ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG ĐIỆN - ĐIỆN TỬ

BÁO CÁO ĐỒ ÁN 1

Đề tài

NGHIÊN CỨU PHẦN MỀM XÁC ĐỊNH KHOẢNG CÁCH VÀ VỊ TRÍ CỦA ĐỐI TƯỢNG QUA CAMERA ĐƠN

Giảng viên hướng dẫn : Ts. Hán Trọng Thanh

Sinh viên thực hiện : Tạ Đức Mạnh

MSSV : 20213995

Hà Nội, 01/2024

Lời nói đầu

Trong thời đại công nghệ số hiện nay, các công nghệ về robot và các xe tự hành đang ngày càng phát triển mạnh mẽ và được ứng dụng trong rất nhiều lĩnh vực của đời sống. Chính vì vậy, việc ứng dụng các hệ thống camera thông minh vào các hệ thống xe tự hành và robot đang trở thành một xu hướng phổ biến và thiết yếu trong việc đảm bảo an toàn và giảm thiểu các sai sót khi vận hành.

Sự phát triển mạnh mẽ của công nghệ máy tính, trí tuệ nhân tạo và xử lý hình ảnh đã mở ra những cơ hội mới trong việc nâng cao độ chính xác và hiệu quả của các hệ thống giám sát và phân tích hình ảnh. Việc ứng dụng các hệ thống camera thông minh không chỉ hỗ trợ trong việc giám sát an ninh mà còn có tiềm năng lớn trong các lĩnh vực khác như giao thông, y tế, công nghiệp và đời sống hàng ngày.

Nhận thấy tiềm năng phát triển của đề tài, đặc biệt là trong lĩnh vực xe tự lái và robot, em quyết định chọn đề tài này, kết hợp với việc sử dụng các thuật toán về học máy và học sâu để nâng cao độ chính xác và khả năng hoạt động trong nhiều điều kiện môi trường khác nhau.

Em xin chân thành cảm ơn Ts. Hán Trọng Thanh đã tận tình chỉ bảo và giúp đỡ em trong quá trình thực hiện đồ án.

Do thời gian và kiến thức còn hạn chế, đồ án của em không thể tránh khỏi những sai sót, em xin chân thành cảm ơn các ý kiến đóng góp, chỉ bảo của các thầy cô.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

Từ viết tắt	Tên đầy đủ
ANN	Artificial Neural Network
CNN	Convolutional Neural Network

DANH MỤC BẢNG BIỂU

CHƯƠNG 1 TỔNG QUAN ĐỀ TÀI

1.1 Tính cấp thiết của đề tài

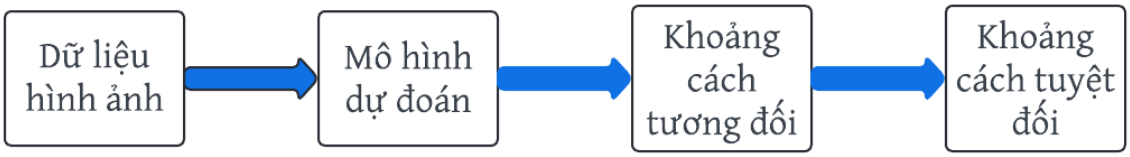
Ngày nay, với sự phát triển của khoa học kỹ thuật, các loại xe tự hành đã và đang ngày một được sử dụng một cách thông dụng trong đời sống con người. Xe tự có thể tăng cường an toàn giao thông bằng cách giảm thiểu tai nạn do lỗi của con người như say rượu, mệt mỏi, mất tập trung. Xe tự lái còn tiết kiệm thời gian và năng lượng khi chọn các tuyến đường tối ưu, giảm tiêu thụ nhiên liệu và điều phối giao thông hiệu quả. Điều này cũng mang lại sự tiện nghi và thoải mái cho người lái, giúp họ có thể làm việc, nghỉ ngơi hoặc giải trí trong quá trình di chuyển. Hơn nữa, xe tự lái còn giúp giảm chi phí vận hành trong các ngành như vận tải và giao hàng do giảm bớt nhu cầu về tài xế chuyên nghiệp.

Tuy nhiên hiện nay xe tự hành vẫn đang chỉ dừng lại ở việc hỗ trợ con người chứ chưa thể hoàn toàn thay thế con người điều khiển phương tiện khi tham gia giao thông. Khi một người lái xe, các quyết định được đưa ra dựa theo thông tin nhận được từ mắt. Xe tự hành cũng cần có một hệ thống tương tự để nhận biết sự thay đổi của môi trường xung quanh. Do đó, để nâng cao hiệu quả của xe tự hành và tránh xảy ra va chạm trong quá trình tham gia giao thông, việc sử dụng một hệ thống camera tích hợp phần mềm dự đoán khoảng cách nhằm đảo bảo xe tự hành luôn giữ đúng khoảng cách an toàn đối với các phương tiện phía trước là điều vô cùng cần thiết, góp phần quan trọng trong việc giảm thiểu tai nạn khi sử dụng xe tự hành.

Vì vậy, em chọn đề tài ‘Nghiên cứu phần mềm xác định khoảng cách và vị trí của đối tượng qua camera đơn’ để tìm hiểu và nghiên cứu phương án để giải quyết vấn đề trên.

1.2 Mục đích nghiên cứu

Mục tiêu của đề tài là nghiên cứu và phát triển hệ thống có khả năng ước lượng khoảng cách từ vật thể đến camera thông qua việc sử dụng một camera hoặc một hình ảnh duy nhất.



Hình 1.1: Mô hình tổng quan của đề tài

1.3 Kết luận

Như vậy chương 1 đã đưa ra tổng quan về đề tài bao gồm tính cấp thiết của đề tài và mục đích nghiên cứu. Chương 2 sẽ đề cập về cơ sở lý thuyết sẽ được sử dụng trong đồ án này.

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

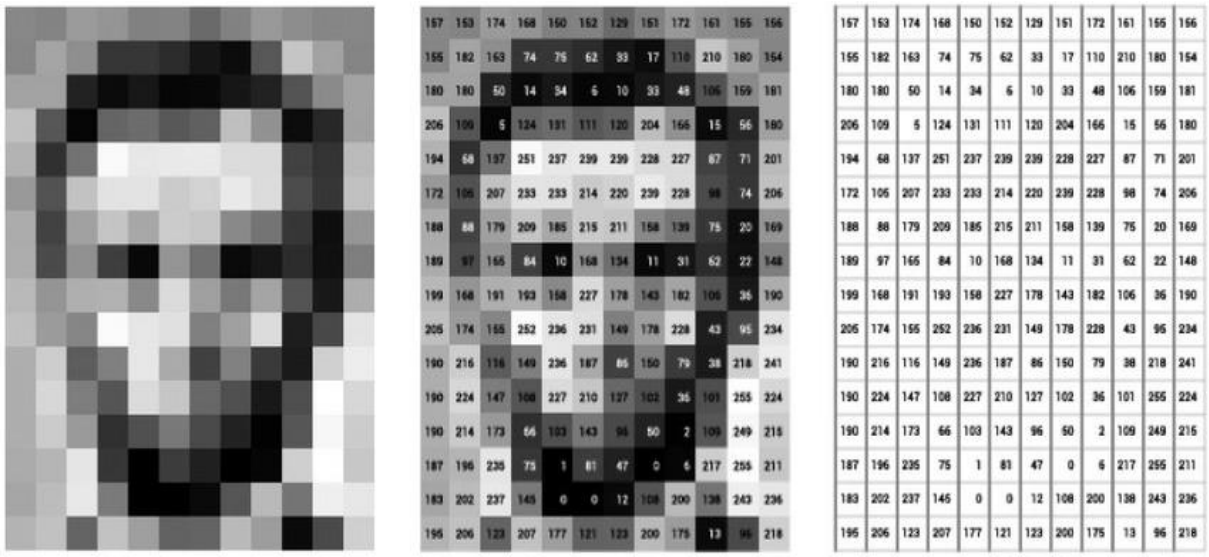
2.1 Tổng quan về xử lý ảnh

2.1.1 Các khái niệm cơ bản về xử lý ảnh

2.1.1.1 Dữ liệu hình ảnh

Một hình ảnh được định nghĩa là một hàm hai chiều F (x, y), trong đó x và y là các tọa độ không gian, và biên độ của F tại bất kỳ cặp tọa độ (x, y) nào được gọi là cường độ của hình ảnh tại điểm đó. Khi các giá trị x, y và biên độ của F là hữu hạn, ta gọi nó là hình ảnh số hóa (Digital image). Nói cách khác, một hình ảnh có thể được xác định bởi một mảng hai chiều, sắp xếp cụ thể theo hàng và cột.

Hình ảnh số hóa bao gồm một số lượng hữu hạn các phần tử, mỗi phần tử có một giá trị cụ thể tại một vị trí cụ thể. Các phần tử này được gọi là phần tử hình ảnh (pixel). Hình ảnh được biểu thị bằng kích thước (chiều cao và chiều rộng) dựa trên số lượng pixel.



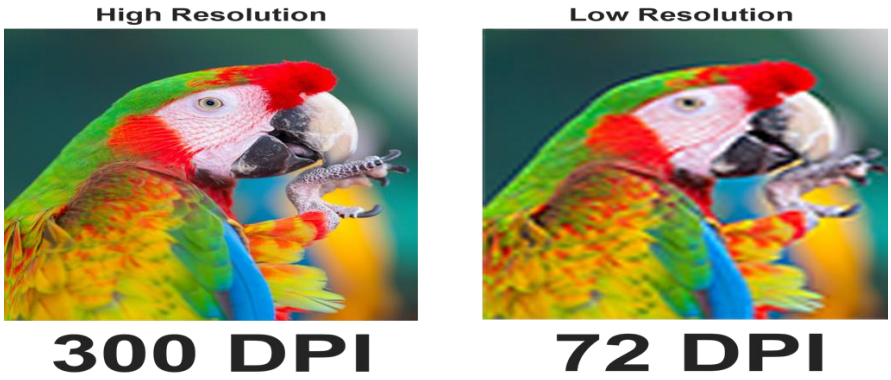
Hình 2.1: Hình ảnh dưới dạng ma trận các pixel *[1]*

Các loại hình ảnh số:

- Ảnh nhị phân: Chỉ chứa những pixel có giá trị 0 hoặc 1 trong đó giá trị 0 chỉ màu đen và giá 1 chỉ màu trắng. Hình ảnh này còn được gọi là ảnh đơn sắc.
- Ảnh đen – trắng
- Ảnh màu 8 bit: Đây là định dạng hình ảnh nổi tiếng nhất, có 256 sắc thái màu khác nhau và thường được gọi là hình ảnh thang độ xám. Trong định dạng này 0 chỉ màu đen, 255 chỉ màu trắng và 127 chỉ màu xám
- Ảnh màu 16 bit: Đây là định dạng ảnh màu có 65.536 màu sắc khác nhau, nó còn được gọi là định dạng màu cao. Ở định dạng này, sự phân bố màu sắc không giống như hình ảnh Thang độ xám mà thường được chia thành 3 kênh màu Đỏ, Xanh lục và Xanh lam.

2.1.1.2 Độ phân giải của hình ảnh

Độ phân giải của hình ảnh chỉ lượng thông tin được chứa đựng trong một tập tin hình ảnh hiển thị trên màn hình.Hiểu một các đơn giản đó là số lượng điểm ảnh chứa trên một màn hình hiển thị. Độ phân giải của hình ảnh thương được tính bằng số pixel trên một đơn vị khoảng cách (dpi – dot per inch).



Hình 2.2: Sự khác biệt giữa hình ảnh có độ phân giải cao và hình ảnh có độ phân giải thấp

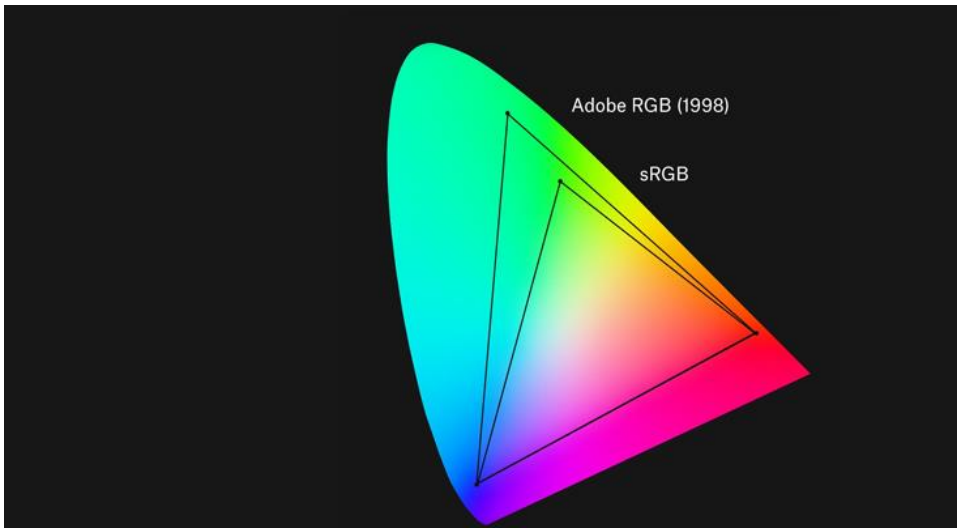
2.1.1.3 Không gian màu

Không gian màu là mô hình toán học dùng để mô tả các màu sắc trong thực tế dưới dạng số học. Không có mô hình nào có thể biểu diễn đầy đủ mọi khía

cạnh của màu. Do đó phải sử dụng những mô hình khác nhau để mô tả những tính chất được nhận biết khác nhau của màu.

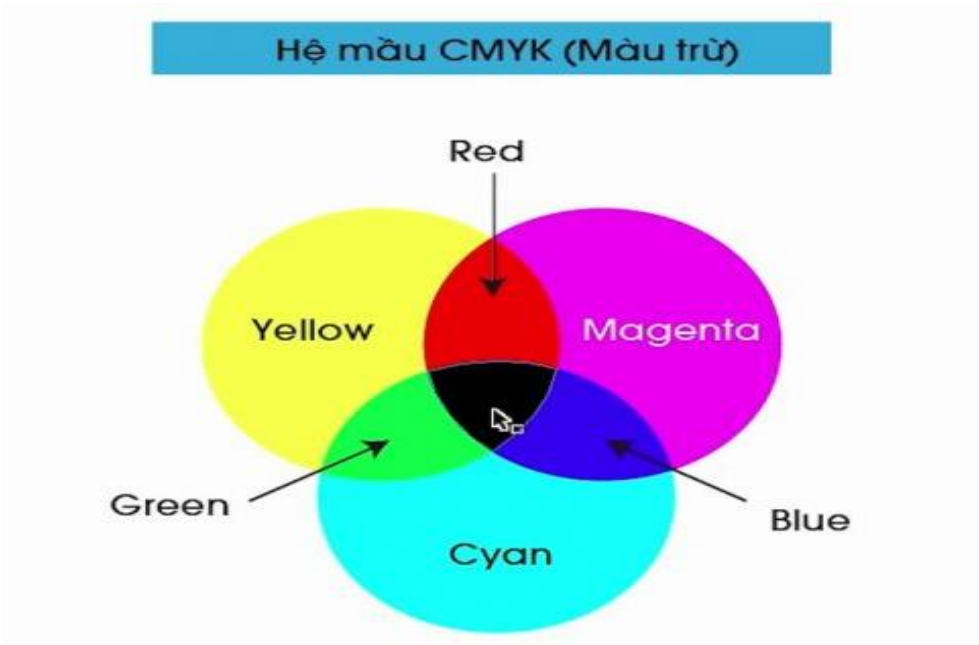
Các không gian màu thông dụng:

- Không gian màu RGB: Đây là không gian màu được sử dụng phổ biến hiện nay, sử dụng ba màu cơ bản là Red (đỏ), Green (lục), Blue (lam) để biểu diễn màu sắc. Mỗi màu được mã hoá 8 bit, ứng với một pixel ảnh màu chiếm 24 bit. Như vậy số lượng màu tối đa đạt được là 256 x 256 x 256 = 16.777.216 màu.



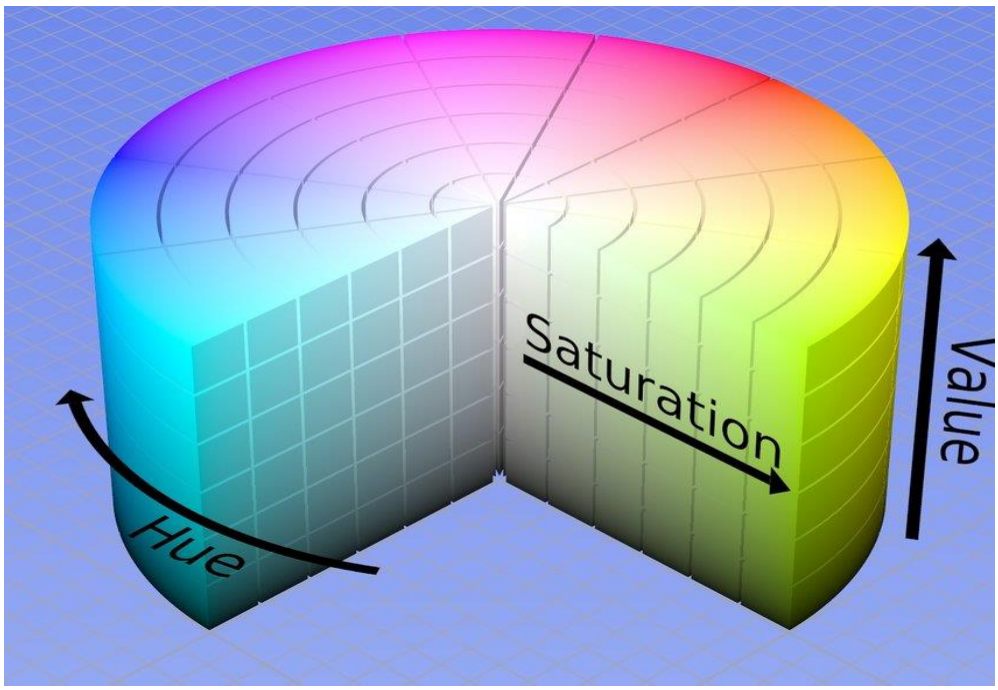
Hình 2.3: Không gian màu RGB [2]

- Không gian màu CMYK: Mô hình màu dựa trên cơ sở trộn chất của các màu Cyan (xanh lơ), Magenta (hồng sẫm), Yellow (vàng) và Key (đen). Nó hoạt động dựa trên nguyên lý hấp thụ ánh sáng, màu mà mắt người nhìn thấy là phần tử của ánh sáng không bị hấp thụ. Trong CMYK, hồng sẫm cộng với màu vàng cho màu đỏ, xanh lơ với vàng sinh ra xanh lá cây. Hồng sẫm kết hợp cùng xanh lơ tạo ra xanh lam, tổ hợp xanh lơ – hồng sẫm – vàng sinh ra màu đen.



Hình 2.4: Không gian màu CMYK [2]

- Không gian màu HSV (HSB): Đây là không gian màu mô tả màu sắc một cách tự nhiên hơn, dựa trên ba thuộc tính: Hue, Saturation và Bright (hoặc Value). Trong đó, Hue là bước sóng của ánh sáng, Saturation là độ bão hòa và Bright (Value) là cường độ hay độ chói ánh sáng.



Hình 2.5: Không gian màu HSV [2]

2.1.1.4 Xử lý ảnh là gì ?

Xử lý ảnh là quá trình chuyển đổi một hình ảnh sang dạng kỹ thuật số và thực hiện các thao tác nhất định để nhận được một số thông tin hữu ích từ hình ảnh đó. Hệ thống xử lý hình ảnh là tín hiệu 2D khi áp dụng một số phương pháp xử lý tín hiệu đã xác định trước.

Các loại xử lý hình ảnh chính bao gồm:

- Nhận diện hoặc phân biệt các đối tượng trong hình ảnh
- Làm sắc nét và phục hồi
- Nhận dạng mẫu
- Truy xuất dữ liệu hình ảnh

2.1.2 Các bước xử lý hình ảnh cơ bản

Các bước xử lý hình ảnh bao gồm:

- Thu thập hình ảnh
- Tăng cường hình ảnh
- Phục hồi hình ảnh
- Xử lý hình ảnh màu
- Xử lý đa phân giải
- Nén hình ảnh
- Xử lý hình thái
- Phân đoạn
- Trình bày và mô tả
- Nhận dạng đối tượng trong hình ảnh

2.1.3 Quá trình tiền xử lý ảnh

Tiền xử lý ảnh là quá trình chuẩn bị dữ liệu ảnh trước khi áp dụng các thuật toán xử lý ảnh hoặc học máy. Các bước tiền xử lý ảnh nhằm cải thiện chất lượng

ảnh và làm cho dữ liệu phù hợp hơn cho các bước xử lý tiếp theo. Các bước này thường bao gồm:

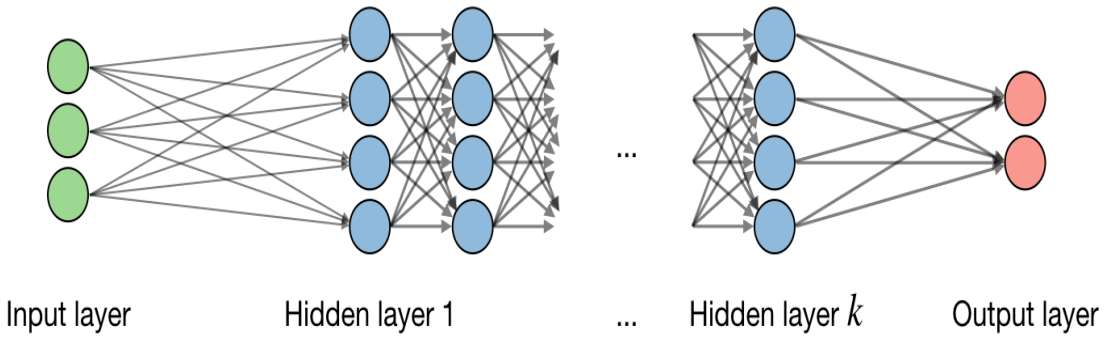
- Chuyển đổi định dạng hình ảnh
- Thay đổi kích thước hình ảnh
- Loại bỏ nhiễu, làm mịn ảnh
- Phân đoạn hình ảnh
- Trích xuất các đặc trưng của đối tượng trong ảnh

Tiền xử lý ảnh đóng vai trò quan trọng trong việc cải thiện chất lượng hình ảnh, đảm bảo tính nhất quán của dữ liệu, đảm bảo rằng tất cả hình ảnh đầu vào có cùng định dạng và chất lượng. Bằng cách chuẩn hoá kích thước và hình dạng, tiền xử lý ảnh giúp giảm độ phức tạp khi tính toán, tăng hiệu quả và độ chính xác của các thuật toán.

2.2 Tổng quan về mạng neuron nhân tạo (ANN)

Được lấy cảm hứng từ sự hoạt động của các neuron thần kinh của não bộ con người, mạng neuron nhân tạo (Artifitial Neural Network) là một lớp các mô hình được cấu tạo từ các layers.

2.2.1 Mô hình cơ bản



Hình 2.6: Mô hình cơ bản của một mạng neuron nhân tạo [3]

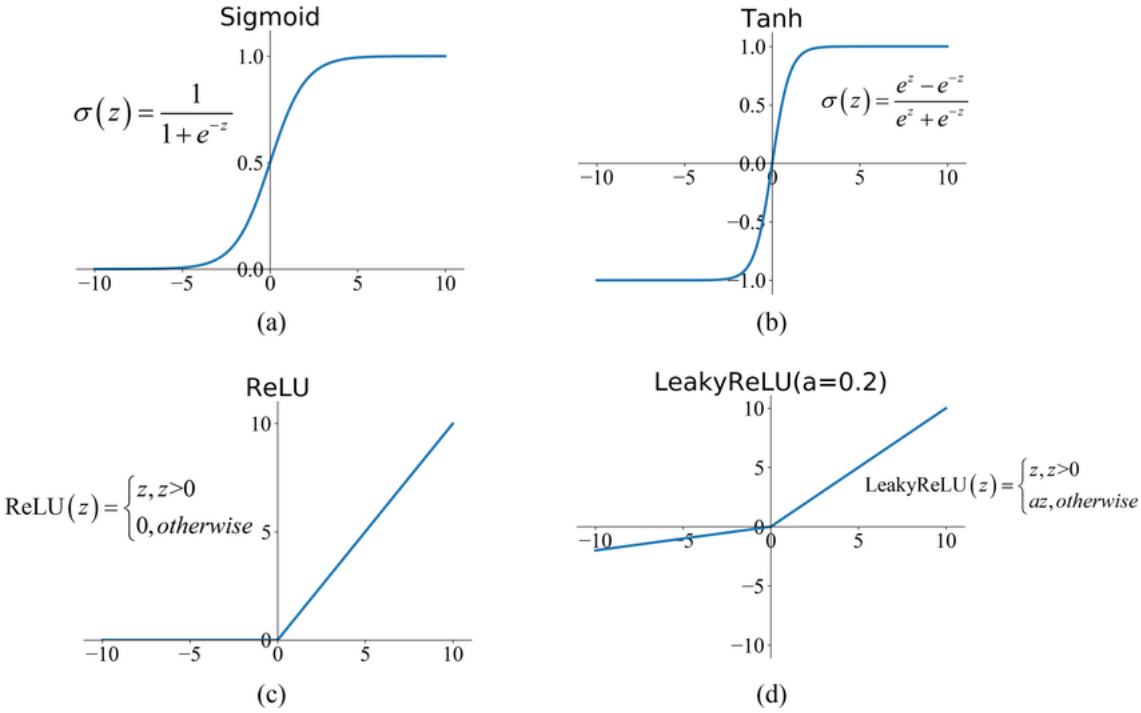
Một mô hình mạng neuron nhân tạo bao gồm 3 layer chính là input layer, hidden layer và output layer, các hình tròn được gọi là node. Mỗi mô hình mạng neuron nhân tạo luôn có một input layer và output layer, có thể có hoặc không

có hidden layer. Mỗi một node trong hidden layer và output layer đều liên kết với các node ở layer trước đó với trọng số (w - weight) riêng. Mỗi node cũng sẽ có hệ số bias (b) riêng.

Quá trình tính toán của một mạng neuron bao gồm hai bước là tính tổng tuyến tính và áp dụng các hàm kích hoạt (Activation function):

$$z_j^{[i]} = W_j^{[i]^T} x + b_j^{[i]} \quad (1)$$

Quá trình dữ liệu được đưa vào mô hình và thu được đầu ra được gọi là quá trình lan truyền thẳng (Feed forward) và lan truyền ngược (Back propagation). Đầu ra của mỗi của mỗi layer sẽ được đưa vào một hàm kích hoạt để tạo ra độ phức tạp phi tuyến cho mô hình trước khi được sử dụng ở các layer tiếp theo. Các hàm kích hoạt thông dụng được sủ dụng có thể kể đến: hàm sigmoid, hàm tanh, hàm ReLU (Rectified Linear Unit), …



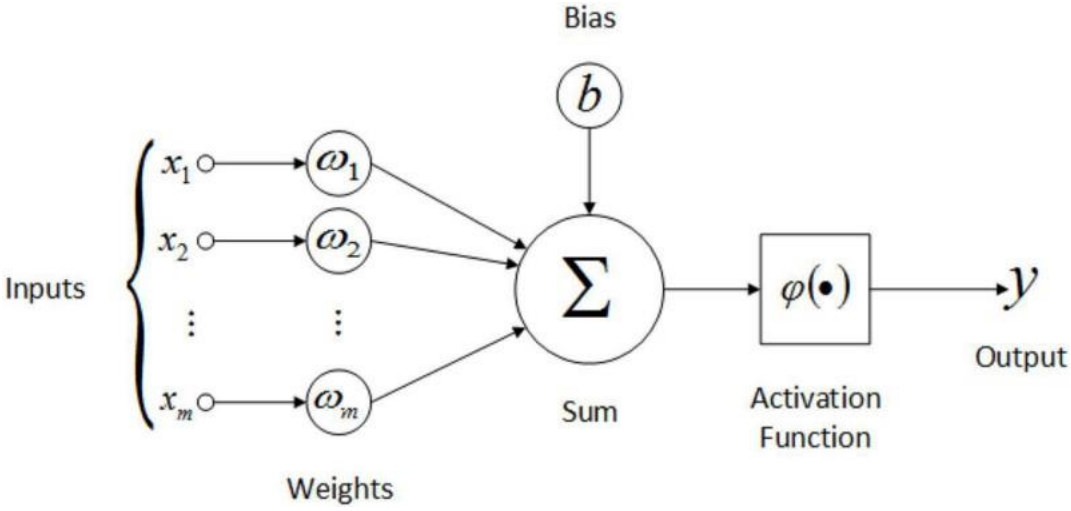
Hình 2.7: Một số hàm kích hoạt thông dụng [3]

2.2.2 Quá trình lan truyền thẳng và lan truyền ngược

Quá trình làn truyền thẳng và lan truyền ngược là hai quá trình đóng vai trò quan trọng trong việc huấn luyện bất kì mạng neuron nào. Hai quá trình này

giúp mô hình học các biểu diễn phức tạp của dữ liệu, tạo nên một hệ thống mạng nơ-ron mạnh mẽ và hiệu quả, mang lại những thành tựu đáng kể trong nhiều lĩnh vực ứng dụng, bao gồm cả thị giác máy tính, xử lý ngôn ngữ tự nhiên, và nhiều bài toán phức tạp khác.

Khi dữ liệu được đưa vào mạng neuron, dữ liệu này sẽ được lan truyền tuần tự qua các hidden layer của mạng neuron đến lớp output cuối cùng. Mỗi lớp tính toán và chuyển tiếp thông tin đến lớp tiếp theo bằng cách sử dụng hàm kích hoạt và các trọng số đã riêng của từng node. Điều này cho phép mạng neuron học các đặc trưng phức tạp và biểu diễn thông tin của dữ liệu đầu vào một cách hiệu quả.



Hình 2.8: Quá trình Feedforward [4]

Để giúp tăng độ chính xác của một mô hình mạng neuron nhân tạo, người ta sử dụng phương thức lan truyền ngược để cập nhật trọng số cho giữa node của mạng bằng cách sử dụng output thực tế và output mong muốn. Sử dụng chain rule, đạo hàm của hàm mất mát (Loss function) với trọng số w được tính theo công thức:

$$\frac{\partial \mathcal{L}(x,y)}{\partial w} = \frac{\partial \mathcal{L}(z,y)}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w} \tag{2}$$

Từ đó trọng số w được cập nhật theo:

$$w \longleftarrow w - \alpha \frac{\partial \mathcal{L}(x,y)}{\partial w} \tag{3}$$

2.2.3 Hàm mất mát và các phương pháp tối ưu

2.2.3.1 Hàm mất mát

Hàm mất mát (Loss function) là một hàm số được dùng để đo lường sự sai khác giữa đầu ra của mô hình so với đầu ra mong muốn, hàm mất mát trả lời cho câu hỏi mô hình đang làm việc như thế nào. Tối ưu hoá hàm mất mát để có được một mô hình hoàn hảo nhất là việc làm rất quan trong khi triển khai bất kì mạng neuron nào.

Đối với các bài toán hồi quy, các hàm mất mát thường được dùng là hàm MAE (Mean absolute error) và MSE (Mean square error).

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Hình 2.9: Hàm mất mát MAE và MSE [5]

Đối với các bài toán phân loại, hàm mất mát thường được dùng là hàm cross-entropy.

$$\mathcal{L}(z, y) = -[y \log z + (1 - y)\log(1 - z)] \qquad (4)$$

2.2.3.2 Các phương pháp tối ưu thông dụng

Để tối ưu hàm mất mát trong các mô hình học máy (Machine learning) và học sâu (Deep learning), phương pháp được sử dụng thông dụng nhất có thể kể đến Gradient Descent và các biến thể của nó như Stochastic Gradient Descent (SGD), Batch Gradient Descent ,…

Gradient Descent

Gradient descent là phương pháp tối ưu cơ bản, phương pháp này cập nhật trọng số bằng cách di chuyển theo hướng ngược chiều của gradient của hàm mất mát. Tuy nhiên, Gradient Descent thường gặp khó khăn khi tìm được điểm cực tiểu toàn cục và dễ rơi vào điểm cực tiểu cục bộ. Công thức cập nhật trọng số của Gradient Descent cho mỗi trọng số w trong mạng neuron:

$$w_{t+1} = w_t - \alpha \mathcal{L}'(w_t) \quad (5)$$

Trong đó:

- w_{t+1}: Trọng số ở lần cập nhật thứ t+1
- w_t: Trọng số ở lần cập nhật thứ t
- α: Tốc độ học (learning rate)

Stochastic Gradient Descent (SGD)

SGD là một biến thể của Gradient Descent, giúp tăng tốc quá trình huấn luyện và tránh rơi vào các điểm cực tiểu cục bộ. Thay vì tính gradient trên toàn bộ tập dữ liệu, SGD tính gradient chỉ trên một điểm dữ liệu ngẫu nhiên trong từng bước cập nhật. Điều này giúp giảm thời gian tính toán nhưng có thể làm nhiễu gradient. Công thức cập nhật trọng số của SGD cho mỗi trọng số w trong mạng nơ-ron:

$$w_{t+1} = w_t - \alpha \mathcal{L}'(w_t, x_i, y_i) \quad (6)$$

Trong đó:

- w_{t+1}: Trọng số ở lần cập nhật thứ t+1
- w_t: Trọng số ở lần cập nhật thứ t
- x_i: Điểm dữ liệu thứ i trong quá trình huấn luyện
- y_i: Nhãn tương ứng của điểm dữ liệu

2.3 Tổng quan về mạng neuron tích chập (CNN)

Mạng neuron tích chập (Convolutional Neural Network) là một mô hình học sâu tiên tiến cho phép xây dựng những hệ thống có độ chính xác cao và thông minh. Nhờ khả năng đó, CNN có rất nhiều ứng dụng, đặc biệt là những bài toán cần nhận dạng vật thể (object) trong ảnh. CNN vô cùng quan trọng để tạo nên những hệ thống nhận diện thông minh với độ chính xác cao trong thời đại công nghệ ngày nay.

Cũng giống như ANN, CNN cũng hoạt động theo phương thức nhận dữ liệu đầu vào và biến đổi biến đổi dữ liệu đầu vào này thông qua các layer. Tuy nhiên do lấy cảm hứng từ xử lý ảnh nên đầu vào của CNN không có dạng vector như ANN mà là một tensor ba chiều.

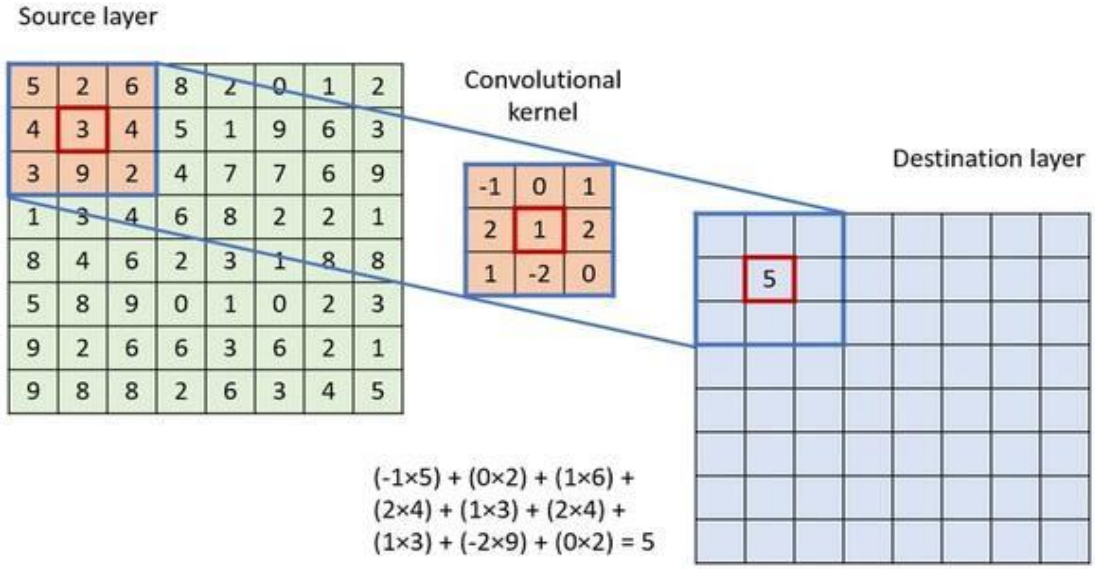
Ngoài ra CNN cũng bổ sung thêm hai khái niệm là tích chập (convolution) và cơ chế chia sẻ trọng số (weight sharing) so với các mạng neuron truyền thống. Nhờ có tích chập, CNN có thể xác định các đặc trưng (features) của hình ảnh một cách hiệu quả. Khi thực hiện phép toán tích chập trên một hình ảnh, ta sử dụng một hoặc nhiều bộ lọc (filter) nhỏ có kích thước hữu hạn. Mỗi bộ lọc này thực hiện việc trượt qua toàn bộ hình ảnh và tính tích chập với từng phần nhỏ của hình ảnh, tạo ra một bản đồ đặc trưng (features map) chứa thông tin về đặc trưng của hình ảnh. Bằng việc sử dụng cơ chế chia sẻ trọng số, các bộ lọc của CNN sẽ được áp dụng cho tất cả các vùng hình ảnh bất kể vị trí của nó. Điều này giúp cho CNN có thể học cách nhận diện các đặc trưng cụ thể trong hình ảnh một cách tổng quát, giúp giảm số lượng tham số cần tối ưu hóa và tránh overfitting.

2.3.1 Các lớp của CNN

Mạng CNN bao gồm những lớp cơ bản: Convolutional layer, Activation layer, Pooling layer, Fully Connected layer.

Convolutional layer

Đây là lớp quan trọng nhất trong CNN, nó có nhiệm vụ thực thi các tính toán, có nhiệm vụ trích xuất các đặc trưng từ ảnh đầu vào. Trong lớp này, các bộ lọc (filter) có kích thước nhỏ sẽ trượt qua toàn bộ ảnh đầu vào để tính tích chập, tạo ra các bản đồ đặc trưng (feature maps) mới. Số lượng bản đồ đặc trưng chính là số lượng bộ lọc sử dụng trong lớp này, và chúng chứa các đặc trưng cụ thể của ảnh. Các yếu tố quan trọng trong lớp convolutional là: padding, stride, filter map và feature map.



Hình 2.10: Ví dụ về lớp convolutional [6]

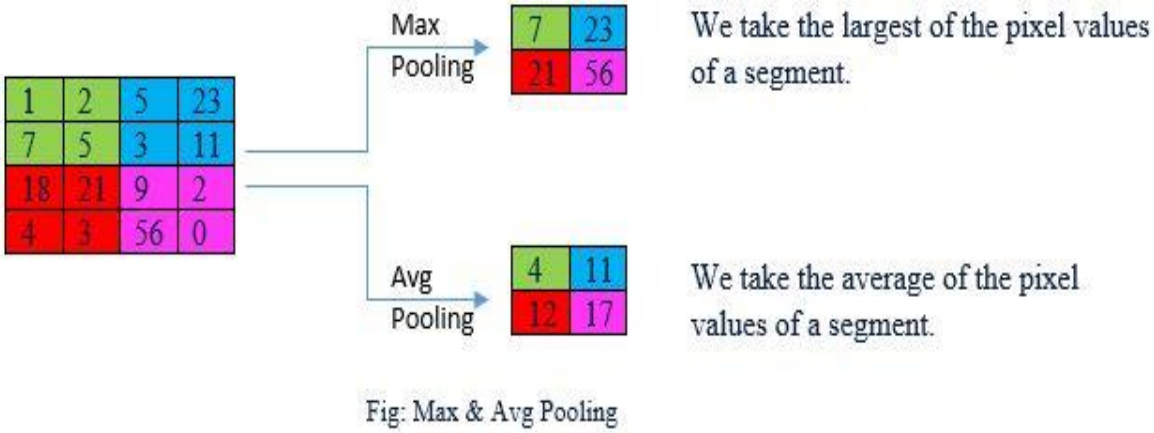
Activation layer

Lớp này mô phỏng lại cá neuron có tỷ lệ truyền xung quanh axon. Lớp này sử dụng các hàm kích hoạt phi tuyến như ReLu, tanh, sigmoid,… để giữ lại các đặc trưng quan trọng và loại bỏ các giá trị âm. Hàm ReLU thường được sử dụng nhiều và thông dụng nhất do hàm này có khả năng hỗ trợ trong tốc độ tính toán nên layer này còn được gọi là ReLU layer.

Pooling layer

Khi ma trận đầu vào có kích thước quá lớn, các lớp Pooling sẽ được đặt vào giữa những lớp Convolutional để làm giảm những parameters và giữ lại

những thông tin quan trọng. Hiện tại, hai loại lớp Pooling được sử dụng phổ biến là Max pooling và Average pooling.



Hình 2.11: Ví dụ lớp Pooling sử dụng max-pooling và average-pooling [6]

Fully Connected layer

Sau khi các lớp tích chập và tổng hợp, các bản đồ đặc trưng đã được giảm kích thước và thu gọn thông tin. Để đưa thông tin bản đồ đặc trưng vào lớp kết nối đầy đủ, ta cần làm phẳng bản đồ đặc trưng thành một véc tơ một chiều, quá trình này gọi là "flatten". Sau khi đã hoàn thành bước "flatten", vectơ dữ liệu một chiều sẽ được đưa vào lớp kết nối đầy đủ, và các lớp này
sẽ thực hiện việc phân loại cuối cùng hoặc dự đoán dựa trên thông tin đã được trích xuất từ ảnh đầu vào.

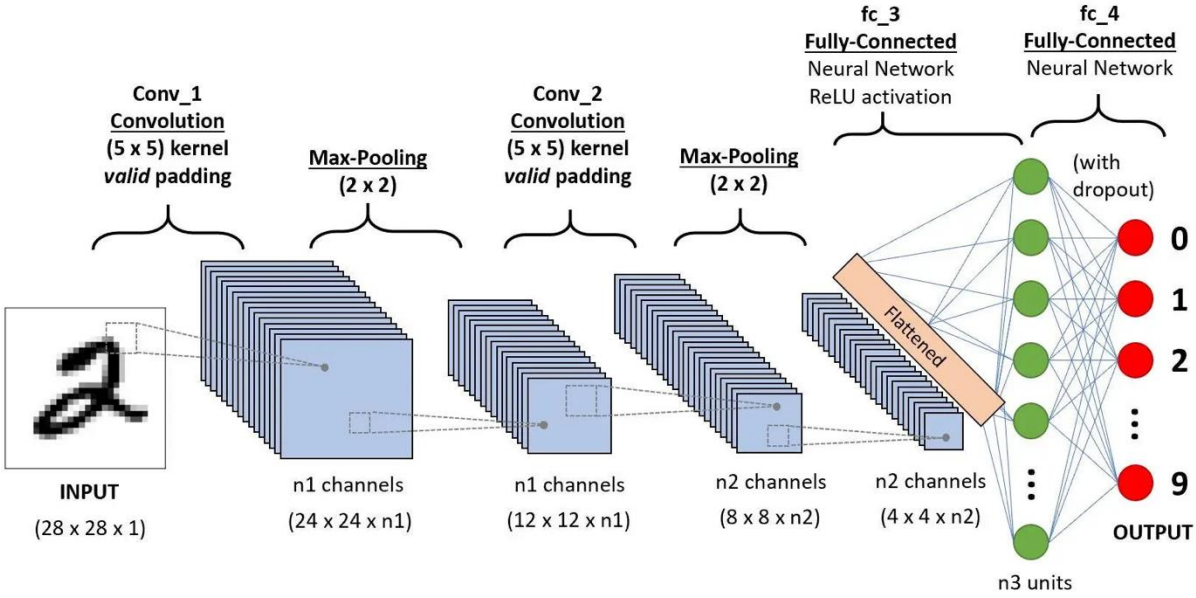
2.3.2 Cấu trúc CNN

Mạng CNN là tập hợp của các lớp Convolutional xếp chồn lên nhau. CNN sử dụng các hàm kích hoạt phi tuyến nhằm kích hoạt các trọng số trong node. Khi đã thông qua hàm, lớp này sẽ thu được các trọng số trong các node và tạo ra nhiều thông tin trừu tượng hơn cho các lớp kế cận.

Mô hình CNN có hai khía cạnh cần phải đặc biệt lưu ý là tính bất biến và tính kết hợp, do đó độ chính xác hoàn toàn có thể bị ảnh hưởng nếu có cùng một đối tượng được chiếu theo nhiều phương diện khác biệt. Với các loại chuyển dịch, co giãn và quay, người ta sẽ sử dụng pooling layer và làm bất biến những

tính chất này. Từ đó, CNN sẽ cho ra kết quả có độ chính xác ứng với từng loại mô hình. Tính kết hợp cục bộ sẽ cho thấy những cấp độ biểu diễn, dữ liệu từ thấp đến cao với mức trừu tượng thông qua Convolution từ filter. Mạng CNN có những lớp liên kết nhau dựa vào cơ chế Convolution.

Các lớp tiếp theo sẽ là kết quả từ những lớp trước đó, vì vậy mà bạn sẽ có những liên kết cục bộ phù hợp nhất. Trong quá trình huấn luyện mạng, mạng nơ ron này sẽ tự học hỏi những giá trị thông qua filter layer dựa theo cách thức mà bạn thực hiện.

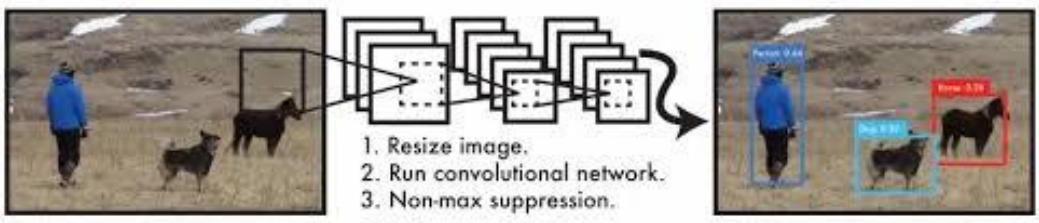


Hình 2.12: Cấu trúc cơ bản của một mô hình CNN [7]

2.4 Mô hình YOLO về xác định vật thể trong ảnh

YOLO trong object detection có nghĩa là “You only look once” , chỉ cần nhìn một là đã có thể phát hiện ra vật thể. Về độ chính xác, YOLO có thể không phải là thuật toán toán nhất trong việc phát hiện và nhận diện vật thể. Tuy nhiên YOLO có tốc độ gần như real time trong khi độ chính xác vẫn giữ được ở mức cao và không quá thua kém so với các mô hình thuộc top đầu. YOLO là thuật toán object detection nên mục tiêu của mô hình không chỉ là dự báo nhãn cho vật thể như các bài toán classification mà nó còn xác định location của vật thể. Do đó YOLO có thể phát hiện được nhiều vật thể có nhãn khác nhau trong một

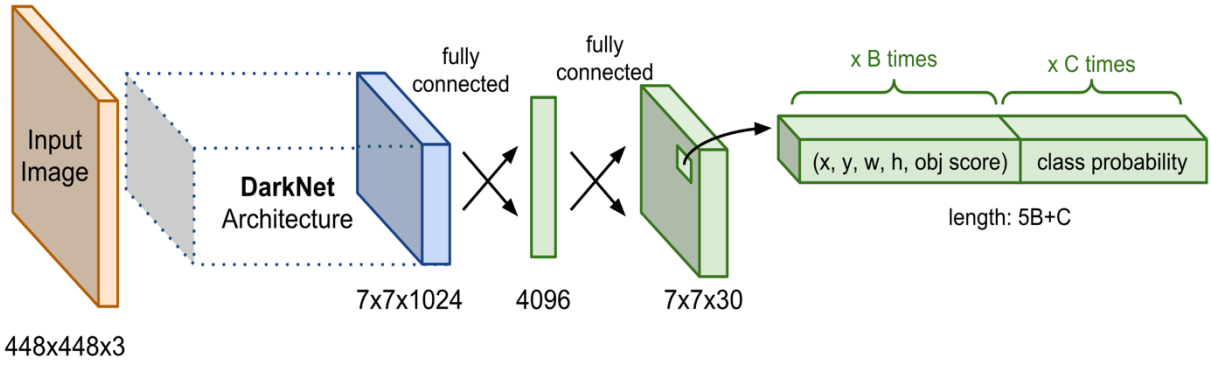
bức ảnh thay vì chỉ phân loại duy nhất một nhãn cho một bức ảnh.Sở dĩ YOLO có thể phát hiện được nhiều vật thể trên một bức ảnh như vậy là vì thuật toán có những cơ chế rất đặc biệt.



Hình 2.13: Hệ thống phát hiện vật thể bằng YOLO [8]

2.4.1 Cấu trúc mô hình YOLO

Kiến trúc YOLO bao gồm các base network là các mạng tích chập làm nhiệm vụ trích xuất đặc trưng. Phần phía sau là những Extra Layers được áp dụng để phát hiện vật thể trên feature map của base network.



Hình 2.14: Cấu trúc mô hình YOLO sử dụng DarkNet làm base network [9]

Đầu ra của base network có kích thước 7x7x1024 sẽ là đầu vào của Extra Layers có tác dụng dự đoán nhãn và toạ độ bounding box của vật thể

Trong YOLOv3 tác giả áp dụng một mạng feature extractor là DarkNet-53. Mạng này gồm 53 convolutional layers kết nối liên tiếp, mỗi layer được theo sau bởi một batch normalization và một activation Leaky Relu. Để giảm kích thước của output sau mỗi convolution layer, tác giả down sample bằng các filter với kích thước là 2. Mẹo này có tác dụng giảm thiểu số lượng tham số cho mô hình.

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

Hình 2.15: Các layer trong DarkNet-53 [9]

Các hình ảnh khi được đưa vào mô hình sẽ được scale về chung một kích thước phù hợp với kích thước đầu vào của mô hình và sau đó được gom lại thành batch đưa vào huấn luyện.

Kích thước của feature map sẽ phụ thuộc vào đầu vào. Đối với đầu vào kích thước 416x416 thì feature map có các kích thước là 13x13, 26x26 và 512x512. Khi đầu vào có kích thước 608x608 sẽ tạo ra feature map 19x19, 38x38, 72x72.

2.4.2 Đầu ra của YOLO

Đầu ra của YOLO là một vector bao gồm các thành phần:

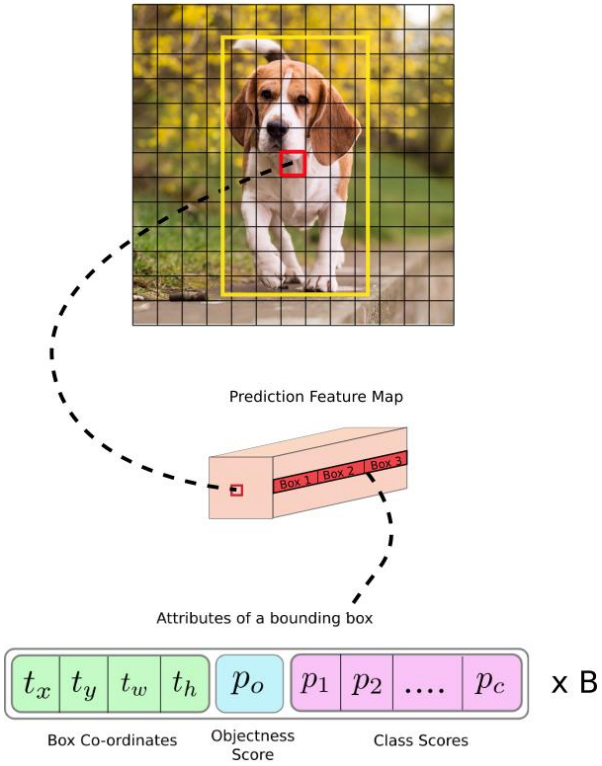
$$y^T = [p_o, (t_x, t_y, t_w, t_h), (p_1, p_2, \dots, p_c)] \quad (7)$$

Trong đó:

- p_o: Xác suất dự báo vật thể xuất hiện trong bounding box.
- (t_x, t_y, t_w, t_h): Giúp xác định bounding box. Trong đó t_x, t_y là toạ độ tâm, t_w, t_h là chiều dài, chiều rộng của bouding box.

- $(p_1, p_2, \ldots, p_c)$: Vector phân phối xác suất dự báo của các classes.

Việc hiểu output khá là quan trọng để chúng ta cấu hình tham số chuẩn xác khi huấn luyện model qua các open source như darknet. Như vậy output sẽ được xác định theo số lượng classes theo công thức (n_class+5). Nếu huấn luyện 80 classes thì bạn sẽ có output là 85. Trường hợp bạn áp dụng 3 anchors/cell thì số lượng tham số output sẽ là (n_class+5) × 3 = 85 × 3 = 255.



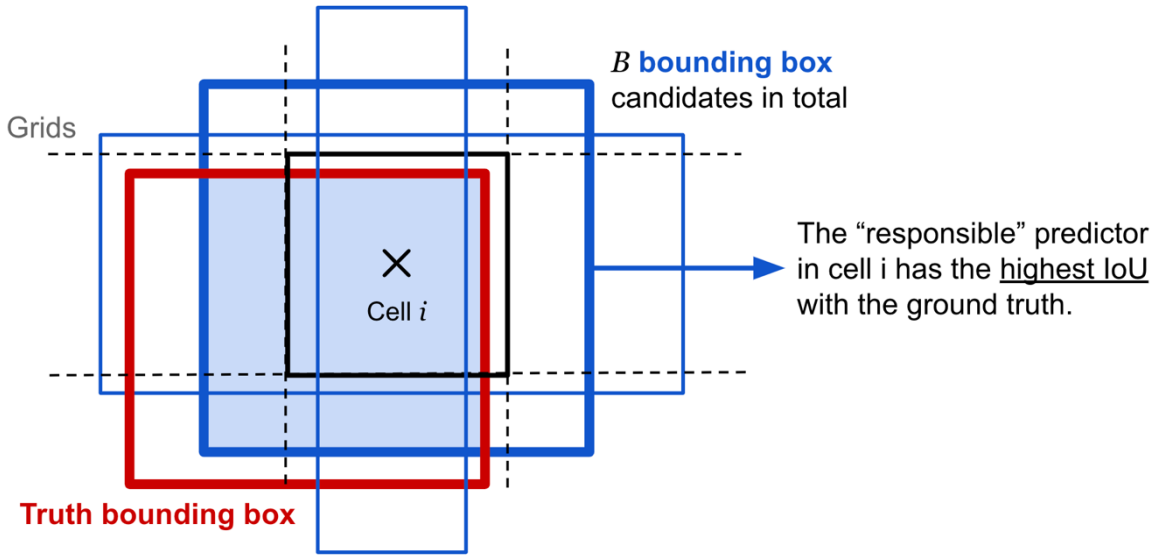
Hình 2.16: Cấu trúc đầu ra của mô hình YOLO [9]

Trên mỗi một cell của feature map chúng ta lựa chọn ra 3 anchor boxes với kích thước khác nhau lần lượt là Box 1, Box 2, Box 3 sao cho tâm của các anchor boxes trùng với cell. Khi đó output của YOLO là một véc tơ concatenate của 3 bounding boxes. Các attributes của một bounding box được mô tả như dòng cuối cùng trong hình

YOLO cũng có khả năng dự đoán trên nhiều feature map. Những feature map ban đầu có kích thước nhỏ giúp dự đoán các vật thể có kích thước lớn. Những feature map sau có kích thước lớn hơn trong khi kích thước của anchor box được giữ cố định sẽ giúp dự đoán các vật thể có kích thước nhỏ hơn.

2.4.3 Anchor box

Để tìm được bounding box cho vật thể, YOLO sẽ cần các anchor box làm cơ sở ước lượng. Những anchor box này sẽ được xác định trước và bao quanh vật thể một cách tương đối chính xác. Sau này thuật toán hồi quy bounding box sẽ tinh chỉnh lại anchor box để tạo ra bounding box dự đoán cho vật thể



Hình 2.17: Xác định anchor box cho một vật thể [9]

Trong một mô hình YOLO, mỗi một vật thể trong hình ảnh huấn luyện được phân bố về một anchor box. Trong trường hợp có từ 2 anchor boxes trở lên cùng bao quanh vật thể thì ta sẽ xác định anchor box mà có IoU với ground truth bounding box là cao nhất. Mỗi một vật thể trong hình ảnh huấn luyện được phân bố về một cell trên feature map mà chứa điểm mid point của vật thể.

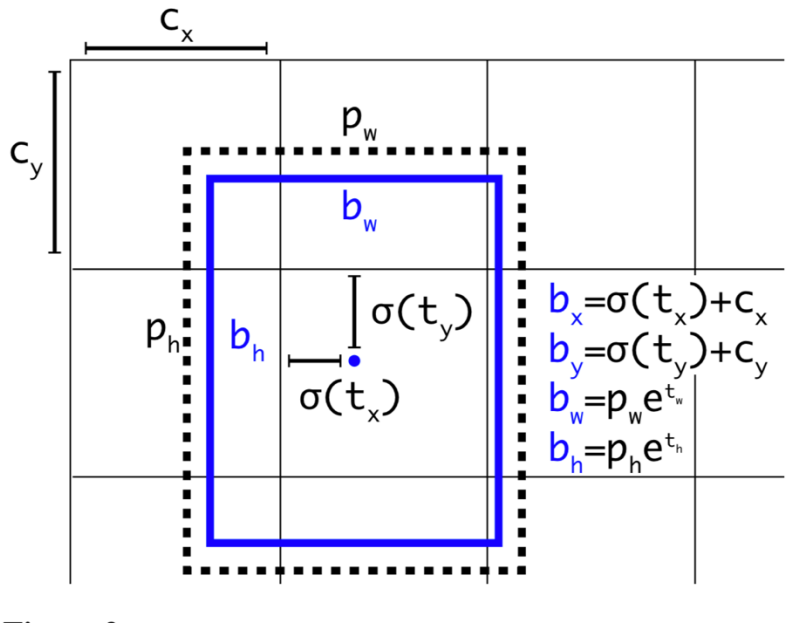
Như vậy khi xác định một vật thể ta sẽ cần xác định 2 thành phần gắn liền với nó là (cell, anchor box). Không chỉ riêng mình cell hoặc chỉ mình anchor box.

2.4.4 Dự đoán bounding box

Thay vì đưa ra các dự đoán tuỳ ý trên bounding box, YOLOv2 và YOLOv3 sử dụng phép biến đổi từ anchor box và cell để dự đoán bounding box.

Với (p_w, p_h) là kích thước của anchor box tại cell nằm trên feature map có các grid với kích thước (c_x, c_y), mô hình dự đoán bốn tham số (t_x, t_y, t_w, t_h) trong đó hai tham số đầu là độ lệch so với góc trên cùng bên trái của cell và hai tham số sau là tỷ lệ so với anchor box. Với các tham số này, một bounding với kích thước (b_w, b_h), tâm tại (b_x, b_y) được tính như sau:

$$b_x = \sigma(t_x) + c_x \quad (8)$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$



Hình 2.18: Dự đoán bounding box [9]

2.4.5 Hàm mất mát

Hàm mất mát của YOLO chia làm hai phần là sai số của bounding box ($\mathcal{L}_{loc}$) và sai số phân phối xác. suất của classes ($\mathcal{L}_{cls}$).

$$\mathcal{L}_{loc} = \lambda_{coor} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2$$

$$+ \left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2$$

$$\mathcal{L}_{cls} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left[\mathbb{1}_{ij}^{obj} + \lambda_{noobj}(1 - \mathbb{1}_{ij}^{obj}) \right] (C_{ij} - \tilde{C}_{ij})$$

$$+ \sum_{i=0}^{S^2} \sum_{c \in C} \mathbb{1}_{i}^{obj} (p_i(c) - \tilde{p}_i(c))^2$$

$$\mathcal{L} = \mathcal{L}_{loc} + \mathcal{L}_{cls}$$

Trong đó:

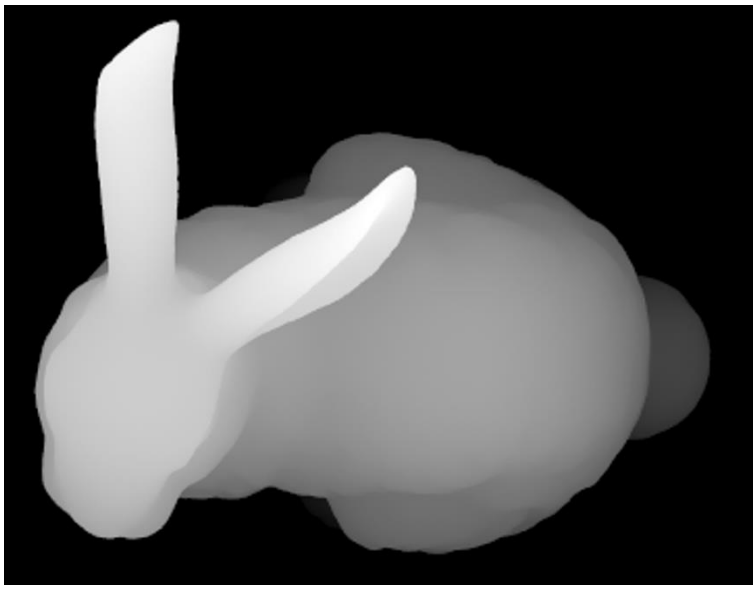
- $\mathbb{1}_{i}^{obj}$: Hàm indicator có giá trị 0, 1 nhằm xác định xem cell i có chứa vật thể hay không. Bằng 1 nếu chứa vật thể, bằng 0 nếu không chứa vật thể.
- $\mathbb{1}_{ij}^{obj}$: Cho biết bounding box thứ j của cell i có phải là bounding box của vật thể được dự đoán hay không.
- C_{ij}: Điểm tin cậy của ô i.
- $\tilde{C}_{ij}$: Điểm tự tin dự đoán.
- C: Tập hợp của tất cả các lớp.
- $p_i(c)$: Xác suất có điều kiện ô i có chứa có chứa một đối tượng c thuộc lớp C hay không.
- $\tilde{p}_i(c)$: Xác suất có điều kiện dự đoán.

Ngoài ra để điều chỉnh hàm mất mát trong trường hợp dự đoán sai bounding box ta thông qua hệ số điều chỉnh λ_{coor}. Nếu muốn giảm nhẹ hàm mất mát trong trường hợp cell không chứa vật thể ta sử dụng hệ số λ_{noobj}. [9]

2.5 Một số phương pháp về dự đoán độ sâu của ảnh (khoảng cách tương đối từ vật tới camera)

2.5.1 Depthmap của ảnh

Depthmap là một hình ảnh hoặc một kênh hình ảnh chứa thông tin về khoảng cách của vật thể đến một điểm tham chiếu (thường là camera). Mỗi một pixel được gán một giá trị biểu diễn cho khoảng cách từ điểm tham chiếu của chính các pixel đó, từ đó tạo ra biểu diễn quang cảnh 3D cho bức ảnh RGB [10].



Hình 2.19: Ví dụ về depthmap của cảnh [10]

Trong ví dụ trên các pixel có màu trắng đại diện cho các phần của vật thể gần với camera nhất, trong khi đó các pixel có màu đen đại diện cho những phần ở xa camera nhất. Các pixel có màu càng sáng (giá trị pixel càng cao) thì phần của vật thể mà pixel đó đại diện cho càng ở gần so với camera và ngược lại.

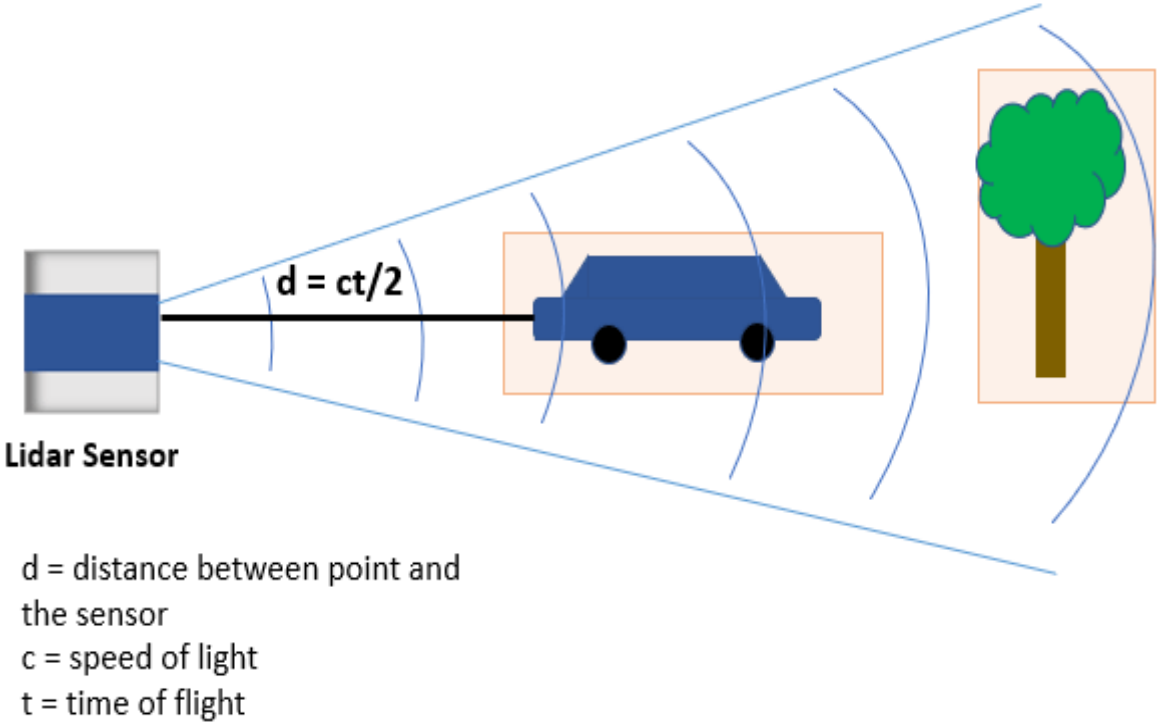
2.5.2 Các phương pháp cổ điển

2.5.2.1 LiDAR (Light Detection and Ranging)

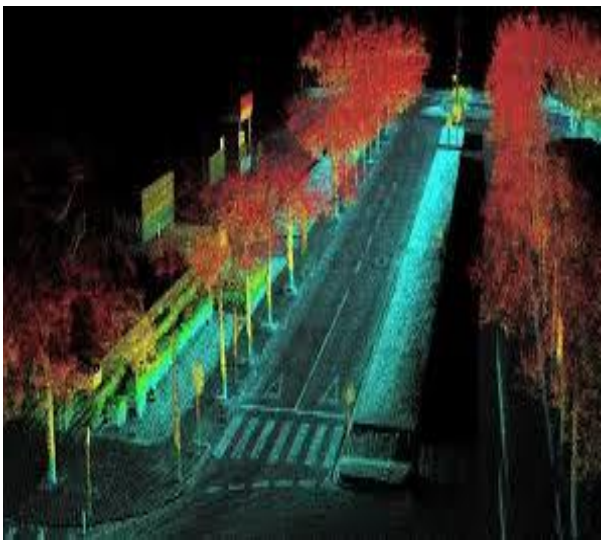
Cảm biến LiDAR (Light Detection and Ranging) là công nghệ sử dụng ánh sáng tia laser để đo khoảng cách và xây dựng bản đồ 3D của vật thể. Mặc dù cảm biến LiDAR đã xuất hiện từ thập niên 60 của thế kỉ trước khi LiDAR được

gắn trên các máy bay quân sự. Thế nhưng phải hơn 20 năm sau LiDAR mới dần trở nên phổ biến hơn nhờ vào sự xuất hiện của hệ thống định vị toàn cầu GPS.

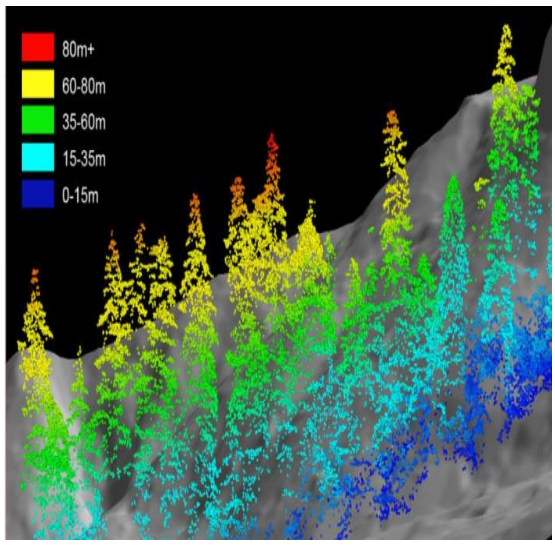
LiDAR hoạt động bằng cách phát xạ ra xung quanh 360 độ, ghi lại thông tin tín hiệu và trả về khoảng cách 3D của những vật thể xung quanh. Với cơ chế hoạt động chiếu hàng triệu điểm khi quay laser liên tục, hệ thống cảm biến LiDAR có thể đo được khoảng cách giữa các vật thể. Dữ liệu truyền về là những mây điểm (Point Cloud), sau đó sẽ được xây dựng bản đồ số 3D.



Hình 2.20: Nguyên lý hoạt động của LiDAR [11]



Hình 2.21: Một số hình ảnh dạng Point Cloud thu được từ LiDAR [12]



Hình 2.22: Một số hình ảnh dạng Point Cloud thu được từ LiDAR [12]

LiDAR có khả năng cung cấp một lượng lớn dữ liệu về khoảng cách và độ sâu và được coi là tiêu chuẩn cho nhiều công ty làm việc trong lĩnh vực xe tự hành tuy nhiên LiDAR vẫn có nhiều nhược điểm khi được lắp đặt trên xe tự hành. LiDAR có chi phí vận hành và lắp đặt rất cao, thậm chí LiDAR đã từng là cảm biến đắt đỏ nhất được lắp đặt trên các phương tiện giao thông. LiDAR cũng không hoạt động quá hiệu quả trong điều kiện thời tiết xấu, giới hạn về tầm nhìn và độ phân giải, độ phức tạp trong tích hợp và bảo trì, yêu cầu cao về tốc độ xử lý dữ liệu, kích thước lớn, ảnh hưởng đến tính thẩm mỹ của xe, và tiêu thụ năng lượng lớn. Ngoài ra, LiDAR cũng chỉ tái tạo về hình ảnh môi trường xung quanh chứ không phải hình ảnh đại diện cho những gì đang diễn ra. Mặc dù vậy LiDAR vẫn được sử dụng rộng rãi do khả năng ứng dụng tốt trong việc xây dựng bản đồ 3D.

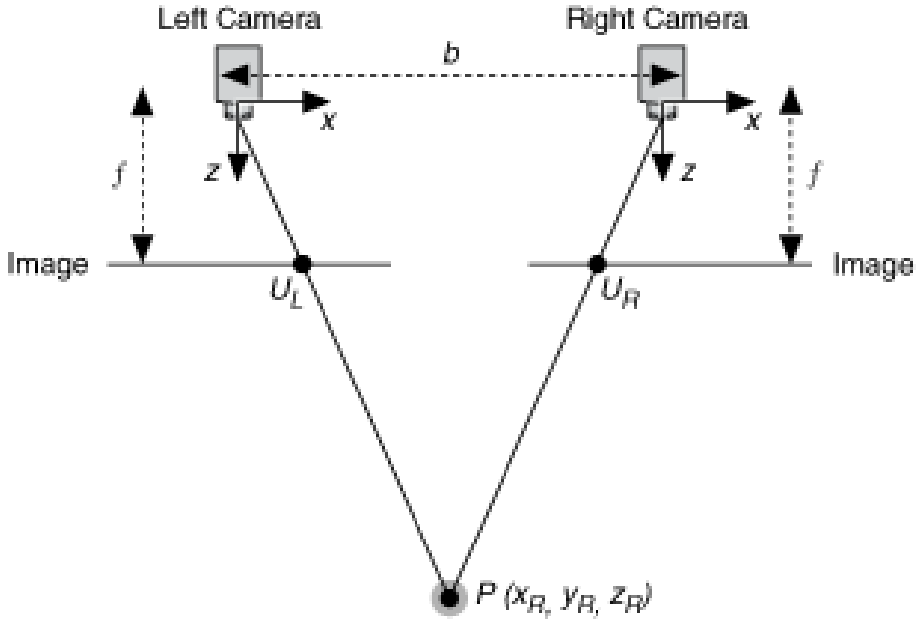
2.5.2.2 Stereo vision

Stereo vision là quá trình trích xuất thông tin 3D từ những bức ảnh kỹ thuật số được chụp bởi hai camera nằm trên cùng một đường thẳng nhưng có vị trị khác nhau từ đó lấy được hai quang cảnh khác nhau của cùng một bối cảnh.

Stereo vision được lấy cảm hừng từ chính đôi mắt của con người khi xác định độ sâu đến các vật thể bằng cách so sánh độ chênh lệch của cùng một vật

thể từ hình ảnh thu được bởi mắt trái và mắt phải (disparity). Não bộ sử dụng sự chênh lệch này để trích xuất thông tin về độ sâu từ các hình ảnh hai chiều trên võng mạc.

Một hệ thống stero vision thường được minh hoạ lại như sau:



Hình 2.23: : Hình minh hoạ hệ thống stereo vision

Trong đó b (baseline) là khoảng cách giữa hai camera, f (focal length) là tiêu cự của hai camera, x (X-axis) là trục X của camera, z (optical axis) là trục quang học của camera, P là một điểm trong thực tế, U_L và U_R lần lượt là điểm P trong ảnh được chụp bởi camera trái và camera phải.

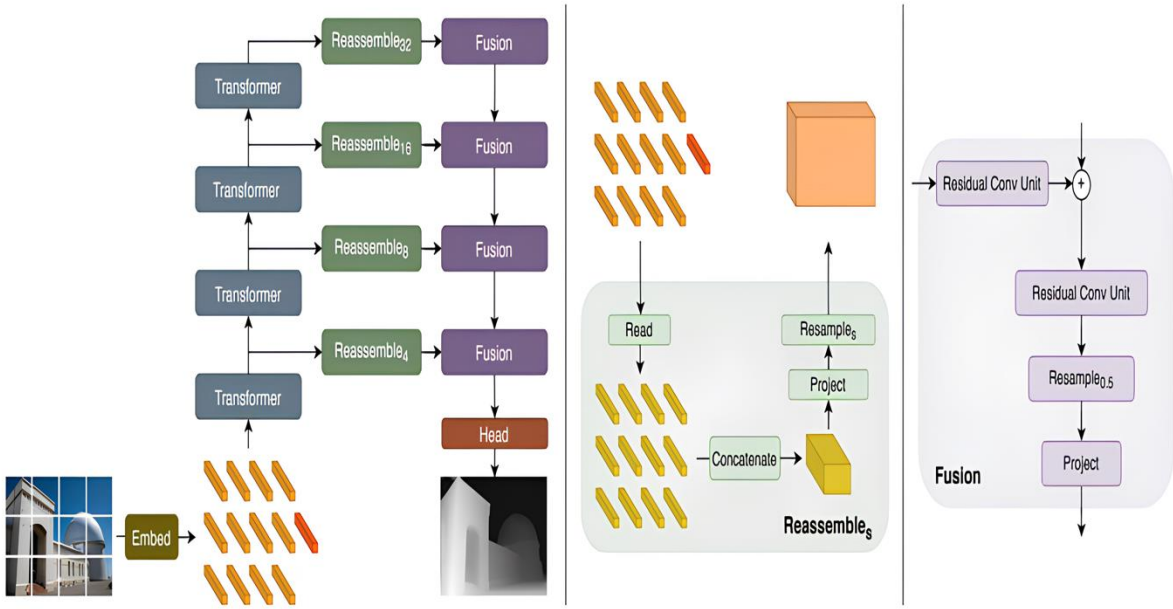
Khi đó độ sâu z sẽ được tính theo công thức:

$$z = f\frac{b}{d} = f\frac{b}{U_L - U_R} \quad (9)$$

2.5.3 Mô hình Dense Prediction Transformer (DPT)

Dense Prediction Transfomer (DPT) là một mô hình thuộc nhóm Transformer ứng dụng trong thị giác máy tính. DPT đã được chứng minh là một phương pháp chính xác và hiệu quả đối với các bài toán về dense prediction (đưa ra dự đoán đối với mỗi pixel đầu của bức ảnh đầu vào, trái với sparse – đưa ra một dự đoán duy nhất đối với toàn bộ các điểm ảnh). Bằng cách tận dụng khả năng của mô hình ViT (vision transformer), các mô hình DPT đã cung cấp độ phân giải đặc trưng cao hơn cùng với đó là quy trình encoding – decoding hiệu quả.

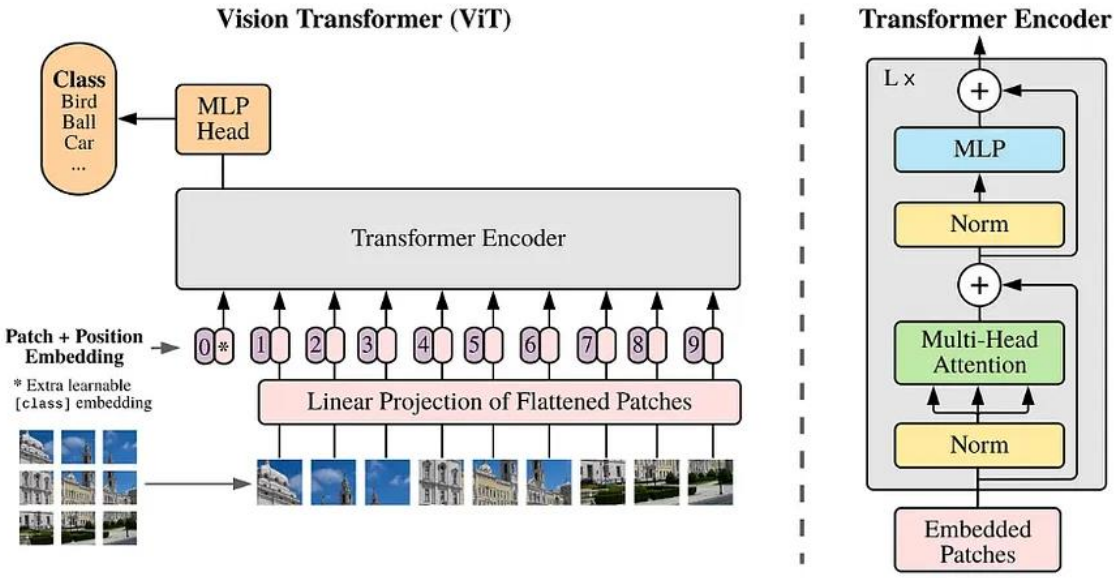
Cấu trúc của mô hình DPT gồm ba thành phần chính là khối Transfomer encoder, khối Reassemble và khối Fusion. Trước khi trả về kết quả dự đoán cuối cùng, đầu ra của khối Fusions sẽ được đi qua Head để giảm kích thước về đúng với hình ảnh dầu vào, từ đó cho ra kết quả cuối cùng.



Hình 2.24: Cấu trúc của mô hình DPT [13]

2.5.3.1 Transformer encoder

Mô hình DPT sử dụng mô hình ViT làm nền tảng cho khối Transfomer encoder của mình.



Hình 2.25: : Cấu trúc mô hình ViT [14]

Tiến trình embedding

Embedding là kỹ thuật đưa một vector có số chiều lớn, thưởng ở dạng thưa về một vector có chiều nhỏ hơn, thường ở dạng dày đặc.

Đối với mô hình ViT, hình ảnh đầu vào $x \in \mathbb{R}^{W \times H \times C}$ sẽ được chia thành các patch phẳng 2D $x_p \in \mathbb{R}^{N \times (P^2 \times C)}$ với (H,W) là độ phân giải của hình ảnh đầu vào, C số channel, P là độ phân giải của mỗi patch và N là số lượng các patch. N sẽ được tính bằng công thức sau [14]:

$$N = \frac{HW}{P^2} \tag{10}$$

Transformer sử dụng vector ngầm với kích thước không đổi D xuyên suốt tất cả các layer. Vì vậy sau khi hình ảnh đầu vào đã được chia thành các patch và làm phẳng, ta sử dụng phép biến đổi tuyến tính các patch với ma trận E (ma trận patch embedding) có kích thước $(P^2 \times C) \times$ D, từ đó thu được một tọken: $x_p^i E$ có kích thước (N $\times D$) với $i \in [1, N]$.

Tiếp theo đó mỗi một token sẽ được trang bị thêm một thành phần có khả năng học được về vị trí của token đó trong hình ảnh đầu vào là e_{pos}^{N}.

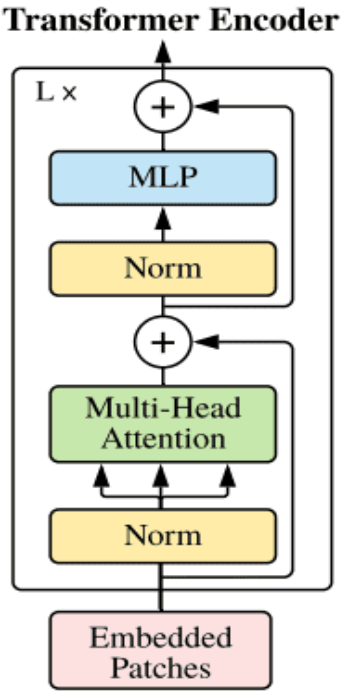
Ngoài ra mô hình ViT còn sử dụng một token đặc biệt không có trong hình ảnh đầu vào mà phục vụ cho việc phân loại sau này là classification token (readout token trong mô hình DPT).

Như vật sau tiến trình embedding ta thu được đầu vào của khối Transformer encoder có dạng tổng quát như sau:

$$Z_0 = [x_{class}, x_p^1 E, \ldots, x_p^N E] + E_{pos}$$

Cấu trúc Transformer encoder

Khối Transformer encoder được cầu thành từ các Transformer layer, mỗi một Transformer layer bao gồm ba thành phần chính là LayerNorm (LN), Multiheaded Self-Attention layer (MSA), Multi-layer Perceptron (MLP) hai layer sử dụng hàm kích hoạt phi tuyến GELU (Gaussian Error Linear Units) kết hợp cùng cơ chế skip connections.



Hình 2.26: : Cấu trúc Transformer Encoder bao gồm L Transformer layers [14]

LayerNorm

Lớp này không mang thêm bất kì yếu tố phụ nào giữa các hình ảnh huấn luyện, đông thời lớp này giúp chuẩn hoá đầu vào trước khi đưa vào lớp MSA hoặc MLP. Điều này giúp mô hình có thể cải thiện về thời gian đào tạo, hiệu suất tổng thể cũng như tăng độ ổn định cho mô hình.

Multiheaded Self-Attention (MSA)

Trong Transformer, cơ chế Self-Attention được sử dụng để biển diễn nối quan hệ của từng vùng ảnh đối với các vùng ảnh xung quanh thông qua "Attention Score". Attention Score và Attention được tính theo công thức [15]:

$$AttentionScore = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (11)$$

$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (12)$$

Trong đó:

- Q: Query
- K: Key
- V: Value
- d_k: Số chiều của Q, K, V

Tuy nhiên cơ chế Selft-Attention lại không thể bóc tách từng phần nhỏ thông tin tại những khu vực nhất định. Do đó cơ chế Multiheaded self-attention được sử dụng.

Multiheaded Self-Attention là cơ chế mà mô hình sẽ sử dụng nhiều Self-Attention, mỗi Self-Attention học một phần thông tin của toàn bộ hình ảnh, mỗi Self-Attention này được gọi là "Head".

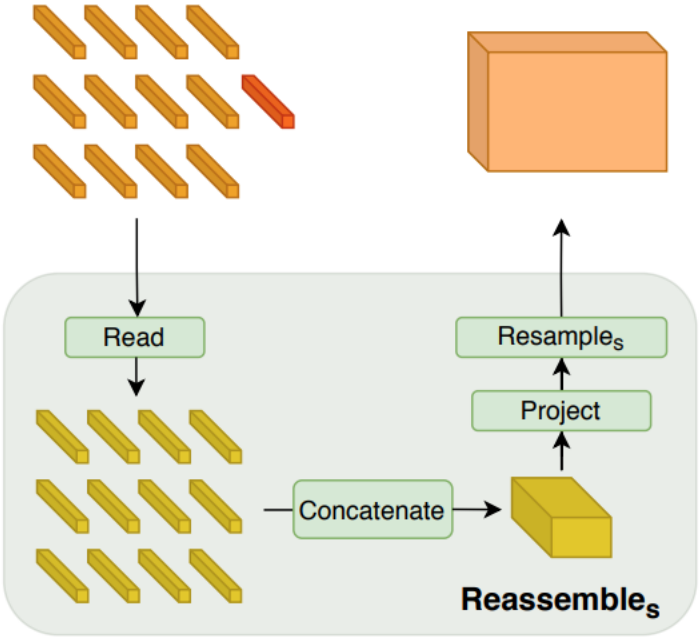
Việc sử dụng cơ chế Multiheaded Self-Attention giúp độ phức tạp khi tính toán của mỗi lớp do có thể thực hiện các phép tính song song đồng thời cũng bóc tách được từng phần nhỏ thông tin trong từng khu vực của hình ảnh.

Cơ chế skip connections

Cơ chế skip connections giúp lưu trữ thông tin về vị trí của tokens trong hình ảnh đầu vào, đảm bảo các tokens luôn lưu giữ đúng vị trí trong toang bộ mô hình DPT.

2.5.3.2 Reassemble

Khối Reassemble có nhiệm vụ tái tổng hợp lại các token đã được xử lý ở khối Transformer encoder thành một dạng biểu diễn giống hình ảnh ở nhiều độ phân giải khác nhau.



Hình 2.27: Cấu trúc khối Reassemble [13]

Quá trình tái tổng hợp được diễn ra theo ba bước:

$$\text{Reassemble}_s^{\breve{D}}(t) = (Resample_s \circ Concatenate \circ Read)(t) \quad (13)$$

Trong đó s là tỷ lệ kích thước đầu ra của đại diện giống dạng hình ảnh được phục hồi so với kích thước hình ảnh đầu vào, $\breve{D}$ là số chiều của đặc trưng đầu ra.

Read

Trong quá trình này, (Np + 1) tokens được ánh xạ thành một tập hợp gồm Np tokens để có thể ghép nối thành dạng biểu diễn giống hình ảnh.

$$\text{Read}: \mathbb{R}^{N_p+1} \longrightarrow \mathbb{R}^{N_p \times D} \quad (14)$$

Quá trình này là vô cùng cần thiết để có thể xử lý readout token mộ cách hợp lý nhất. Có ba cách để xử lý readout token:

$$\mathrm{Read}_{\mathrm{ignore}}(\mathrm{t}) = \mathrm{t}_1, \mathrm{t}_2, .., \mathrm{t}_\mathrm{n} \quad (15)$$

$$Read_{add}(t) = t_1 + t_0, t_2 + t_0, .., t_n + t_0 \quad (16)$$

$$Read_{proj}(t) = \{mlp(cat(t_1, t_0)), .., mlp(cat(t_n, t_0))\} \quad (17)$$

Concatenate

Sau quá trình Read, N_p tokens còn lại sẽ được tái định hình bằng cách đặt từng token vào đúng vị trí của các patch ban đầu của hình ảnh đầu vào. Kết quả cho ta một feature map với kích thước $\frac{N}{H} \times \frac{N}{H}$ và D channels.

$$Concatenate: \mathbb{R}^{N_p \times D} \longrightarrow \mathbb{R}^{\frac{N}{H} \times \frac{N}{H} \times \mathrm{D}} \quad (18)$$

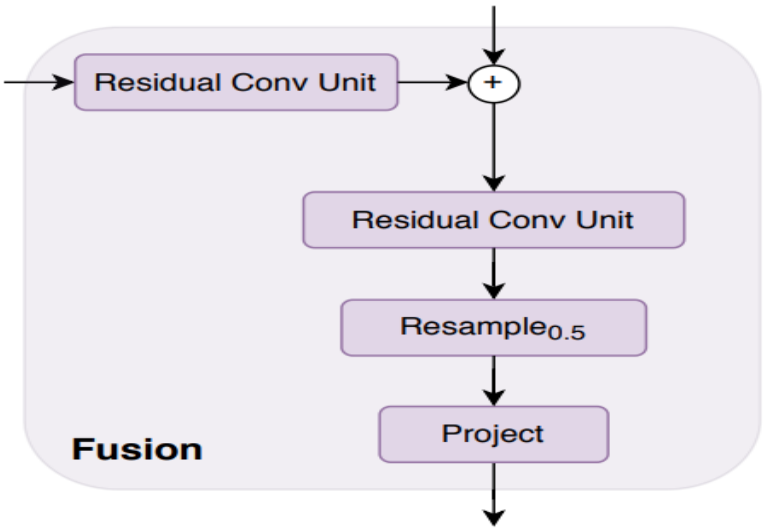
Resample

Cuối cùng quá trình đầu ra của quá trình Concatenate được đưa vào quá trình Resample để đạt được kích thước đầu ra là $\frac{N}{H} \times \frac{N}{H} \times \breve{D}$.

$$Resample: \mathbb{R}^{\frac{N}{H} \times \frac{N}{H} \times \mathrm{D}} \longrightarrow \mathbb{R}^{\frac{N}{H} \times \frac{N}{H} \times \breve{D}} \quad (19)$$

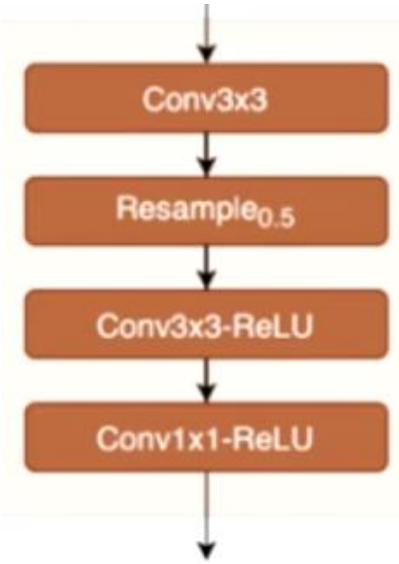
2.5.3.3 Fusion và Head

Khối Fusion được sử dụng để kết hợp các đầu ra của khối Reassemble đồng thời giảm kích thước ở đầu ra của khối này để tạo thành một hình ảnh.



Hình 2.28: Cấu tạo khối Fusion [13]

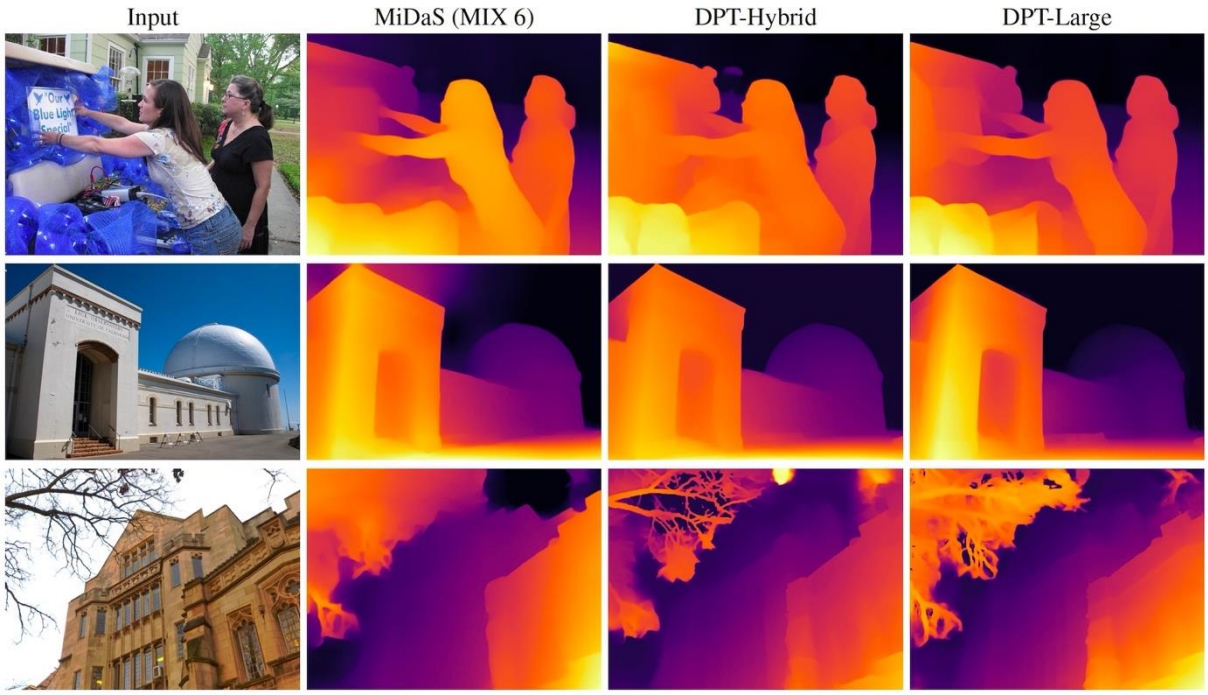
Khối Head là sự kết hợp của các hàm tích chập và hàm kích hoạt ReLU để giảm kích thước đầu ra của hình ảnh có được cừ khối Fusion.



Hình 2.29: : Cấu tạo lớp Head [13]

2.5.3.4 Đầu ra của mô hình

Đầu ra của mô hình DPT là một depthmap mang thông tin về khoảng cách tương đối từ vật thể đến camera và là đầu ra của khối tạo depthmap được sử dụng trong đồ án này.

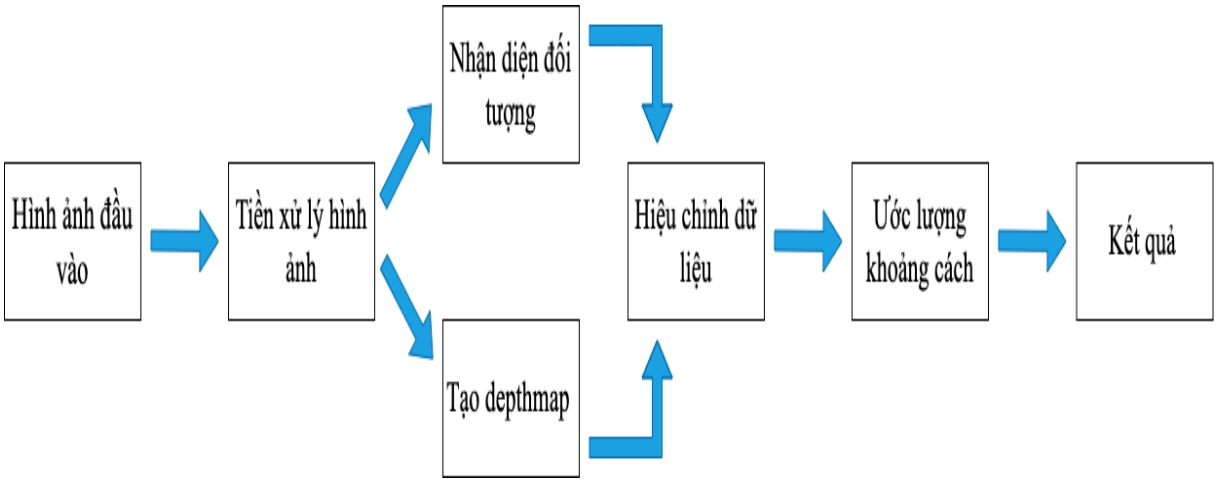


Hình 2.30: : Đầu ra của mô hình DPT và so sánh mới mô hình MiDaS [13]

CHƯƠNG 3 GIẢI PHÁP THỰC HIỆN

3.1 Tổng quan về mô hình

Mô hình được xây dựng để ước lượng khoảng cách từ mặt người tới camera.



Hình 3.1: Mô hình tổng quan của hệ thống ước lượng khoảng cách

3.2 Phương pháp thực hiện

3.2.1 Khối phát hiện đối tượng

Mô hình được sử dụng trong bài toán nhận diện đối tượng là YOLOv8. Sau khi nhận diện được đối tượng, toàn bộ dữ liệu trong bounding box có được từ mô hình YOLOv8 sé được sử dụng để ước lượng khoảng cách trong khối ước lượng khoảng cách.

3.2.2 Khối tạo depthmap

Hình ảnh đầu vào sẽ được đưa qua mô hình DPT để tạo ra depthmap sau đó kết hợp với bounding box có được từ mô hình phát hiện đối tượng để làm đầu vào cho khối hiệu chỉnh dữ liệu.

3.2.3 Khối ước lượng khoảng cách

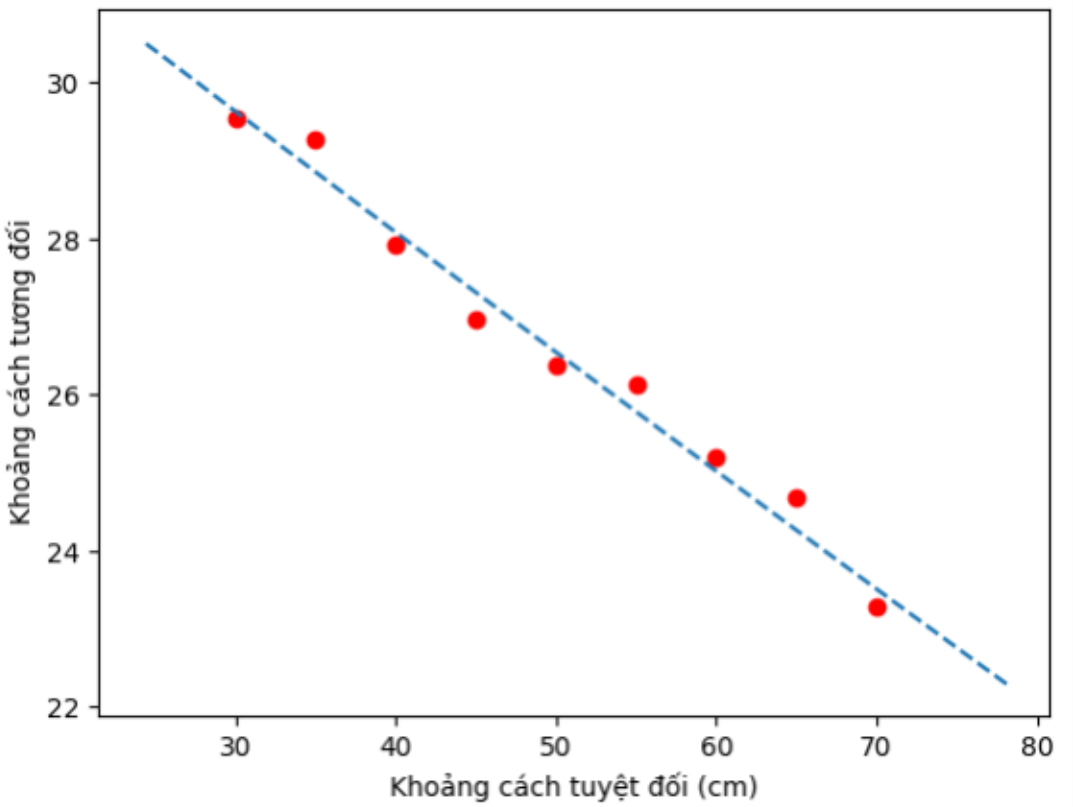
Sau khi khối hiệu chỉnh nhận được đầu vào là sự kết hợp giữa đầu ra của hai khối phát hiện đối tượng và khối tạo depthmap sẽ cho ra kết quả là khoảng cách giữa đối tượng và camera. Tuy nhiên khoảng cách này chỉ là khoảng cách tương đối giữa đối tượng và camera, không phải khoảng cách tuyệt đối (khoảng cách từ đối tượng đến camera trong thực tế, là khoảng cách cần tìm).

Để có thể tìm được khoảng cách tuyệt đối từ đối tượng đến camera thông qua khoảng cách tương đối, ta cần phải xây dựng mối quan hệ hàm số giữa hai đại lượng này. Dựa trên kỹ thuật của Taha và Jizat [16], mối liên hệ giữa khoảng cách tương đối và khoảng cách tuyệt đối có thể được biểu diễn dưới dạng hàm bậc hai như sau:

$$Y = (c_0 + c_1 X + c_2 X^2) \times h \qquad (20)$$

Trong đó Y (cm) là khoảng cách tuyệt đối cần tìm, X là khoảng cách tuyệt đối có được là đầu ra của khối hiệu chỉnh dữ liệu, h là khoảng cách từ camera tới mặt đất. Do đối tượng được sử dụng làm đầu vào trong đồ án này là mặt người nên ta có thể gán giá trị h = 1.

Việc xác định các hệ số c_0, c_1, c_2 có thể thông qua phương pháp bình phương tối thiểu. Từ các dữ liệu đã được đo đạc trước, ta có biểu đồ thể hiện mối quan hệ giữa khoảng cách tương đối và khoảng cách tuyệt đối như sau:



Hình 3.2: Biểu đồ mối quan hệ giữa khoảng cách tương đối và khoảng cách tuyệt đối

Từ các dữ liệu như biểu đồ trên ta tìm được hàm số thể hiện mối quan hệ giữa khoảng cách tương đối và khoảng cách tuyệt đối như sau:

$$Y = 235.61763103 + (-7.45535009)X + 0.017384634X^2$$

3.3 Kết luận

Như vậy chương 3 đã khái quát về tổng quan của mô hình ước lượng khoảng cách cũng như phương thức thực hiện của mô hình này. Các kết quả có được sau khi áp dụng mô hình này sẽ được trình bày trong chương 4.

CHƯƠNG 4 ĐÁNH GIÁ KẾT QUẢ

4.1 Tập dữ liệu

4.1.1 Tập dữ liệu KITTI

Tập dữ liệu KITTI là một trong những tập dữ liệu phổ biến nhất trong lĩnh vực thì giác máy tính, đặc biệt là trong việc dự đoán độ sâu và tư thế của vật thể. Tập dữ liệu này bao gồm hơn 200 video về hình ảnh đường phố buổi sáng được chụp bởi camera RGB và depthmap được chụp bởi Velodyne laser scanner.

4.1.2 Tập dữ liệu Coco

Tập dữ liệu Coco (Common object in context) là tập dữ liệu về phát hiện vật thể, phân đoạn quy mô lớn. Nó được thiết kế đẻ khuyến khích nghiên cứu với nhiều loại vật thể khác nhau và thường được sử dụng để kiểm chuẩn cho các mô hình thị giác máy tính. Đây là một tập dữ liệu cần thiết cho các nhà nghiên cứu và phát triển làm về các nhiệm vụ phát hiện vật thể, phân đoạn và ước tính tư thế vật thể.

4.2 Các tham số đánh giá

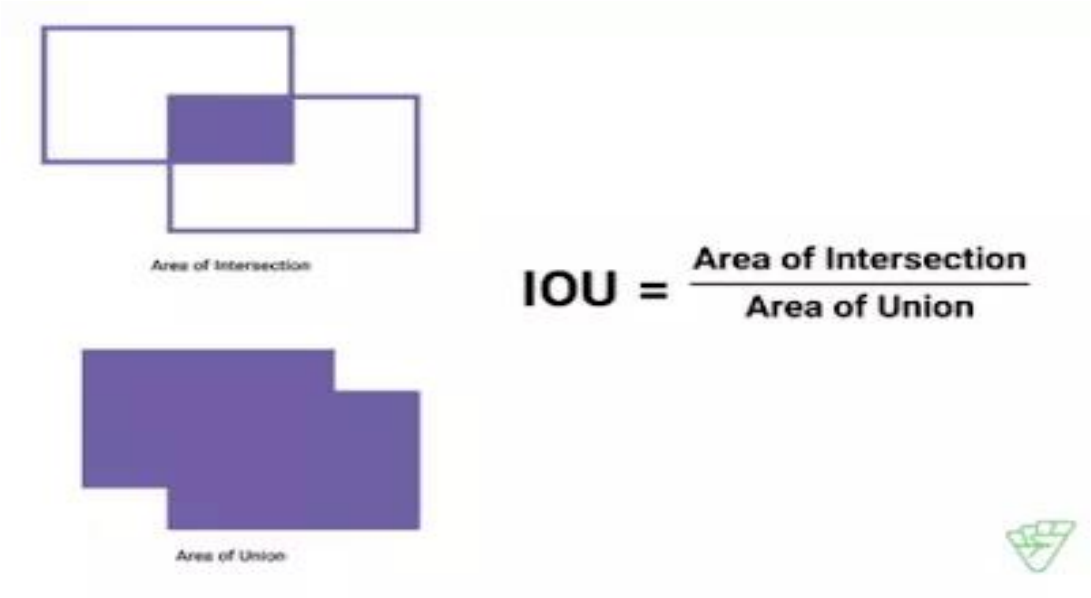
4.2.1 Khối xác định đối tượng

4.2.1.1 Intersection over Union (IoU)

Intersection over Union (IoU) là chỉ số đánh giá được sử dụng để đo độ chính xác của một mô hình nhận diện vật thể trên tập dữ liệu cụ thể. IoU đơn giản chỉ là một chỉ số đánh giá. Mọi thuật toán có khả năng predict ra các bounding box làm output đều có thể được đánh giá thông qua IoU.

Để áp dụng được IoU để đánh giá một mô hình nhận diện vật thể bất kì ta cần:

- Grond-truth bounding box: Bounding box đúng của đối tượng
- Predicted bounding box: Bounding box được mô hình sinh ra



Hình 4.1: Công thức tính IoU [17]

Giá trị IoU tốt nhất có thể đạt được là 1, khi vùng dự đoán và vùng tham chiếu hoàn toàn trùng khớp. Ngược lại, IoU bằng 0 khi hai vùng không có phần nào chồng lấp lên nhau. Do đó, IoU không thể dùng làm hàm mất mát vì nó không có tính liên tục. Dựa vào các chỉ số này ta có các khái niệm về True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN):

- True Positive (TP): Là các predicted box với IoU ≥ ngưỡng. TP hiểu là dự đoán đúng, thành positive, trong TH này dự đoán đúng là các predicted box với IoU > threshold, positive nghĩa là có vật thể.
- True Negative (TN): Không được dùng. Đây là những phần của ảnh không chứa đối tượng (không được gán ground-truth box) và được dự đoán không chứa đối tượng (thực chất mô hình chỉ đưa ra các vùng có khả năng chứa đối tượng). Điều này có nghĩa rằng các vùng khác trong ảnh được dự đoán là không chứa đối tượng. Số lượng TN như vậy là vô số.

- False Positive (FP): Là các predicted box với IoU < threshold. FP hiểu là dự đoán sai thành positve, trong TH này dự đoán sai là predicted box với IoU < threshold. Ở đây ám chỉ có dự đoán ra được bounding box
- False Negative (FN): Là mô hình không dự đoán được đối tượng trong ảnh đối với ground truth box hay groudth-truth box không được gắn với predicted bounding box nào

4.2.1.2 Precision và Recall

Precision là tỉ lệ dự đoán đúng (khớp với ground truth boxes) so với tổng số các dự đoán (predicted bounding boxes). Do đó ta có công thức tính "Precision" như sau:

$$\text{Precision} = \frac{\text{vật thể dự đoán đúng}}{\text{tổng số các dự đoán}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad (21)$$

Recall (độ nhạy) thể hiện tỉ lệ dự đoán đúng trên tổng số ground truth:

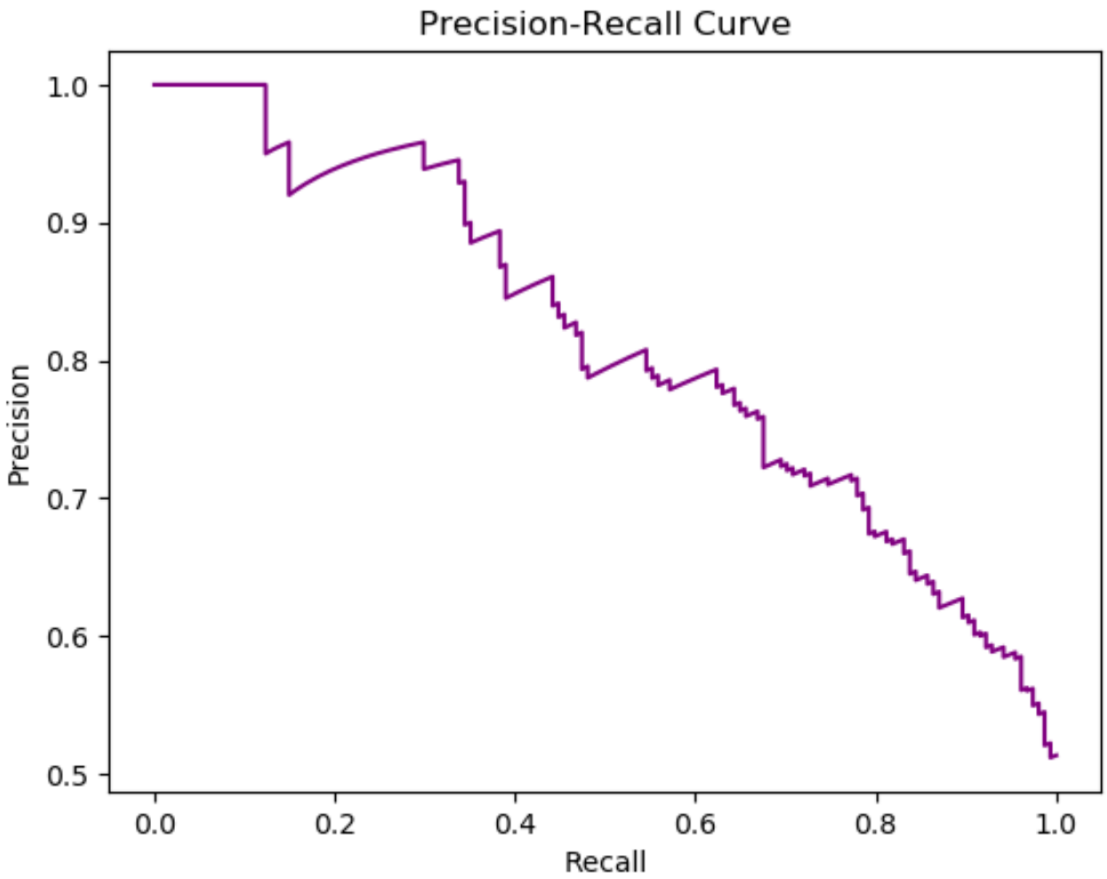
$$\text{Recall} = \frac{\text{vật thể dự đoán đúng}}{\text{tổng số ground truths}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad (22)$$

Như vậy một mô hình xác định vật thể tốt phải có cả Precision và Recall cao.

4.2.1.3 Đường cong Precision-Recall

Đường cong Precision-Recall là đường thể hiện mối quan hệ của precision so với recall khi thay đổi ngưỡng confidence score.

Thông thường khi tăng ngưỡng thì precision tăng và recall giảm.



Hình 4.2: : Biểu đồ thể hiện mối quan hệ giữa Precision và Recal [17]

4.2.1.4 Mean average precision (mAP)

Chỉ số average precision (AP) là phần diện nằm bên dưới đường cong Precision-Recall:

$$AP = \int_0^1 P(r)dr \tag{23}$$

Trong đó P(r) thể hiện sự phụ thuộc của Precision vào Recall.

Giá trị của Precision và Recall luôn nằm trong [0,1] do đó giá trị của AP cùng nằm trong [0,1].

Tuy nhiên, do việc tìm hàm P(r) có thể rất phức tạp, AP có thể được tính theo hai cách sau:

11-point interpolation

Trong cách tính này, ta chia các giá trị Recall trong [0,1] thành 11 điểm cách đều nhau sau đó tính trung bình Precision tại 11 điểm này.

$$AP = \frac{1}{11} \sum_{r \in [0,1]} AP_r = \frac{1}{11} \sum_{r \in [0,1]} P_{interp}(r) \quad (24)$$

Trong đó giá trị nội suy của Precision cho Recall được xác định như sau:

$$P_{interp}(r) = \max_{\tilde{r} \geq r} P(\tilde{r}) \quad (25)$$

All-point interpolated AP

Trong phương pháp này thay vì chỉ nội suy tại 11 điểm cố định, ta nội suy tại tất cả các điểm trong tập dữ liệu. Khi đó AP sẽ được tính theo công thức:

$$AP = \sum (r_{n+1} - r_n) P_{interp}(r_{n+1}) \quad (26)$$

$$P_{interp}(r) = \max_{\tilde{r} \geq r} P(\tilde{r}) \quad (27)$$

4.2.2 Khối ước lượng khoảng cách

4.2.2.1 Chênh lệch tương đối

Chênh lệch tương đối đo lường sự khác biệt trung bình giữa giá trị dự đoán và giá trị thực tế, được chuẩn hoá theo giá trị thực tế. Công thức tính chênh lệch tương đối là:

$$AbsRel = \frac{1}{N} \sum_{d \in N} \frac{|d - d^*|}{d^*} \quad (28)$$

Trong đó:

- N: Tổng số điểm dữ liệu
- d: Giá trị dự đoán
- d^*: Giá trị thực tế

4.2.2.2 Chênh lệch tương đối bình phương

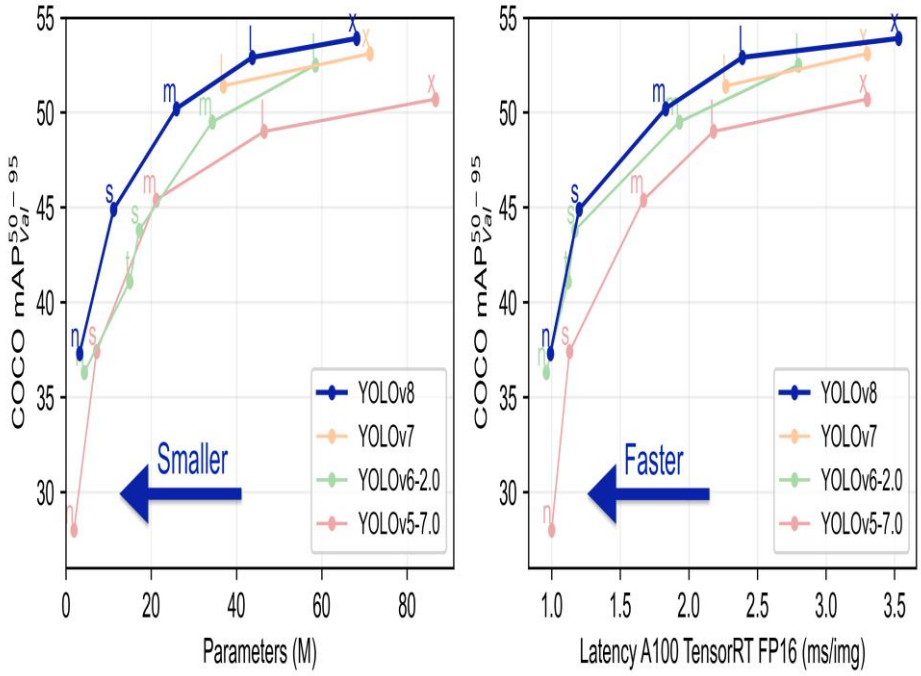
Chênh lệch tương đối bình phương cũng tương tự so với chênh lệch tương đối tuy nhiên nó sử dụng bình phương sai khác giữa giá trị dự đoán và giá trị thực tế thay vì sử dụng giá trị tuyệt đối. Công thức tính chênh lệch tương đối bình phương là:

$$SqrRel = \frac{1}{N}\sum_{d \in N}\frac{(d - d^*)^2}{d^*} \quad (29)$$

4.3 Kết quả thử nghiệm

4.3.1 Khối xác định đối tượng

Kết quả so sánh về mAP@0.5:0.95 giữa các mô hình YOLOv5, YOLOv6, YOLOv7 và YOLOv8 được thể hiện qua biểu đồ sau:



Hình 4.3: : Biểu đồ so sánh giữa các mô hình YOLO trên tập dữ liệu COCO [18]

Kết quả so sánh về các chỉ số Recall và Precision giữa YOLOv6, YOLOv7 và YOLOv8 được cho ở bảng sau:

	Precision	Recall
YOLOv6	0.9	0.8
YOLOv7	0.95	0.85
YOLOv8	0.95	0.9

Bảng 1: Bảng so sánh Recall, Precision của ba mô hình YOLO

4.3.2 Khối ước lượng khoảng cách

Sau khi thử nghiệm thực tế ta có bảng kết quả sau:

Khoảng cách chính xác (cm)	Khoảng cách dự đoán (cm)	Sai số (cm)
33	35.35	2.35
38	38.42	0.42
43	39.4	3.6
46	43.4	2.6
54	52.16	1.84
57	51.47	5.53
64	63.55	0.45
AbsRel = 2.40 (cm)		

Bảng 2: Một số kết quả đo đạc thực tế

4.4 Kết luận

Như vậy chương 4 đã đưa ra khái quát về hai tập dữ liệu được sử dụng phổ biến trong học sâu và học máy là tập dữ liệu KITTI và tập dữ liệu COCO.

Chương này cũng đã đưa ra các tham số tham chiếu để đánh giá hai khối chính của mô hình là khối xác định vật thể và khối ước lượng khoảng cách cũng như đưa ra được các kết quả thử nghiệm thực tế của hai khối này.

KẾT LUẬN CHUNG

Nhìn chung mô hình được sử dụng trong đề tài đã đưa ra được kết quả khả quan trong việc ước lượng khoảng cách từ vật thể tới camera. Tuy nhiên vẫn còn nhiều hạn chế trong việc ước lượng khoảng cách trong các điều kiện môi trường và độ chính xác còn phụ thuộc nhiều điều kiện sáng của dữ liệu thử nghiệm so với dữ liệu huấn luyện. Độ đa dạng về các đối tượng trong mô hình cũng là một điểm còn hạn chế do độ đa dạng về các đối tượng chưa cao.

Mặc dù vậy mô hình vẫn có tiềm năng để phát triển cả về độ chính xác cũng như độ đa dạng về số lượng các đối tượng được sử dụng trong mô hình.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] T. Smits, "ResearchGate," Jan 2018. [Online]. Available: https://researchgate.net/figure/mage-of-Abraham-Lincoln-as-a-matrix-of-pixel-values_fig1_330902210. [Accessed 25 June 2024].

[2] Funix, "Một số khái niệm về ảnh số và cấu trúc của ảnh số," Funix, Việt Nam, 2022.

[3] F. Feng, "ResearchGate," September 2019. [Online]. Available: https://www.researchgate.net/figure/Commonly-used-activation-functions-a-Sigmoid-b-Tanh-c-ReLU-and-d-LReLU_fig3_335845675. [Accessed 26 June 2024].

[4] Afshine Amidi, Shervine Amidi, "stanford.edu," Stanford University, Stanford.

[5] P. Sharma, "Introduction to Feed-Forward Neural Network in Deep Learning," Analytics Vidhya, 2024.

[6] P. AI, "Understanding the 3 most common loss functions for Machine Learning Regression," Towards Data Science Practicus AI, 2019.

[7] aptech, "Mạng CNN là gì và những kiến thức cơ bản cần biết về mạng CNN," aptech.

[8] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "arxiv.org," 8 June 2015. [Online]. Available: https://arxiv.org/abs/1506.02640. [Accessed 5 July 2024].

[9] P. Đ. Khánh, "Khoa học dữ liệu - Khanh's blog," 6 January 2020. [Online]. Available: https://phamdinhkhanh.github.io/2020/01/06/ImagePreprocessing.html. [Accessed 04 July 2024].

[10] L. glass, "Depth Maps: How Software Encodes 3D Space," 2023.

[11] "Introduction to Lidar," Mathworks.

[12] A. U. Alberto Volkmann BA, "Producing Geographic Data with LIDAR," 2015.

[13] René Ranftl, Alexey Bochkovskiy, Vladlen Koltun, "arxiv.org," 21 March 2021. [Online]. Available: https://arxiv.org/abs/2103.13413. [Accessed 5 July 2024].

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby, "arxiv.org," 22 October 2020. [Online]. Available: https://arxiv.org/abs/2010.11929v2. [Accessed 05 July 2024].

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin , "arxiv.org,"

12 June 2017. [Online]. Available: https://arxiv.org/abs/1706.03762. [Accessed 3 July 2024].

[16] Zahari Taha, Jessnor Arif Mat Jizat, "ReserachGate," January 2012. [Online]. Available: https://www.researchgate.net/publication/216535956_A_Comparison_of_Two_Approaches_for_Collision_Avoidance_of_an_Automated_Guided_Vehicle_Using_Monocular_Vision. [Accessed 29 06 2024].

[17] V. Buởi, "MAP TRONG OBJECT DETECTION," Viblo, 2023.

[18] Padilla, Rafael and Passos, Wesley L. and Dias, Thadeu L. B. and Netto, Sergio L. and da Silva, Eduardo A. B., "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," vol. 10.