhã: 33% |  |
| 15 | Tự chấm điểm (thang điểm 10) | Nguyễn Văn Huy: 10  Nguyễn Ngọc Thịnh: 10  Nguyễn Thanh Nhã: 10 |  |
| 16 | Các vấn đề khác nếu có |  |  |

**MỤC LỤC**

[DANH MỤC BẢNG 5](#_Toc60227391)

[TÓM TẮT 6](#_Toc60227392)

[1. Giới thiệu đề tài, dữ liệu. 7](#_Toc60227393)

[1.1. Đề tài 7](#_Toc60227394)

[1.2. Dữ liệu 7](#_Toc60227395)

[2. Mô hình sử dụng: CenterNet 8](#_Toc60227396)

[2.1. Hướng tiếp cận 8](#_Toc60227397)

[2.2. Nguyên lý cơ bản và các hàm tối ưu 9](#_Toc60227398)

[2.2.1. Nguyên lý cơ bản của tìm điểm đặc trưng (keypoint estimation) 9](#_Toc60227399)

[2.2.2. Ước lượng độ lệch của keypoint 12](#_Toc60227400)

[2.2.3. Dự đoán kích thước của vật 13](#_Toc60227401)

[2.2.4. Hàm tối ưu tổng 13](#_Toc60227402)

[2.3. Các backbone sử dụng 13](#_Toc60227403)

[2.4. Các điểm yếu của mạng 14](#_Toc60227404)

[3. Xây dựng mô hình và đánh giá kết quả 15](#_Toc60227405)

[3.1. Tiền xử lý dữ liệu 15](#_Toc60227406)

[3.2. Quá trình training và đánh giá kết quả 15](#_Toc60227407)

[3.3. Cải tiến mô hình 16](#_Toc60227408)

[KẾT LUẬN 17](#_Toc60227409)

[TÀI LIỆU THAM KHẢO 18](#_Toc60227410)

# DANH MỤC BẢNG

[Figure 1: Demo 7](#_Toc60240512)

[Figure 2: Compare model 8](#_Toc60240513)

[Figure 3: Nguyên lý hoạt động 9](#_Toc60240514)

[Figure 4: Keypoint heatmap 9](#_Toc60240515)

[Figure 5: Keypoint heatmap example 10](#_Toc60240516)

[Figure 6: Labeling 11](#_Toc60240517)

[Figure 7: Keypoint heatmap 12](#_Toc60240518)

[Figure 8: Results CenterNet 14](#_Toc60240519)

[Figure 9: Demo 16](#_Toc60240520)

# TÓM TẮT

Phát hiện Biển báo Giao thông là một thành phần quan trọng trong các hệ thống tự hành như ô tô tự lái. Mô-đun phát hiện phải nhanh và chính xác vì các quyết định tiếp theo được lập kế hoạch và thực hiện dựa trên các đầu ra của hệ thống này. Với sự bùng nổ của AI, machine learning và đặc biệt là deep learning như hiện nay thì chúng ta càng đến gần hơn với giấc mơ xây dựng ô tô tự lái. Xuất phát từ nhu cầu này và cuộc thi do Zalo tổ chức, nhóm chúng em quyết định nghiên cứu về đề tài nhận diện biển báo giao thông đường bộ.

# Giới thiệu đề tài, dữ liệu.

## Đề tài

Xuất phát từ việc xây dựng xe tự lái và cuộc thi do Zalo tổ chức, nhóm em đã tham gia và sử dụng đề tài làm bài tập lớn môn học Deep learning. Bài toán thuộc dạng object detection, gồm có 2 nhiệm vụ chính:

1: Phát hiện khu vực chứa object trong ảnh.

2: Phân loại được object trong khu vực ấy thuộc class nào.

Link cuộc thi: <https://challenge.zalo.ai/portal/traffic-sign-detection>



Figure 1: Demo

## Dữ liệu

Bộ dữ liệu gồm có 4500 ảnh training với 11000 boxes, 586 ảnh trong public test, đã được đánh nhãn cho 7 class:

1: No entry

2: No parking / waiting

3: No turning

4: Max Speed

5: Other prohibition signs

6: Warning

7: Mandatory

# Mô hình sử dụng: CenterNet

Nhóm sử dụng CenterNet là một mạng object detection đạt được cân bằng giữa tốc độ và độ chính xác tốt, được ra mắt năm 2019.

Bài báo gốc: [Objects as Points - Xingyi Zhou, Dequan Wang, Philipp Krähenbühl](https://arxiv.org/abs/1904.07850)

## Hướng tiếp cận

Vấn đề của các mạng object detection thành công nhất hiện nay là chúng phải thực hiện lần qua tất cả các vị trí có thể có vật và thực hiện phân loại mỗi vị trí đó. Điều đó dẫn đến việc lãng phí tài nguyên tính toán, không hiệu quả và cần thực hiện các bước hậu xử lý (Non-maximum suppression).

Hướng tiếp cận mới của CenterNet là đưa bài toán phát hiện vật (object detection) về bài toán tìm điểm đặc trưng (keypoint estimation), từ đó cũng suy ra kích thước và tính toán được bounding box cho bài toán phát hiện vật. Kiến trúc mạng cũng có thể dễ dàng được sửa lại để output ra vị trí 3D, hướng và tư thế cho các bài toán khác.

Từ hình dưới có thể thấy rõ hướng tiếp cận của CenterNet khá hiệu quả. Nó vượt qua các thuật toán 1 stage phổ biến nhất hiện nay là YOLO v3, RetinaNet trong sự cân bằng giữa tốc độ và độ chính xác. Hơn nữa độ chính xác của CenterNet còn ngang ngửa Faster RCNN - một mạng phát hiện vật 2 stage.

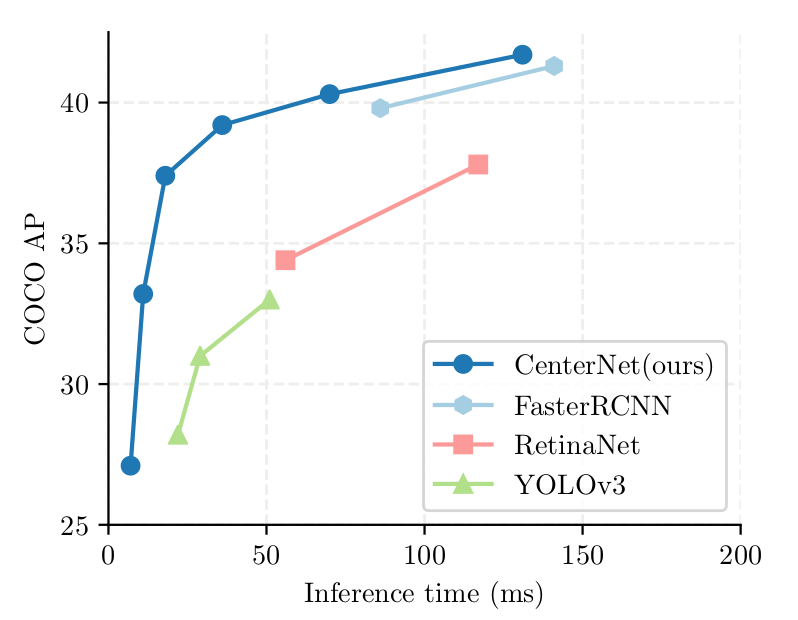


Figure 2: Compare model

## Nguyên lý cơ bản và các hàm tối ưu

Nguyên lý hoạt động của CenterNet được mô tả như hình dưới.

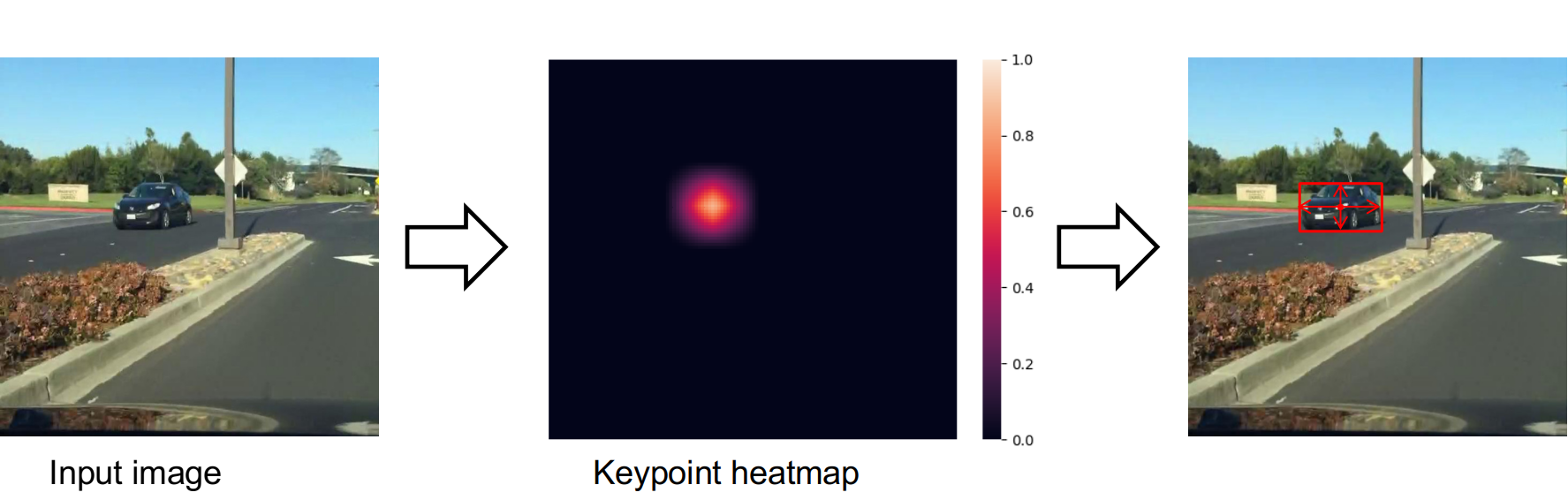


Figure 3: Nguyên lý hoạt động

Sau khi qua mạng backbone, ảnh đầu vào sẽ được biến đổi thành một heatmap (bản đồ nhiệt). Mỗi ô trong bản đồ heatmap này thể hiện xác suất trong ô đó chứa tâm của vật. Tiếp đó CenterNet thực hiện lọc các điểm cực đại trên heatmap để xác định tâm của các vật trên ảnh. Từ đó có thể suy ra được kích thước của vật (với bài toán phát hiện vật) và các đặc điểm khác với các bài toán khác.

### Nguyên lý cơ bản của tìm điểm đặc trưng (keypoint estimation)

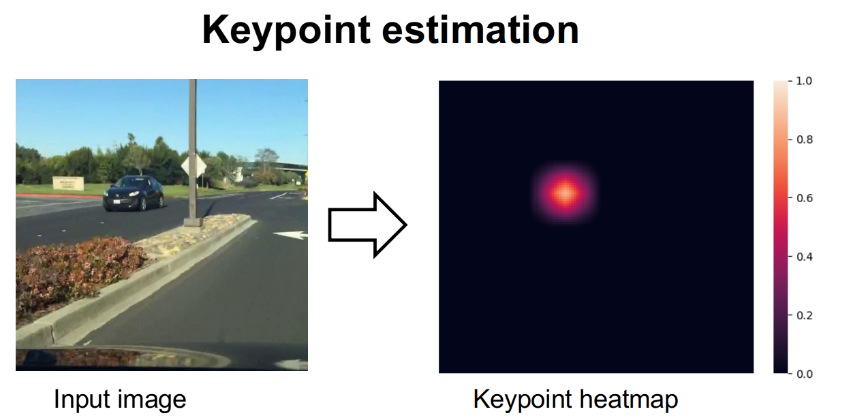


Figure 4: Keypoint heatmap

Sau khi đưa ảnh đầu vào qua mạng backbone (CNN), ta thu được feature map, là một ma trận có kích thước *dài \* rộng \* chiều sâu*. Ứng với mỗi điểm trong feature map này (1 \* 1 \* chiều sâu), ta sẽ thực hiện bước keypoint estimation (tìm điểm đặc trưng). *Stride=4* của CenterNet có nghĩa là feature map có kích thước chiều dài và chiều rộng đều nhỏ hơn chiều dài và chiều rộng của ảnh đầu vào 4 lần. Điều đó cũng có nghĩa mỗi ô trong feature map tương đương với 4 ô trong ảnh đầu vào. Feature map càng nhỏ hơn ảnh đầu vào ở kích thước *chiều dài \* chiều rộng*, ta sẽ cần nhiều anchorbox để bắt hết các vị trí của vật trong ảnh, vì có thể nhiều vật nằm trong cùng một ô của feature map.

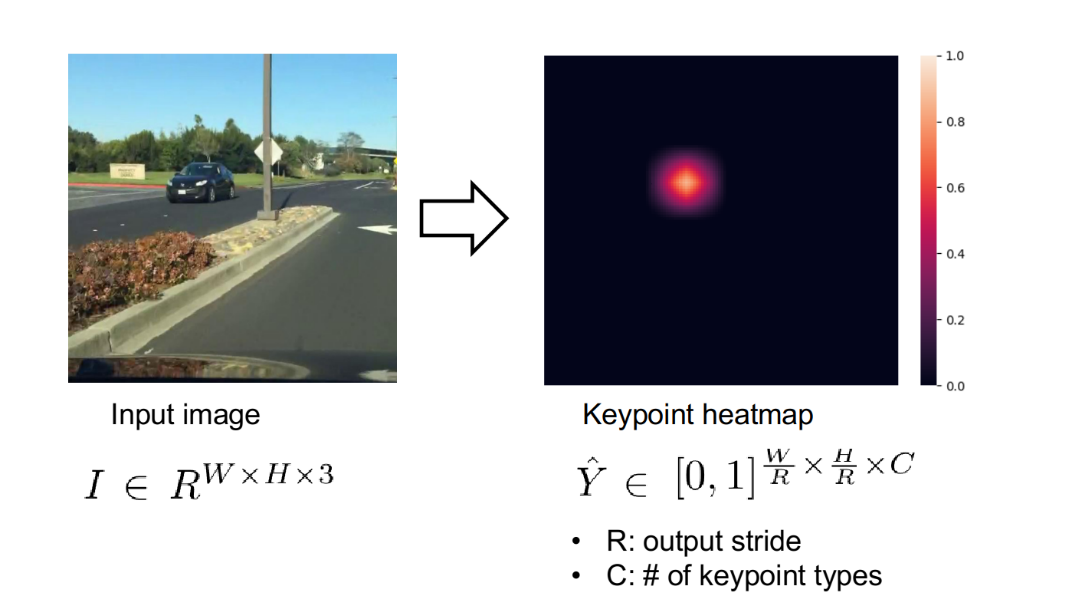


Figure 5: Keypoint heatmap example

Với ảnh đầu vào có chiều rộng W, chiều dài H, chúng ta cần huấn luyện mạng để tạo ra heat map , trong đó R là stride của heatmap và C là số loại keypoint. Trong bài toán nhận dạng tư thế người, C là số khớp, trong bài toán phát hiện vật, C là số class (C = 80 trong tập dữ liệu MS COCO). Trong mỗi ô của heatmap này, ứng với một điểm background (nền).

Ở CenterNet, tác giả sử dụng stride R = 4, do vậy kích thước heatmap sẽ nhỏ hơn ảnh đầu vào 4 lần. Giả sử chúng ta đang làm bài toán phát hiện vật với số class là 80, ảnh đầu vào 512\*512\*3 (3 là số kênh màu trong hệ màu RGB). Như vậy, heatmap sẽ có kích thước 128\*128\*80.

### Huấn luyện

Để huấn luyện mạng keypoint prediction, trước hết chúng ta tạo ground truth heatmap để làm mục tiêu hướng đến.

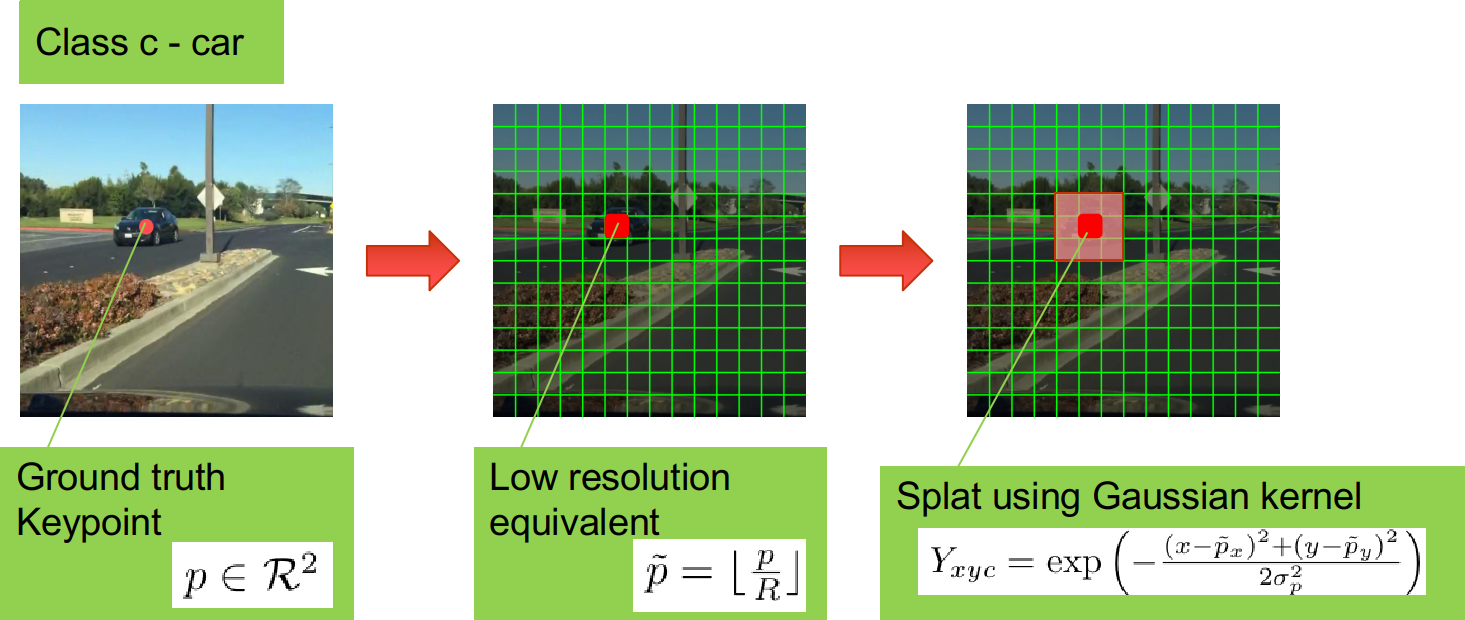


Figure 6: Labeling

Ground truth heatmap được tạo riêng biệt cho mỗi class trong bài toán object detection. Giả sử ở đây ta tạo ground truth cho class car - xe ô tô.

Từ bounding box của đối tượng xe ô tô trong ảnh, ta dễ dàng tìm được điểm keypoint là tâm của vật . Giả sử ảnh đầu vào có kích thước 512\*512\*3, bounding box của xe ô tô trong ảnh là thì tâm của vật là . Từ đây ta tìm điểm tương ứng với trên heatmap là . Sở dĩ có công thức này vì heatmap nhỏ hơn ảnh đầu vào lần, do vậy ta chỉ cần lấy toạ độ của chia cho và lấy phần nguyên của kết quả. Tiếp đó ta lan rộng điểm ra với phân phối chuẩn , . Với tham số của công thức phân phối Gaussian được tính trên kích thước của vật.

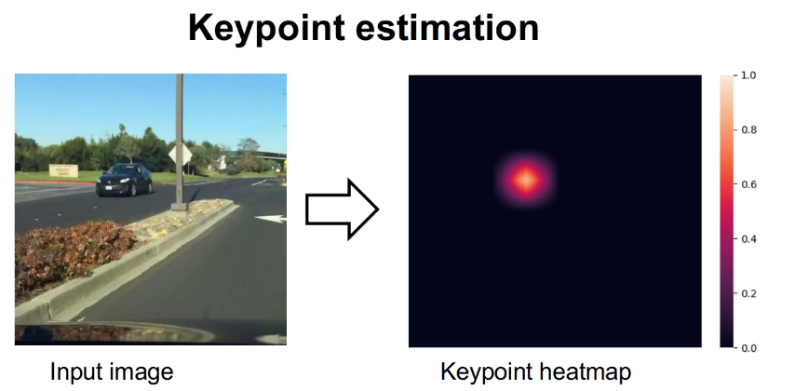
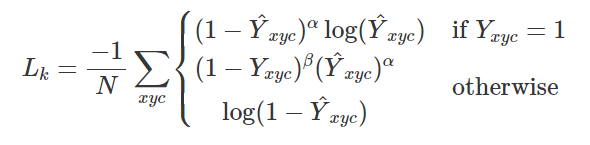


Figure 7: Keypoint heatmap

Hàm tối ưu được sử dụng cho bước này là Focal loss. Focal loss được cải tiến từ cross-entropy nhằm hạn chế sự ảnh hưởng của mất cân bằng class. Mất cân bằng class trong trường hợp này chính là sự mất cân bằng giữa số điểm trên heatmap ứng với tâm vật (thường khá nhỏ) và số điểm là background, giữa các điểm khó phân biệt và các điểm dễ phân biệt. Công thức hàm loss cho keypoint prediction như sau:



### Ước lượng độ lệch của keypoint

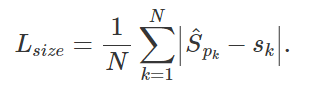
Việc tạo ra ground truth heatmap nhỏ hơn ảnh đầu vào R lần và tính vị trí keypoint trên ground truth heatmap bằng công thức thực chất đã tạo ra sai số (sử dụng phép phần nguyên). Điều này được tác giả xử lý bằng cách thêm bộ ước lượng độ lệch (offset predictor) với công thức . Số 2 ở đây đại diện cho chiều dọc và chiều ngang. CenterNet sử dụng chung một offset predictor cho tất cả các class và huấn luyện offset predictor với hàm loss L1:

Trong đó chính là phần chênh lệch tạo ra khi làm tròn giá trị

### Dự đoán kích thước của vật

Giả sử bounding box của vật có dạng . Có thể dễ dàng tính ra điểm tâm của vật . Song song với việc dùng keypoint predictor để tìm ra điểm tâm này, CenterNet tìm ra kích thước của vật tương ứng với mỗi vật *k.* Để giảm chi phí tính toán, CenterNet dùng một bộ ước lượng kích thước cho tất cả các class:

. Số 2 ở đây đại diện cho chiều dài và chiều rộng. Việc huấn luyện bộ ước lượng kích thước lại tiếp tục dựa vào hàm tối ưu L1:



### Hàm tối ưu tổng

Hàm tối ưu tổng được tạo ra dựa trên các hàm tối ưu được dùng ở trên với các trọng số và .



## Các backbone sử dụng

CenterNet sử dụng tập dữ liệu MS COCO để đánh giá độ chính xác cho phát hiện vật.

Có 3 backbone chính được giới thiệu: Hourglass đạt độ chính xác tốt nhất, Deep Layer Aggregation DLA đạt được cân bằng về tốc độ và độ chính xác và ResNet (18) cho tốc độ cao nhất.

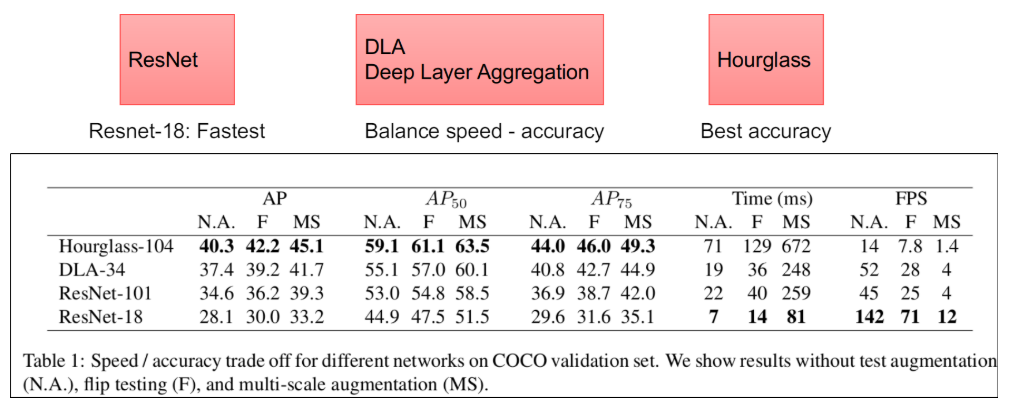


Figure 8: Results CenterNet

## Các điểm yếu của mạng

CenterNet là một hướng đi mới, và cũng có những hạn chế của nó. Sau đây là một vài nhược điểm nhóm em đã tìm hiểu được.

### Center point collision: Hoạt động kém với các vật có tâm ở gần nhau

Phân tích của tác giả: Trong tập huấn luyện của MS COCO, là một tập dữ liệu khá lớn và đa dạng có 614 cặp vật có tâm gần nhau, bị trùng trong heatmap khi ở stride 4, tuy nhiên có 86001 vật tất cả => Chỉ < 0.1% số vật thể không thể phát hiện vì lý do này. Tỷ lệ lỗi này khá nhỏ so với RCNN, Fast RCNN khi region proposal không tốt và các thuật toán dựa trên anchorbox khi không đặt đủ số anchorbox.

Nhược điểm này có quan trọng hay không còn phụ thuộc vào bài toán, vấn đề mà mình cần giải quyết nữa.

### Thời gian huấn luyện khá lâu

Trong paper Training-Time-Friendly Network for Real-Time Object Detection, các tác giả có viết rằng thời gian huấn luyện mạng CenterNet cho MS COCO là khoảng 140 epoch trong khi thời gian huấn luyện cho các mạng kiểu SSD, YOLOv3 chỉ rơi vào khoảng 12 epoch.

Đây cũng không phải là yếu tố quá quan trọng trong đa số các trường hợp.

# Xây dựng mô hình và đánh giá kết quả

## Tiền xử lý dữ liệu

Ảnh đầu vào có size 1622 x 626. Nhóm em đã tiến hành cắt đôi ảnh thành 2 phần theo chiều ngang, thu được 2 bức ảnh (điều này cũng làm giảm bớt bộ nhớ của mỗi bức ảnh). Sau đó, với mỗi bức ảnh bọn em tiến hành scale 1 và 1.5 lần để thu được 1 ảnh size lớn hơn, giúp cho việc detect các object nhỏ được dễ dàng hơn.

Đối với ban đầu có N ảnh, thì sau việc tiền xử lý bọn em có N \* 2 \* 2 ảnh.

Đối với các ảnh trước khi đưa vào model, cần có w và h phải chia hết cho 32 đối với DLA34 (128 đối với Hourglass). Vì vậy ảnh sẽ phải được resize về size chia hết cho 32 bằng phép so bit (inp\_w = w | 31, inp\_h = h | 31).

## Quá trình huấn luyện và đánh giá kết quả

Bộ dữ liệu được zalo đóng gói ở dạng COCO format, tương thích với repo Centernet. Vì vậy không cần bước chuyển đổi dữ liệu.Quá trình huấn luyện mô hình như đã được giới thiệu trong phần giới thiệu về CenterNet. Nhóm em tiến hành huấn luyện mô hình với backbone là DLA34 và Hourglass, và đạt được mô hình tốt nhất với Houghglass.

Quá trình xuất ra kết quả cho tập test của mô hình gồm có 4 giai đoạn:

1: Tiền xử lý ảnh đầu vào.

2: Đưa ảnh đã được xử lý vào model, lúc này chúng ta có kết quả của, heatmap, size object và offset.

3: Gộp các kết quả trên theo các class.

4: Merge kết quả, chọn ra box đúng nhất trong các box trùng nhau.

5: Merge 2 ảnh được cắt đôi ở bước tiền xử lý lại với nhau.

Dưới đây là một ví dụ về kết quả test của mô hình:



Figure 9: Demo

Mô hình đạt score 0.49502 trên tập public của ban tổ chức, tuy nhiên mô hình đó nhóm em có sử dụng pretrain model (DLA 34 train trên COCO dataset). Kết quả không sử dụng pretrain chỉ đạt 0.39504

Vì một số vấn đề trên môi trường với docker nên nhóm em đã không submit được trên tập private.

## Cải tiến mô hình

Mô hình có thể được cải thiện score bằng một số phương pháp tiền xử lý ảnh khác hoặc sử dụng mạng khác (YOLO,..).

Ví dụ như team đạt top 1 đã sử dụng những box được đánh 1 label duy nhất để predict cho những box bị đánh cho nhiều label, qua đó góp phần làm giảm nhiễu cho mô hình.

# KẾT LUẬN

Nhóm em đã tìm hiểu được về CenterNet và áp dụng được vào bài toán cụ thể: nhận diện biển báo giao thông. Kết quả còn chưa được tốt lắm và có thể cải thiện được hơn.

# TÀI LIỆU THAM KHẢO

1. Source code: <https://github.com/nguyenvanhuybk99/zalo_traffic_sign>
2. CenterNet , <https://github.com/xingyizhou/CenterNet>
3. *Objects as Points -* Xingyi Zhou, Dequan Wang, Philipp Krähenbühl, <https://arxiv.org/abs/1904.07850>
4. *Tìm hiểu mô hình CenterNet –* VietAnhDev, <https://aicurious.io/posts/2020-04-23-tim-hieu-ve-centernet/>