PHƯƠNG PHÁP PHÁT HIỆN ĐỐI TƯỢNG TRONG HỆ THỐNG NHÚNG PHỤC VỤ BÀI TOÁN ĐIỀU KHIỂN ROBOT BÁM ĐỐI TƯỢNG THỜI GIAN THỰC

OBJECT DETECTION METHOD IN EMBEDDED SYSTEM FOR REAL-TIME OBJECT TRACKING ROBOT CONTROL PROBLEM

Sái Văn Cường[1,*], Nguyễn Văn Đức[1], Bùi Thị Duyên[2]

DOI: http://doi.org/10.57001/huih5804.2024.361

TÓM TẮT

Trong bài báo này, chúng tôi thực hiện nghiên cứu so sánh kiến trúc mạng nơ-ron SSD nguyên bản sử dụng VGG-16 làm mạng cơ sở với kiến trúc SSD sửa đổi bằng cách thay thế mạng cơ sở VGG-16 bằng các phiên bản khác nhau của mạng MobileNet. Mục tiêu của nghiên cứu là xây dựng được mô hình mạng nơ-ron tích chập sâu tối ưu, đảm bảo được sự cân bằng giữa độ chính xác và tốc độ trong bài toán phát hiện và bám đối tượng để có thể thực thi trên nền tảng thiết bị nhúng với tài nguyên tính toán hạn chế. Các mô hình được đánh giá so sánh trên mạch nhúng Jetson Nano trên các tập dữ liệu có kích thước và độ phức tạp khác nhau để có kết luận toàn diện về độ chính xác và tốc độ. Phương pháp đề xuất dựa trên mạng Mobilenet đã đạt được độ chính xác gần như tương đương và đạt được tốc độ suy luận nhanh hơn rất nhiều so với mô hình SDD nguyên bản sử dụng mạng VGG-16, cụ thể đạt độ chính xác tổng thể mAP cao nhất là 84% trên tập dữ liệu kiểm tra và tốc độ suy luận trung bình ~25 FPS sau khi tối ưu.

Từ khóa: *Phát hiện đối tượng, CNN, SSD, VGG16, MobileNet.*

ABSTRACT

In this paper, we conduct a comparative study of the original SSD neural network architecture using VGG-16 as the backbone network with a modified SSD architecture by replacing the VGG-16 backbone network with different versions of the MobileNet network. The goal of the study is to build an optimal deep convolutional neural network model that ensures a balance between accuracy and speed in the object detection and tracking problem so that it can be executed on an embedded device platform with limited computational resources. The models are evaluated on a Jetson Nano for datasets of different sizes and complexities to have a comprehensive conclusion about accuracy and speed. The proposed method based on MobileNet network achieved almost equivalent accuracy and achieved much faster inference speed than the original SDD model using VGG-16 network, specifically achieving the highest overall mAP accuracy of 84% on the test dataset and an average inference speed of ~25 FPS after optimization.

***Keywords*:** *Object Detection, CNN, SSD, VGG16, MobileNet.*

[1]Viện Tự động hoá Kỹ thuật Quân sự, Viện Khoa học và Công nghệ Quân sự
[2]Trường Đại học Điện lực
[*]Email: svcuonghvktqs@gmail.com
Ngày nhận bài: 10/9/2024
Ngày nhận bài sửa sau phản biện: 15/11/2024
Ngày chấp nhận đăng: 28/11/2024

1. ĐẶT VẤN ĐỀ

Phát hiện đối tượng (*object detection*) là một trong những bài toán quan trọng của thị giác máy tính dùng để phân loại và xác định vị trí các đối tượng vật thể có trong ảnh hoặc video, là cơ sở của nhiều tác vụ thị giác máy tính khác, chẳng hạn như phân đoạn trường hợp (*instance*

segmentation) [1] và bám đối tượng (*object tracking*) [2]. Các ứng dụng của phát hiện đối tượng trải rộng trong nhiều lĩnh vực khác nhau như: công nghệ robot (*robottics*), xử lý ảnh y khoa, các hệ thống giám sát, hệ thống tương tác người-máy, giao thông thông minh,... Trong công nghệ robot, phát hiện đối tượng hỗ trợ việc định vị cũng như nhận dạng các đối tượng nhờ đó robot có thể tương tác chính xác với các đối tượng trong thực tế.

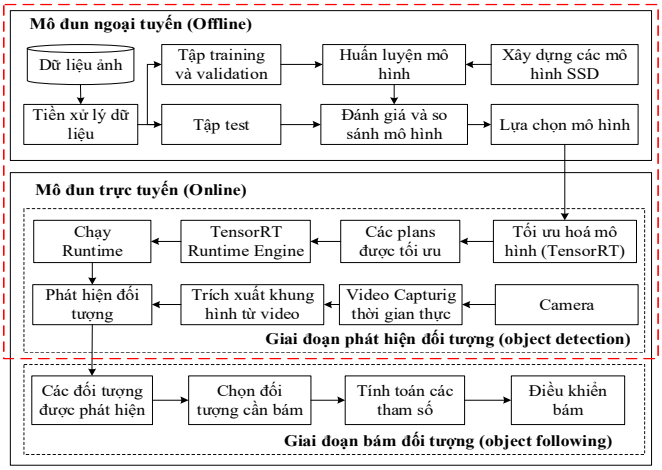
Trong những năm gần đây, nhờ sự phát triển nhanh chóng về mặt dữ liệu cũng như các bước tiến trong lĩnh vực học sâu dựa trên mạng nơ-ron tích chập (*Convolutional Neural Networks, CNN*), bài toán phát hiện đối tượng đã đạt được nhiều bước tiến đáng kể và được ứng dụng rất nhiều trong thực tế, trở thành chức năng thiết yếu trong các việc phát triển robot và xe tự hành. Tuy nhiên, để thực thi các tác vụ sử dụng CNN một cách hiệu quả, chúng ta vẫn cần nhiều sức mạnh tính toán. Do vậy, việc chạy một hệ thống phát hiện đối tượng trên một thiết bị có tài nguyên phần cứng hạn chế có thể là một thách thức. Có nhiều thuật toán nhận dạng và phát hiện đối tượng tiên tiến dựa trên mạng CNN đã được đề xuất. Một số thuật toán phát hiện đối tượng nổi bật được cộng đồng học sâu đề xuất phải kể đến bao gồm Faster R-CNN [3], R-FCN [4], YOLO [5] và SSD [6],... Các phương pháp phát hiện đối tượng này có thể được chia thành hai loại chính là thuật toán phát hiện một giai đoạn và thuật toán phát hiện hai giai đoạn. Các thuật toán phát hiện hai giai đoạn có khối lượng tính toán lớn và khó đạt được hiệu suất cao trên các nền tảng nhúng có tài nguyên hạn chế, vì vậy chỉ có các thuật toán phát hiện một giai đoạn là phù hợp trong bài toán này [7]. Gần đây, hai phương pháp YOLO [5] và SSD [6] nổi lên là các phương pháp phát hiện đối tượng một giai đoạn đạt được hiệu quả tốt nhất. Mặt khác, các tác giả trong [8] đã đề cập rằng bộ phát hiện YOLO không hiệu quả trong việc phát hiện nhiều mục tiêu dày đặc và các vật thể lớn. SSD khắc phục khuyết điểm của YOLO bằng cách sử dụng các khung bao đối tượng mặc định với nhiều tỉ lệ khác nhau, và dự đoán đồng thời trên nhiều bản đồ đặc trưng với kích thước khác nhau. Do đó, trong phạm vi nghiên cứu này, chúng tôi quyết định lựa chọn kiến trúc thuật toán SSD để phát triển các mô hình phát hiện đối tượng thời gian thực, phục vụ bài toán điều khiển robot bám đối tượng trên nền tảng mạch nhúng.

Bởi vì bộ phát hiện đối tượng được đề xuất cần chạy trong một hệ thống nhúng nên độ phức tạp tính toán của mô hình mạng nơ-ron là một điều kiện thiết kế tiên quyết và quan trọng, mô hình phải đáp ứng các tính năng nhanh và có độ trễ thấp. SSD nguyên bản sử dụng VGG-16 làm mạng cơ sở để trích xuất đặc trưng, đây là mạng tiêu biểu và đạt độ chính xác rất cao trong các bài toán phân loại hình ảnh và phát hiện đối tượng. Tuy nhiên VGG-16 có cấu trúc phức tạp và số lượng tham số lớn, do đó việc trển khai nó trong hệ thống nhúng (trong các UAV hoặc robot di động) có tài nguyên tính toán và nguồn cung cấp năng lượng hạn chế khó có thể đạt được hiệu suất xử lý thời gian thực. Vì vậy chúng tôi tập trung vào việc thay đổi mạng cơ sở của SSD bằng các mạng nơ-ron có số lượng tính toán ít và độ chính xác cao để có thể áp dụng trong hệ thống nhúng giới hạn tài nguyên trong thời gian thực. Chúng tôi lựa chọn mạng MobileNetV1 và MobileNetV2 làm mạng cơ sở để trích xuất đặc trưng cho SSD vì hai mạng này có số lượng tính toán ít và độ chính xác cao, được hỗ trợ bởi nhiều nền tảng nhúng khác nhau. Trong nghiên cứu này chúng tôi đề xuất sơ đồ kiến trúc hệ thống phát hiện và bám đối tượng cho robot; và thực hiện nghiên cứu so sánh kiến trúc mạng nơ-ron SSD nguyên bản sử dụng VGG-16 làm mạng cơ sở với kiến trúc SSD sửa đổi bằng cách thay thế mạng cơ sở VGG-16 bằng các phiên bản khác nhau của mạng MobileNet nhằm tìm ra mô hình tối ưu nhất cho hệ thống.

2. PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Hệ thống đề xuất

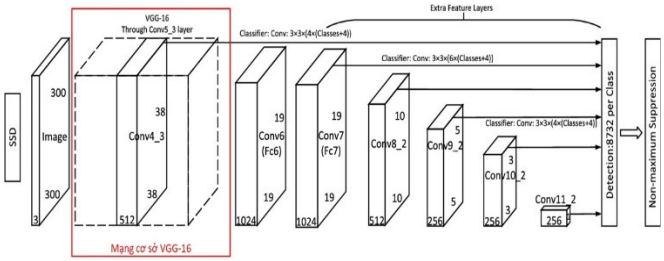
Sơ đồ kiến trúc hệ thống phát hiện và điều khiển robot bám đối tượng được chúng tôi đề xuất trong hình 1. Mục đích chính của hệ thống là phát hiện và theo dõi đối tượng trong thời gian thực. Hệ thống gồm mô đun ngoại tuyến để xây dựng và lựa chọn mô hình phát hiện đối tượng và mô đun trực tuyến triển khai mô hình đã được lựa chọn để thực thi nhiệm vụ bám đối tượng. Mô đun trực tuyến gồm hai giai đoạn đó là phát hiện đối tượng và giai đoạn điều khiển bám đối tượng.



Hình 1. Sơ đồ kiến trúc hệ thống phát hiện và bám đối tượng cho robot đề xuất

Trong thực tế có thể sử dụng trực tiếp mô hình đã huấn luyện từ mô đun ngoại tuyến để phát hiện đối tượng trong mô đun trực tuyến, tuy nhiên qua nghiên cứu, chúng tôi thấy rằng, đối với các hệ thống nhúng có tài nguyên tính toán hạn chế thì mặc dù dùng các mô hình mạng nơ-ron nhẹ cũng khó đảm bảo cho hệ thống hoạt động trơn tru được trong thời gian thực. Vì vậy trong hệ thống này chúng tôi đề xuất thêm một bước đó là tối ưu các mô hình đã được huấn luyện trước khi sử dụng chúng để phát hiện đối tượng.

2.2. Thuật toán phát hiện đối tượng dựa trên SSD



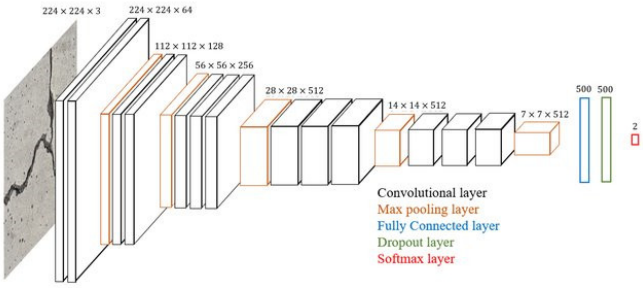
Hình 2. Kiến trúc của mô hình SSD truyền thống [6]

SSD [6] là mô hình một giai đoạn được thiết kế để phát hiện đối tượng trong thời gian thực. Kiến trúc tổng quát của SSD (hình 2) bao gồm hai phần: (i) mô hình mạng cơ sở để trích xuất đặc trưng, và (ii) các lớp nơ-ron phụ trợ (*Extra Feature Layers*) để để dự đoán vị trí của các đối tượng trong ảnh. SSD áp dụng một vài cải tiến bao gồm các đặc trưng đa tỷ lệ (*multi-scale features*) và các hộp mặc định (*default boxes*), cho phép nó đạt được độ chính xác tương đương với các thuật toán hai giai đoạn thậm chí là cao hơn. SSD nguyên bản sử dụng mạng VGG-16 [6] để trích xuất đặc trưng. Điểm rất đặc biệt trong mô hình SSD ở đây là chúng ta *có thể sử dụng cấu trúc mạng bất kì để làm mạng cơ sở cho mô hình*.

2.3. Mạng cơ sở (Backbone networks)

2.3.1. Mô hình học sâu VGG-16

VGG-16 là một mô hình mạng nơ-ron tích chập được đề xuất bởi K. Simonyan và A. Zisserman từ Đại học Oxford trong [10]. Kiến trúc của VGG-16 (hình 3) bao gồm 16 lớp: 13 lớp tích chập (2 lớp conv-conv, 3 lớp conv-conv-conv) đều có kernel 3x3, sau mỗi lớp conv là gộp cực đại giảm kích thước ảnh xuống 0,5 và 3 lớp kết nối hoàn chỉnh. Trong mỗi khối, VGG-16 kết hợp 2 hoặc 3 tầng tích chập, theo sau là tầng gộp cực đại (max pooling) để giảm kích thước bản đồ đặc trưng. VGG-16 chỉ sử dụng các bộ lọc kích thước nhỏ 3x3 (với tầng tích chập) và 2x2 (với tầng gộp) giúp giảm số lượng tham số cho mô hình dẫn đến giảm khối lượng tính toán phải thực hiện. Độ chính xác của mô hình VGG-16 thuộc Top-1 và Top-5 có hiệu quả cao (với 71,3% và 90,01%) đối với bộ dữ liệu ImageNet gồm hơn 14 triệu hình ảnh thuộc 1000 lớp.



Hình 3. Kiến trúc mạng VGG-16 [9]

Mặc dù mô hình mạng VGG-16 có khả năng trích xuất đặc trưng tốt, nhưng kiến trúc mạng của nó quá lớn đối với các nền tảng nhúng. Do đó, VGG-16 có thể vượt quá bộ nhớ hệ thống tối đa và khó đạt được hiệu suất thời gian thực khi chạy trên các hệ thống nhúng. Chính vì vậy, giải pháp chính là sử dụng các mô hình mạng nơ-ron nhỏ, nhẹ và đảm bảo được độ chính xác yêu cầu chính là chìa khoá cho mô hình có thể hoạt động thời gian thực trên các hệ thống nhúng. Mô hình nghiên cứu lựa chọn là MobileNet.

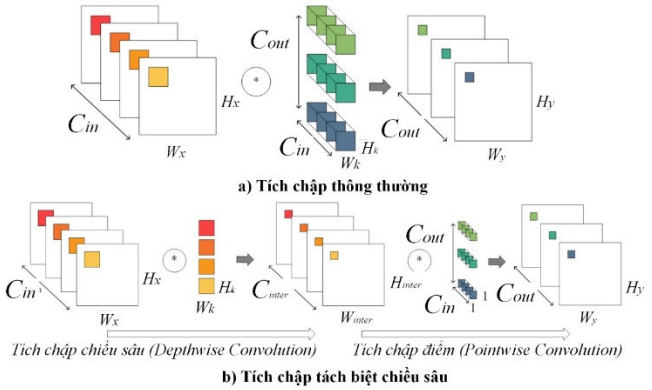
2.3.2. Mô hình học sâu MobileNet

MobileNet là kiến trúc mạng nơ-ron được phát triển bởi nhóm các nhà nghiên cứu Google, được tối ưu hóa cho các thiết bị di động. Kiến trúc này mang lại kết quả chính xác cao trong khi vẫn giữ các tham số và phép toán ở mức thấp nhất có thể nhờ vào cơ chế tích chập tách biệt chiều sâu (*Depthwise Separable Convolution*) có thể thực hiện trích xuất đặc trưng một cách tách biệt trên các channel khác nhau [10]. Vì vậy, khi sử dụng MobileNet để thay thế cho mạng VGG-16 trong SSD có thể tạo ra mô hình phát hiện đạt được hiệu suất thời gian thực.

**** Tích chập tách biệt chiều sâu***

Mô hình MobileNet đầu tiên (trong bài báo sẽ gọi là MobileNetV1) dựa trên các phép tích chập có thể tách theo chiều sâu, là một quy trình phân rã phép tích chập thông thường thành tích chập chiều sâu tích chập chiều sâu (*Depthwise Convolution*) và tích chập điểm 1x1 (*Pointwise Convolution*) [10]. Hình 4 mô tả so sánh nguyên lý hoạt động của phép tích chập thông thường so với phép tích chập tách biệt chiều sâu. Đối với phép tích chập thông thường (hình 4a), mỗi kênh đầu vào yêu cầu một phép tích chập có số lượng nhân tích chập (*convolution kernels*) giống với kênh đầu ra. Kết quả của mỗi kênh đầu ra là tổng của tất cả các nhân tích chập tương ứng và kết quả tích chập của tất cả các kênh đầu vào. Giả sử kích thước đầu vào X là $W_X . H_X . C_{in}$, trong đó W_x, H_x và C_{in} lần

lượt là chiều rộng, chiều cao và số kênh đầu vào tương ứng. Đầu ra Y là $W_Y . H_Y . C_{out}$, trong đó W_Y, H_Y và C_{out} lần lượt là chiều rộng, chiều cao và số kênh đầu ra.



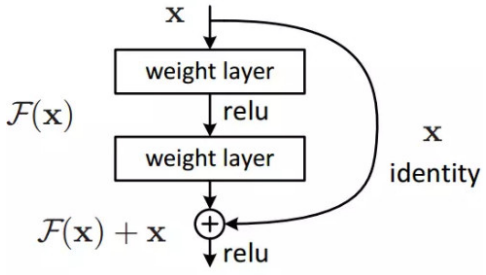
Hình 4. So sánh tích chập thông thường và tích chập tách biệt chiều sâu [11]

So với tích chập thông thường, phép tích tập chiều sâu và tích chập điểm giúp giảm số lượng các trọng số và lượng tính toán đáng kể [11]. Để cùng tạo ra một đầu ra có kích thước $(w.h.c)$ thì tích chập thông thường cần thực hiện $(w.h.c).(k.k.c)$, trong đó $(w.h.c)$ là số lượng pixel cần tính và $(k.k.c)$ là số phép nhân để tạo ra một pixel. Còn đối với phép tích chập tách biệt chiều sâu chỉ phải thực hiện: $(w.h.c).(k.k)$ phép nhân đối với tích chập chiều sâu và $(w.h.c).(w.h)$ phép nhân đối với tích chập điểm. Tỉ lệ các phép tính giữa tích chập thông thường và tích chập chiều sâu như sau:

$$\frac{w.h.c.k.k.c}{(w.h.c).(k.k) + (w.h.c).(w.h)} = \frac{c.k.k}{k.k + w.h} \quad (1)$$

Đây là một tỉ lệ khá lớn cho thấy tích chập chiều sâu tách biệt có chi phí tính toán thấp hơn nhiều so với tích chập thông thường. Do đó phù hợp để áp dụng trên các thiết bị có cấu hình yếu.

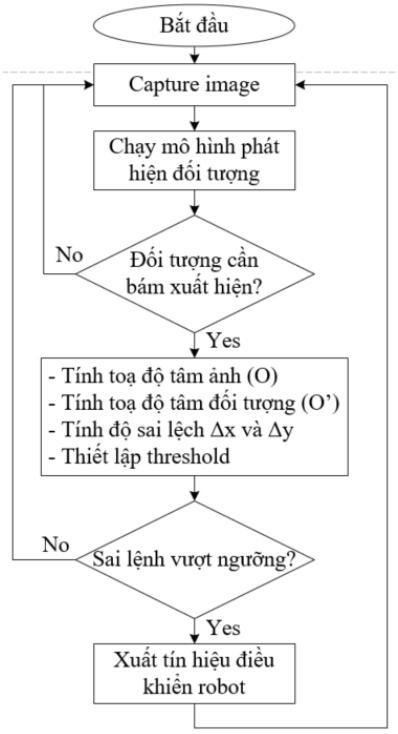
MobileNetV2 tiếp tục sử dụng cơ chế tích chập theo chiều sâu, ngoài ra còn sử dụng những kết nối tắt, tức là các khối ở layer trước được cộng trực tiếp vào layer liền sau. Nếu coi layer liền trước là x, sau khi đi qua các xử lý tích chập hai chiều ta thu được kết quả F(x) thì output cuối cùng là một residual block có giá trị x + F(x) (hình 5).



Hình 5. Kết nối tắt ở MobileNetV2

2.4. Bám đối tượng

Dựa và đối tượng đã được nhận dạng, hệ thống sẽ sử dụng thuật toán đã xây dựng để bám đối tượng và tính toán sai số giữa tâm của đối tượng cần bám đối với tâm chuẩn để đưa ra gợi ý các phương án điều khiển robot. Quá trình hoạt động của phần mềm khi triển khai thuật toán nhận dạng và bám đối tượng vào robot cơ cở thể hiện ở hình 6.



Hình 6. Lưu đồ thuật toán bám đối tượng cho robot

Sau khi nhận dạng được đối tượng mục tiêu. Trong ứng dụng thực tế, để robot luôn bám theo đối tượng mục tiêu, đưa ra yêu cầu của bài toán là chương trình điều khiển của robot phải đáp ứng làm sao cho sai số giữa tâm của đối với tâm chuẩn (tâm của khung hình) luôn nằm trong ngưỡng cho phép theo phương Ox và Oy. Khi đối tượng mục tiêu chuyển động tức là điểm tâm di động thì sẽ xuất hiện sai số Δx theo phương ngang, Δy theo phương dọc so với tâm chuẩn. Bài toán điều khiển robot ở đây là điều khiển robot di chuyển sao cho các sai số Δx và Δy luôn nhỏ hơn ngưỡng cho phép theo phương ngang và phương dọc đã thiết lập.

3. THỰC NGHIỆM, KẾT QUẢ VÀ VÀ THẢO LUẬN

3.1. Dữ liệu thử nghiệm

Trong nghiên cứu này, chúng tôi đánh giá các mô hình phát hiện đối tượng dựa trên hai bộ dự liệu chuẩn là PASCAL VOC [12] và *OpenImages*. Đây là 2 bộ dữ liệu thông dụng, được các nhà khoa học trên toàn thế giới sử dụng để

xây dựng và đánh giá các mô hình trong lĩnh vực phát hiện và nhận dạng đối tượng trong cảnh thực tế. PASCAL VOC gồm 2 bộ dữ liệu là VOC2007 và VOC2012 với 20 lớp đối tượng khác nhau như người, vật, phương tiện giao thông, và các loại đối tượng trong nhà (như chai nước, ghế, tivi, ghế dài - sofa,...). *OpenImages* là bộ dữ liệu lớn được Google phát hành vào năm 2016, là một trong những bộ sưu tập Hình ảnh được gắn nhãn lớn và đa dạng nhất, gồm khoảng 9 triệu hình ảnh với 600 các đối tượng khác nhau. Kể từ đó, Google đã thường xuyên cập nhật và cải thiện bộ dữ liệu này. Phiên bản mới nhất của bộ dữ liệu, Open Images V7, đã được giới thiệu vào năm 2022.

3.2. Tiêu chí đánh giá

Trong bài báo này, chúng tôi sử dụng độ đo Mean Average Precision (mAP) làm tiêu chuẩn đo lường để đánh giá các mô hình phát hiện đối tượng [13]. Việc tính toán mAP cho các mô hình phát hiện đối tượng được thực hiện như sau: đầu tiên, độ chính xác trung bình (AP) được tính toán khi xem xét giao điểm trên liên hợp (IoU) lớn hơn 50% cho mỗi lớp có trong dữ liệu thực tế. Sau đó, tính toán giá trị trung bình của tất cả các giá trị AP trong mỗi lớp. AP lớn đồng nghĩa với việc mô hình có chất lượng phát hiện tốt, khi độ tin cậy và độ nhạy đều cao. Công thức tính độ đo mAP được biểu diễn như sau [13]:

$$\text{mAP} = \frac{1}{n}\sum_{k=1}^{n} \text{AP}_k \quad (2)$$

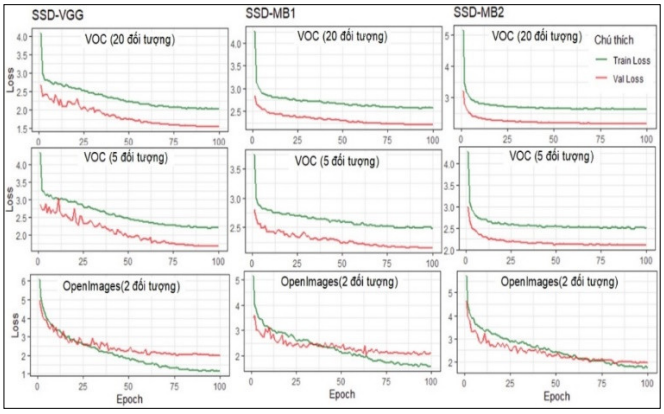
Trong đó, AP_k là Average Precision (AP) của lớp k, n là số lớp đối tượng.

3.3. Thiết lập thử nghiệm

Quá trình xây dựng, thử nghiệm và huấn luyện các mô hình được thực hiện trên hệ thống máy tính YUAN VPP6N0-S-NX sử dụng nền tảng NVIDIA Jetson Orin NX (1,024-core NVIDIA GPU với 32 Tensor Cores, RAM 16GB). Sau khi được huấn luyện các mô hình được sử dụng trên mạch nhúng Jetson Nano (128-cores Nvidia Maxwell GPU, RAM 4GB) để chạy suy luận và đánh giá. Các máy tính nhúng được cài đặt hệ điều hành Ubuntu, chương trình thử nghiệm của chúng tôi được xây dựng trên môi trường Python. Để huấn luyện mạng phát hiện đối tượng, nhiều tham số cấu hình đã được sử dụng để nâng cao hiệu quả. Cụ thể, phương pháp tối ưu hóa *Stochastic Gradient Descent*, kích thước mỗi gói dữ liệu huấn luyện (*batch*) là 32 đối với bộ dữ liệu VOC và *4* đối với bộ dữ liệu OpenImages, tỷ lệ học (learning rate) là 0,01. Ngưỡng của *Intersection over Union* (IoU) là 0,5 để tạo hộp giới hạn tốt nhất. Quá trình huấn luyện mỗi mô hình được thiết lập trên 100 lượt học (*epoch*). Đầu tiên các mô hình được đánh giá trên tập dữ liệu VOC 20 gồm 11540 ảnh huấn luyện (Traning), 5011 ảnh kiểm chứng (Validation) và 4020 để kiểm tra (Test). Tiếp theo chúng tôi giảm độ phức tạp của bộ dữ liệu VOC bằng cách lọc 5 đối tượng từ bộ dữ liệu gốc (dữ liệu sau khi được xử lý bao gồm 8125 ảnh Training, 1578 ảnh Validation và 1339 Test. Cuối cùng là các mô hình được đánh giá trên tập dữ liệu có kích thước nhỏ hơn gồm 2 đối tượng vũ khí (súng và dao) được chúng tôi lựa chọn từ bộ dữ liệu *OpenImages* gồm 1051 ảnh Traning, 72 ảnh Validation và 225 ảnh Test.

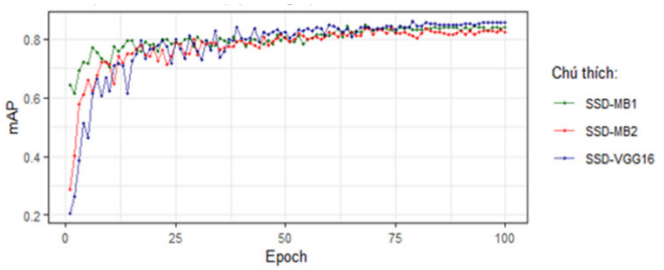
3.4. Kết quả thử nghiệm

Kết quả quá trình huấn luyện SSD-VGG16, SSD-MB1 và SSD-MB2 trên tập dữ liệu VOC (20 đối tượng), VOC (5 đối tượng) và *OpenImages* (2 đối tượng) sau 100 lượt học được thể hiện trong hình 7.



Hình 7. Quá ttrình huấn luyện các mô hình phát hiện đối tượng

Đường màu xanh lá thể hiện sai số trên tập *Train* và màu đỏ trên tập *Validation*. Kết quả huấn luyện cho thấy cả ba mô hình đều có khả năng tổng quát hoá trên các tập dự liệu khác nhau khá tốt. Tuy nhiên, mô hình SSD-VGG16 có độ ổn định thấp hợp so với SSD-MB1 và SSD-MB2. Bên cạnh đó có thể thấy rằng, mô hình SSD-MB1 đạt được sai số nhỏ hơn so với hai mô hình còn lại ngay ngay từ lượt học đầu tiên (điều này cũng thể hiện ở ví dụ trong hình 8: mô hình SSD-MB1 đã đạt độ chính xác trên 60%, còn đối với SSD-VGG16 và SSD-MB2 là dưới 30% trong lượt học đầu tiên).



Hình 8. Độ chính xác trung bình của các mô hình trên tập Validation (OpenImages)

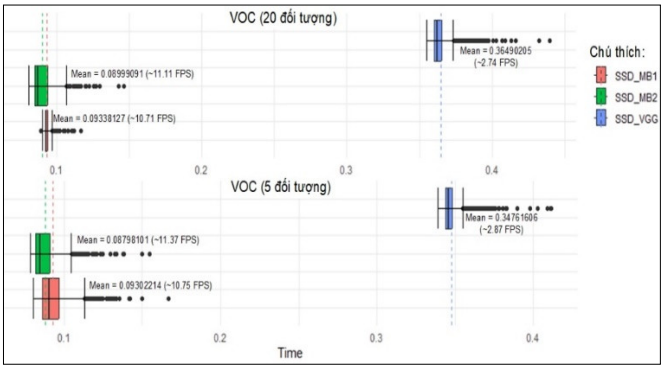
Bảng 1 thể hiện so sánh kết quả của các mô hình trên tập dữ liệu kiểm tra (*Test*) của bộ dữ liệu VOC và OpenImages với các thiết lập khác nhau.

Bảng 1. So sánh độ chính xác giữa các mô hình

Phương pháp	Số lượng tham số (triệu)	VOC (20 đối tượng)		VOC (5 đối tượng)		OpenImages	
		T_huấn luyện (giờ)	mAP (%)	T_huấn luyện (giờ)	mAP (%)	T_huấn luyện (giờ)	mAP (%)
SSD-VGG16	26,3	~ 14,2	80,2	~ 8,0	86,9	~ 3,0	85,5
SSD-MB1	9,4	~ 4,0	73,6	~ 2,8	82,7	~1,5	84,6
SSD-MB2	3,4	~ 4,2	74,2	~ 2,9	83,4	~1,5	82,9

Mạng SSD-VGG16 có số lượng tham số rất lớn (26,3 triệu) nên tốn rất nhiều thời gian huấn luyện (hơn 14 giờ trên tập VOC 20 đối tượng), mạng SSD-MB2 có số lượng tham số ít nhất (3,4 triệu), ít hơn gần 2,8 lần so với mạng SSD-MB1. Tuy nhiên chênh lệch về thời gian huấn luyện giữa hai mô hình này không đáng kể, mô hình SSD-MB1 có thời gian huấn luyện thậm chí còn nhanh hơn so với SSD-MB2.

Kết quả cho thấy, mô hình SSD-VGG16 có độ chính xác cao hơn so với hai mô hình còn lại, tuy nhiên khi kích thước và độ phức tạp của dữ liệu giảm thì chênh lệch độ chính xác giữa ba mô hình không còn đáng kể, cụ thể đối với bộ giữ liệu OpenImages độ chính xác của hai mô hình sử dụng mạng cơ sở MobileNets gần như tương đương so với SSD-VGG16. Mô hình SSD-MB1 có độ chính xác thấp hơn so với hai mô hình còn lại trong tập dữ liệu VOC, tuy nhiên trong tập dữ liệu *OpenImages* nó đạt độ chính xác cao hơn mô hình SSD-MB2. Điều này khẳng định rằng những cải tiến của MobileNetV2 so với MobileNetV1 chưa chắc đã mang lại độ chính xác tốt hơn đối với những bộ dữ liệu có kích thước và độ phức tạp không quá lớn.



Hình 9. Phân phối thời gian thực thi của các mô hình trên 1000 frames ảnh

Xét về độ chính xác SSD-VGG16 cao hơn sao với hai mô hình sử dụng mạng cơ sở là MobileNet, tuy nhiên thời gian thực thi của nó trên máy tính Jetson Nano chậm hơn nhiều. Hình 9 thể hiện biểu đồ Boxplot mô tả phân phối giá trị thời gian thực thi của ba mô hình phát hiện đối tượng trên 1000 frames ảnh từ camera, được thực hiện trên mạch nhúng Jetson Nano. Ở đây chúng tôi hướng tới xây dựng Robot bám theo đối tượng chuyển động là con người, do đó các thử nghiệm tiếp theo về tốc độ sẽ được thực hiện đối với các mô hình được huấn luyện trên bộ dữ liệu VOC. SSD-MB2 có tốc độ thực thi nhanh nhất (khoảng 11,11 FPS đối với mô hình phát hiện 20 đối tượng và 10,71 FPS đối với mô hình phát hiện 5 đối tượng), sau đó tới SSD-MB1 (với 11,71 FPS và 10,75 FPS), cuối cùng là SSD-VGG16 (2,74 FPS và 2,87 FPS). SSD-MB1 (20 đối tượng) thời gian thực thi khá chụm và ít có điểm vọt hơn hai mô hình còn lại.

Kết quả cho thấy trên Jetson Nano, SSD-VGG16 có tốc độ thực thi khá chậm (dưới 3 FPS) vì vậy không phù hợp để triển khai trên Jetson Nano phục vụ bài toán phát hiện và bám đối tượng cho robot thời gian thực. SSD-MB2 là lựa chọn tối ưu nhất đảm bảo sự cân bằng giữa độ chính xác và tốc độ trong trường hợp này, dù phải đánh đổi độ chính xác không đáng kể so với SSD-VGG16 nhưng bù lại thời gian thực thi trên Jetson Nano khá nhanh (~11 FPS). Vì vậy ở bước tiếp theo chúng tôi tiến hành phát triển mô hình SSD-MB2 để phục vụ bài toán điều khiển Robot bám theo đối tượng.

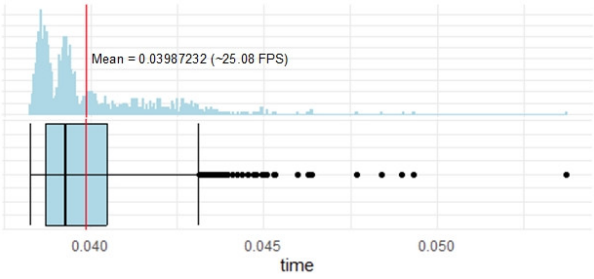
Để triển khai mô hình SSD-MB2 thực thi thời gian thực trên mạch nhúng Jetsnon Nano, mô hình được chúng tôi tối ưu hoá sử dụng công cụ TensorRT [14] với các bước như thể hiện trong hình 1. Sau khi tối ưu, tốc độ thực thi của mô hình trên Jetson Nano đạt ~ 25FPS, nhanh hơn gấp hơn 2 lần so với mô hình SSD-MB2 chạy trực tiếp không qua tối ưu. Phân bố thời gian thực thi trên 1000 frames ảnh đối với mô hình SSD-MB2 tối ưu thể hiện trong hình 10a.

Sau khi phát hiện được vật thể trong khung hình, mô hình sẽ trả về toạ độ các điểm giới hạn dưới dạng các giá trị pixel $(x_{min}, y_{min}), (x_{max}, y_{min}), (x_{min}, y_{max}), (x_{max}, y_{max})$ cho phép xác định hộp giới hạn bao quanh đối tượng và định vị được đối tượng trong khung hình. Từ đó, toạ độ tâm của đối tượng (3.i) và tâm khung hình (3.ii) được tính như sau:

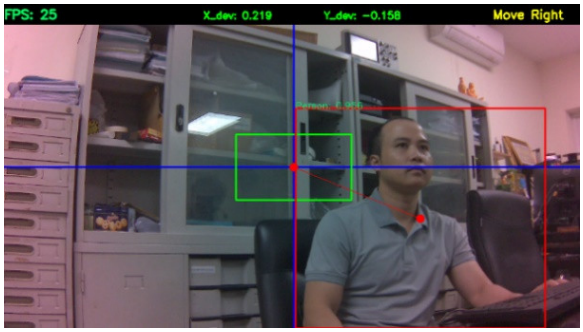
$$\begin{aligned} &(x_{obj}, y_{obj}) = \left(\frac{x_{max}-x_{min}}{2}, \frac{y_{max}-y_{min}}{2}\right) \text{(i)}; \\ &(x_o, y_o) = \left(\frac{img_{rộng}}{2}, \frac{img_{cao}}{2}\right) \text{(ii)} \end{aligned} \quad (3)$$

Sau đó, chúng tôi thiết lập khung ngưỡng giới hạn (màu xanh lá cây thể hiện trong hình 10b). Dựa vào giá trị sai số theo phương ngang Δx và theo phương dọc Δy giữa tâm đối tượng và tâm khung hình so với ngưỡng giới hạn,

chúng tôi có thể đưa ra quyết định xuất các tín hiệu điều khiển robot di chuyển tiến lùi hoặc trái phải khi Δx và Δy vượt các ngưỡng cho phép. Kết quả thực thi phần mềm phát hiện và bám đối tượng minh hoạ trên hình 10b.



a) Phân bố thời gian thực thi trên 1000 frames ảnh



b) Cửa sổ phần mềm phát hiện và bám đối tượng

Hình 10. Kết quả mô hình SSD-MB2 sau khi tối ưu

4. KẾT LUẬN

Mục đích của nghiên cứu này là xây dựng hệ thống phát hiện và bám đối tượng thời gian thực cho thiết bị nhúng với tài nguyên tính toán hạn chế. Kiến trúc SSD nguyên bản sử dụng mạng cơ sở VGG-16 đã được đánh giá so sánh với kiến trúc SSD sửa đổi mạng cơ sở (sử dụng các phiên bản khác nhau của mạng MobileNet) dựa trên độ chính xác và tốc độ xử lý. Kết quả cho thấy các mô hình sử dụng MobileNet làm mạng cơ sở cung cấp sự cân bằng giữa độ chính xác (lớn hơn 80% đối với bô dữ liệu thử nghiệm) và tốc độ ~25 FPS (SSD-MB2 sau khi tối ưu sử dụng công cụ TensorRT), thậm chí đạt đột chính xác tương đương với mạng nơ-ron phức tạp như VGG-16 đối với bộ dữ liệu không lớn.

Trong các nghiên cứu tiếp theo, chúng tôi sẽ hướng tới triển khai phương pháp đề xuất kết hợp với giải thuật điều khiển để thực thi trên robot, đồng thời nghiên cứu cải tiến, tối ưu mô hình cũng như nghiên cứu các phương án sử dụng mạng cơ sở khác.

TÀI LIỆU THAM KHẢO

[1]. He K.M., Gkioxari G., Dollar P., Girshick R., "Mask R-CNN," In *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2980-2988, 2017.

[2]. Kang K., Li H., Yan J., Zeng X., Yang B., Xiao T., Zhang C., Wang Z., Wang R., Wang X., et al., "T-CNN: Tubelets with Convolutional Neural Networks for Object Detection From Videos," I*EEE Trans. Circuits Syst. Video Technol.*, 28, 2896-2907, 2018.

[3]. S. Ren, K. He, R. Girshick, J. Sun, "Faster R-CNN: Towards realtime object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, 39, 6, 1137–1149, 2017.

[4]. J. Dai, Y. Li, K. He, J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. NIPS,* 379-387, 2016.

[5]. J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition,* Las Vegas, USA, pp. 779-788, 2016.

[6]. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, A. C. Berg, "SSD: Single shot multibox detector," in *European Conference on Computer Vision*, Amsterdam, Netherlands, 21-37, 2016.

[7]. Jiang Z., Zhao L., Li S., Jia Y., "Real-time object detection method for embedded devices," In *Computer Vision and Pattern Recognition*, 3, 1-11, 2020.

[8]. Q. Zhao, T. Sheng, Y. Wang, F. Ni, L. Cai, "CFENet: Anaccurate and efficient single-shot object detector for autonomous driving," *CoRR*, arXiv:1806.09790, 2018.

[9]. Choi D., Bell W., Kim D., Kim J., "UAV-driven structural crack detection and location determination using convolutional neural networks," *Sensors*, 21(8), 2650, 2021.

10]. Howard A. G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Adam H., "MobileNets: efficient convolutional neural networks for mobile vision applications," *arXiv preprint* arXiv:1704.04861, 126, 2017.

[11]. Mo Z., Luo D., Wen T., Cheng Y., Li X., "FPGA implementation for odor identification with depthwise separable convolutional neural network," *Sensors,* 21(3), 832, 2021.

[12]. Mark Everingham, Luc J. Van Gool, Christopher K. I. Williams, John M. Winn, Andrew Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision,* 88(2):303-338, 2010.

[13]. Lin Tsung-Yi, et al., "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference*, Zurich, Switzerland, 2014.

[14]. Zhou Y., *Accelerating and Improving Deep Learning Inference on Embedded Platforms via TensorRT*. Doctoral dissertation, Texas State University-San Marcos, 2024.

AUTHORS INFORMATION

Sai Van Cuong[1], Nguyen Van Duc[1], Bui Thi Duyen[2]

[1]Control, Automation in Production and Improvement of Technology Institute (CAPITI), Academy of Military Science and Technology, Vietnam

[2]Electric Power University, Vietnam