Tăng cường độ chính xác trong việc nhận diện đối tượng trên các thiết bị cạnh thông minh

Lê Chí Luận, Tô Hải Thiên

***Tóm tắt*— Nhận diện đối tượng là một trong những chủ đề chính của lĩnh vực AI. Có nhiều mô hình (models) AI được tạo ra với độ chính xác cao, chạy tốt trên các thiết bị có cấu hình cao. Tuy nhiên, các thiết bị cạnh thông minh (Smart Edge Devices - SED) đang được sử dụng rộng rãi trên nhiều lĩnh vực khác nhau bởi tính linh động nhỏ gọn, đảm bảo chính sách dữ liệu của cá nhân. Nhược điểm của các thiết bị cạnh thông minh này chính là giới hạn về phần cứng, chúng chỉ chạy được các model 8bits 16bits hoặc 32bits. Do đó, các mô hình khi chạy trên các SED phải trải qua bước hoán đổi ("quantization"). Điều này cũng khiến các mô hình nhận diện sẽ bị giảm độ chính xác đáng kể. Bài báo này, chúng tôi đề xuất giải thuật tên là "GreedyPlus" - nhằm tìm kiếm các đối tượng nhỏ trong ảnh bằng cách chọn lựa khung ảnh có chất lượng tốt (không bị nhòe). Sau đó, chia các khung hình (frame) thành các ô, rồi phóng to, nhận diện đối tượng trong các ô. Bước cuối cùng là ghép các đối tượng trong các ô để tạo ra kết quả nhận diện tốt nhất. Phương pháp này đơn giản nhưng đạt hiệu quả cao, cải thiện kết quả nhận diện cho mô hình một cách rõ rệt không cần phải training lại mô hình với dataset mới. Kết quả được thực nghiệm trên các bộ dataset KITTI, CrownAI, Autti.**

***Abstract*—Object recognition is one of the main topics in the AI field. There are many AI models with high accuracy running well on high-configuration devices. However, smart edge devices (SED) are being widely used in many different fields because of their compact flexibility, ensuring personal data policy. Their limitation is hardware that only runs or supports 8bits 16bits or 32bits models. Therefore, running the model on SED must do the swap step ("quantization"). This also causes the recognition models to be significantly reduced in accuracy. In this paper, we propose the solution "GreedyPlus" – to capture high resolution frame (skip blur frame) and search for small objects in the image by cutting frames into small windows. Then, the solution zooms in and identifies objects. The last step determines the number of objects in the frame exactly. The method is simple but highly effective, improving the recognition results for the model significantly without the need to retrain the model with a new dataset. The results are tested and demonstrated on the datasets KITTI, CrownAI, and Autti.**

Từ khóa— mô hình AI, thiết bị cạnh, nhận diện đối tượng, thời gian thực

Keywords—DL model; edge device; real time detection; object detection.

I. GIỚI THIỆU

Ngày nay, các ứng dụng AI ngày càng trở nên phổ biến hơn trong cả đời sống hàng ngày và công nghiệp. Một vài năm trở lại đây, có nhiều nghiên cứu tập trung vào các hệ thống ML (machine learning – học máy) và hệ thống DL (deep learning – học sâu) truyền thống. Các hệ thống này sử dụng 32bit hoặc 64bit float trong các mô hình ML hoặc DL. Các mô hình truyền thống phải chạy trên các thiết bị có khả năng tính toán mạnh như máy tính, máy chủ hoặc trung tâm đám mây. Trong Hình 1 chúng tôi đưa ra bảng so sánh giữa một số thiết bị cạnh thông minh (SED) và thiết bị có khả năng tính toán mạnh (máy tính cá nhân). Việc chạy các mô hình ML, DL truyền thống trên các thiết bị có cấu hình cao, khả năng tính toán mạnh vẫn còn một số hạn chế như kích thước và giá thành. Nhiều công ty đã thiết kế và sản xuất các SED

Bài báo được nhận ngày 08/5/2023. Bài báo được nhận xét bởi phản biện thứ nhất vào ngày 02/10/2023 và được chấp nhận đăng vào ngày 09/10/2023. Bài báo được nhận xét bởi phản biện thứ hai vào ngày 02/10/2023 và được chấp nhận đăng vào ngày 05/10/2023.

Thiết bị	Coral Board/Asus Tinker Edge T	Nano Jetson Board	VGAs (Ex: Nvidia Geforce)	Raspi 4
Giá	Coral Dev Board: 130$ Asus Tinker Edge T 179$	60-90	500$-inf + ...	~35$
Kích thước (Dài (mm) * Rộng (mm) * Cao(mm))	88.1 * 59.9 * 22.4 3.37 x 2.12 x 5 inches	100 * 79 * 27.3	~ 280mm*140mm*30mm	
Thông tin nguồn điện	Hoạt động ổn định tại 5V*2.5A = 12.5 W	Hoạt động ổn định tại 5V*4A = 20W (5V,3A phát sinh lỗi)	250W / 180W + PC's Power	5.1V / 3.0A = 15.3W
Low power ARM Core	Cortex M4F	NA	Không	NA
Kích thước bộ nhớ (Memory size)	1GB	4GB	4GB,8GB, 16GB + PC's CPU + PC's RAM	4GB
DL Accelerator	Google Edge TPU 50$ (Accelerator Module:20$) (4TOPs, Int8 only) Có khả năng mở rộng TPU, nhiều TPU thì tốc độ thực thi nhanh	128 CoreMax-well GPU. (472 GFLOPs, Float16) Không thể mở rộng DL Accelerator	Float32,64	CPU can insert to Coral board the devices that support TPU computing power (EdgeTPU)
Hỗ trợ đa luồng	Yếu	NA	Mạnh	Strong
Hỗ trợ thư viện thị giác máy tính -cv libs (Theo đánh giá cá nhân)	Yếu - Có nhiều thư viện không thể cài đặt được bằng pip	NA	Mạnh	Strong
Tính ổn định của hệ điều hành	Yếu (Không hỗ trợ giao diện người dùng)	Mạnh	Rất mạnh	Very strong
Hỗ trợ TensorFlow	o We have to quantize model to suite with Coral board. Until now, Current quantization methods only work well with simple model.	o	o	o
Hỗ trợ Pytorch	x	o Hỗ trợ "chuyển đổi" - quantized model từ model phức tạp từ model phức tạp sang model nhẹ dễ hơn	o	o
Model	Tensorflow Lite	TensorFlow, PyTorch, Caffe/Caffe2, Keras	All	TensorFlow, PyTorch, Caffe/Caffe2, Keras
Hỗ trợ SD Card, USB Camera, USB Micro	o	o	x	o

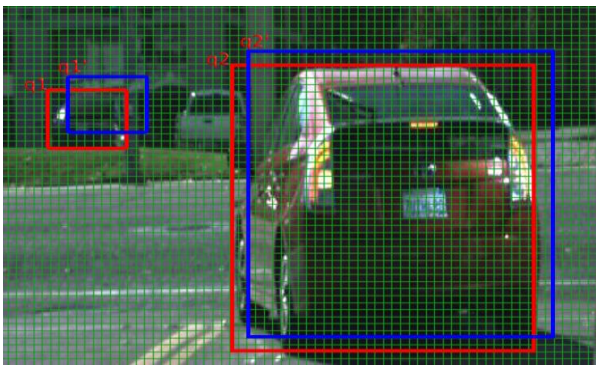
Hình 1. Thông số các SED thông dụng

có thể triển khai các thuật toán AI như máy học (ML) hoặc học sâu (DL). Mặc dù chúng vẫn có một số điểm yếu liên quan đến khả năng tính toán thấp chỉ chạy được các mô hình nhẹ. Tuy nhiên, chúng nhỏ, linh hoạt, dễ di chuyển, tiêu thụ ít năng lượng và rẻ hơn máy tính, máy chủ hoặc trung tâm đám mây. Bên cạnh đó, việc chạy các thuật toán AI ngay trên cục bộ mà không cần kết nối Internet giúp dữ liệu được lưu trữ cục bộ trên thiết bị, bảo toàn và riêng tư. Do đó, việc nghiên cứu các thiết bị cạnh AI đang nhận được sự quan tâm của nhiều nhà nghiên cứu.

Mặc dù có nhiều ưu điểm nổi bật như trên, một mô hình khi chạy trên các SED bắt buộc phải trải qua bước quan trọng là "quantize" – một bước nén mô hình trên các máy tính có cấu hình cao có khả năng chạy trên các thiết bị bit thấp (8 bits hoặc 16 bits). Quá trình này, có thể giảm độ chính xác của mô hình gốc rất nhiều. Do đó, tác giả Kim và các cộng sự [1] đã đưa ra đề xuất tối ưu việc huấn luyện tùy thuộc vào vị trí của vec tơ trọng số (weight vector) để nén mô hình. Trong khi trước đó, Li [2], Krishnamoorthi [3] và các cộng sự tập trung vào việc bắt chước (mimicking) hàm kích hoạt (activation) dựa vào bình quân (mean) và phương sai (variance) để phân bố lại hàm kích hoạt của tập dữ liệu huấn luyện (training dataset). Tuy nhiên, với việc sử dụng mô hình AI chọn bit thấp hơn các tham số phù hợp với sự phân bố trọng số (weight) và độ lệch (biases), làm mô hình trở lên quá mạnh với tập dữ liệu huấn luyện. Tức là khả năng tổng quát của mô hình kém.

Nếu đầu vào là toàn bộ ảnh (chiều rộng x chiều cao), một tập hợp phạm vi nén (quantization) có thể vẽ nguyên bản dưới dạng lưới 255x255 để suy luận 8 bit. Khi đó các đỉnh của khung khoanh vùng đuối tượng có thể điều chỉnh đa dạng để phù hợp với dữ liệu được gán

nhãn Ví dụ: sau khi nén, nếu một giá trị đỉnh chưa được chuyển hoàn toàn đến một giao điểm mới trên lưới, nó sẽ bị buộc quay lại giao điểm trước đó rất nhiều, điều này gây ra sự dịch chuyển trong toàn bộ khung khoanh vùng đối tượng. Một điểm ảnh (pixel) dịch chuyển nhỏ có thể không ảnh hưởng đến các đối tượng lớn, nhưng nó có thể gây ra sự sai lệch cho các đối tượng có kích thước vừa và nhỏ trong ảnh (Hình 2), làm giảm hiệu suất mô hình sau quá trình nén (quantization).

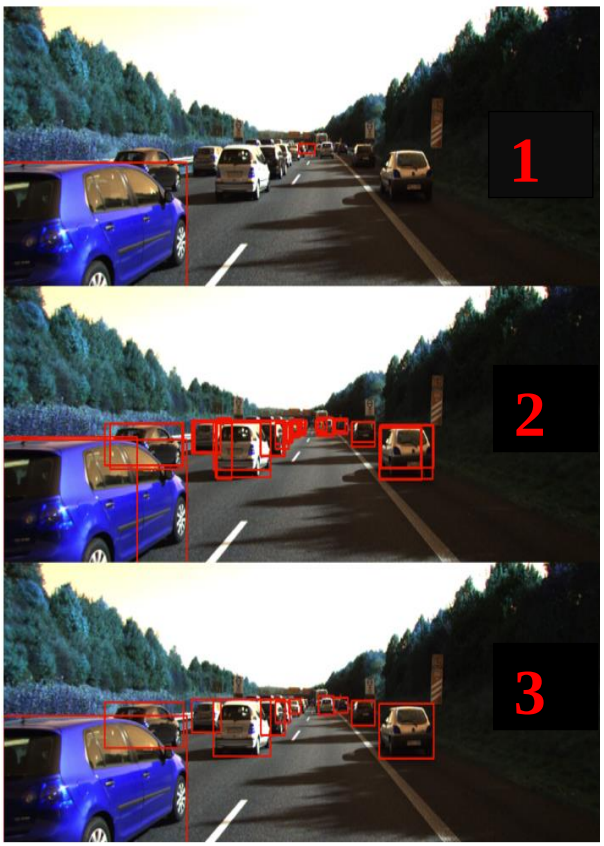


Hình 2. Khung màu đỏ là khung khoanh vùng đối tượng chuẩn, khung màu xanh là khung khoanh vùng đối tượng của mô hình sau khi nén

Chính vì thế một phương pháp mới được gọi là GreedyPlus được đề xuất, giải quyết các vấn đề trên một cách linh hoạt trong phạm vi duy trì sự tổng quát hóa tốt cho mô hình và tích hợp dễ dàng trên SED. Trong GreedyPlus, chúng tôi lấy đầu ra của mô hình được đào tạo để khai thác mức độ hiệu quả của mô hình, khám phá mẫu dữ liệu ẩn và hạn chế của nó khi tổng quát hóa. Sau đó, phương pháp sẽ thực hiện chính sách cửa sổ trượt tham lam (GreedyPlus) để quét qua hình ảnh thu được các khung, vị trí đỉnh của khung trong hình ảnh và so sánh với các kết quả đầu ra hiện tại để chọn các khung chung cho toàn bộ hình ảnh. Chi tiết về chính sách sẽ được giải thích trong phần III. Bằng cách quét qua hình ảnh thay vì sử dụng một hình ảnh duy nhất để phát hiện, phương pháp nhấn mạnh kích thước tốt hơn cho các đối tượng có quy mô thấp hơn và tạo phạm vi phù hợp phù hợp với kích thước đào tạo của mô hình trong quá trình nén. Do đó, nó có thể bỏ qua việc nén quá khớp(overfitting) khi giảm suy luận bit.

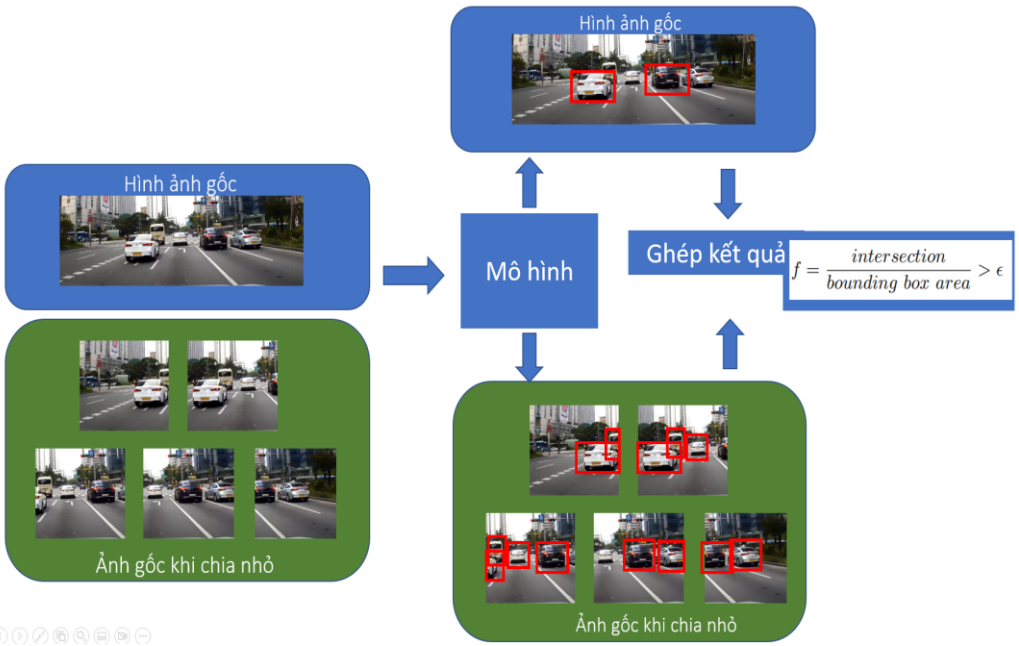
II . CÁC NGHIÊN CỨU LIÊN QUAN

Các phương pháp hiện có cho xác định đối tượng sử dụng CNNs có thể chia thành: 2 tầng xác định hoặc 1 tầng xác định. Các phương pháp 2 tầng như FasterRCNN, R-FCN của các tác giả Ren cùng cộng sự [4], Dai cùng cộng sự [5], Lin cùng cộng sự [6] sẽ khoanh vùng và phân loại đối tượng qua tầng thứ nhất: đề bạt vùng đối tượng (region proposal). Trong khi đó các phương pháp 1 tầng như Yolo, SSD-Mobilenet của các tác giả

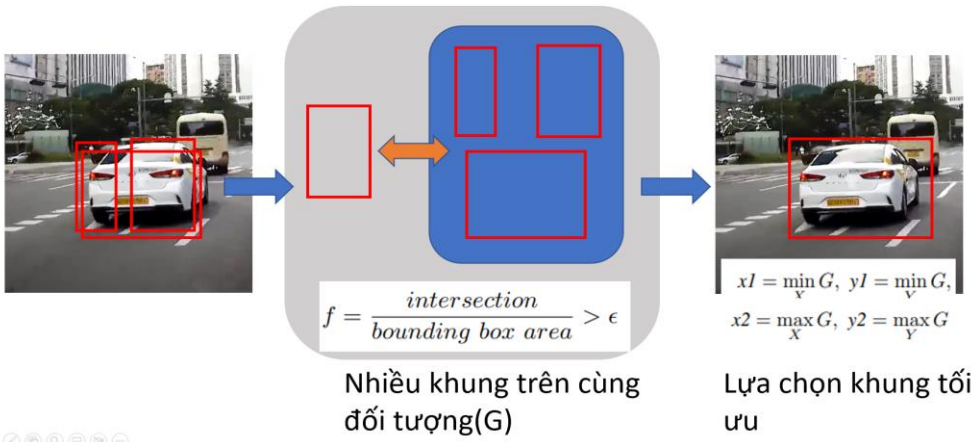


Hình 3. Lựa chọn vùng diện tích phủ lớn nhất (Greedy Selection)

Redmon cùng cộng sự [7], Liu cùng cộng sự [8] loại bỏ tầng trên và thực hiện thẳng tiên đoán và phân loại. Tổng quan, sử dụng 1 tầng sẽ đem lại ưu điểm về tốc độ nhưng sẽ đem lại kết quả với độ chính xác thấp hơn khi sử dụng 2 tầng. Tuy nhiên, nếu đánh giá độ chính xác dựa trên độ so khớp (intersection of union (IoU) = 0.5), thì kết quả thu được trên 1 tầng gần như ngang bằng với 2 tầng. Vì vậy, các nghiên cứu của Sandler [9] hay Howard [10] đã tập trung tối ưu phân trích



Hình 4. Ảnh gốc được chia ra thành các ô ảnh nhỏ hơn. Kết hợp các đối tượng được nhận diện trên hình ảnh gốc và các ô ảnh nhỏ hơn



Hình 5. Miêu tả trực quan việc chọn khung tối ưu cho đối tượng

xuất đặc trưng nhỏ hơn cho phương pháp 1 tầng để đảm bảo ứng dụng chạy tốc độ thực với độ chính xác cao trên các thiết bị biên

Tuy nhiên, mặc dù đạt tốc độ cao, kiến trúc 1 tầng tối ưu trên vẫn gặp phải rất nhiều vấn đề với đối tượng nhỏ do đặc điểm nhận dạng của chúng thường biến mất ở các lớp CNN cuối cùng. Chính vì vấn đề này, các phương pháp phổ thông như phóng to ảnh để tăng số lượng điểm ảnh đã được hướng tới cũng như thêm vào các lớp đặc trưng của những tầng đầu để cải thiện khả năng khoanh vùng và nhận diện. Trong khi tăng kích cỡ ảnh sẽ ảnh hưởng tới tốc độ xử lý và tham số, sử dụng đa tầng đặc trưng sẽ khiến cho mạng nhận diện trở thành mạng 2 tầng. Do đó, mạng 1 tầng gặp rất nhiều vấn đề với đối tượng nhỏ và khi thực hiện các bước nén để hoạt động trên thiết bị biên khiến chúng trở thành một thách thức. Phương pháp trong bài báo, chúng tôi kết hợp hai ý tưởng kể trên. Thay vì tăng kích thước ảnh đầu vào

huấn luyện, phương pháp sử dụng cửa sổ trượt tập trung vào khu vực có nhiều đối tượng nhỏ nhằm phóng to đối tượng

III. GIẢI THUẬT GREEDPLUS

Như đã đề cập ở Phần II, phương pháp đề xuất tập trung vào tối ưu cho mô hình sau khi được huấn luyện hơn là tích hợp vào trong quá trình huấn luyện. Sau quá trình huấn luyện, thuật toán cho phép ứng dụng với nhiều loại mô hình khác nhau với nhiều dải bit khác nhau, phù hợp cho tính phổ quan của mô hình. Để thực hiện thuật toán GreedyPlus, chúng tôi chia thành 3 bước chính: xác định đối tượng qua cửa sổ trượt, nén/khắc chế số lượng ô, và lựa chọn ô tối ưu như mô tả tại Hình 3 (ảnh 1 là hình ảnh thu được khi chạy SSD-MobilenetV2 được nén (lượng tử hóa) có dấu hiệu lệch ô do giá trị sau khi nén bị xê dịch. Ảnh số 2 tại Hình 3 là các ô thu được sau khi cho cửa sổ trượt chạy qua và chia nhỏ khung hình ở các khu vực tiềm tàng đối tượng nhỏ. Ảnh số 3 là kết quả thu được sau khi gom các ô và thực hiện lựa chọn vùng diện tích phủ tốt nhất).

Phương pháp sử dụng thuật toán GreedyPlus được mô tả gồm 3 bước:

Thuật toán GreedyPlus.

Đầu vào: khung ảnh được cắt ra từ camera.

Đầu ra: khoanh đúng danh sách các đối tượng trong khung ảnh

Bắt đầu

Bước 1: Chọn lựa khung ảnh có chất lượng ảnh cao (không bị nhòe)

Bước 2: Xác định đối tượng qua cửa sổ trượt.

Bước 3: Khắc chế số lượng ô cho đối tượng

Bước 4: lựa chọn ô/vùng khoanh tối tối ưu

Kết thúc

Các bước được thể hiện trực quan thông qua Hình 4, 5 và được giải thích chi tiết trong các mục A,B,C,D của phần III.

A. Chọn lựa khung ảnh có chất lượng ảnh cao (không bị nhòe)

Việc nhận diện đối tượng thường bị ảnh hưởng bởi các yếu tố như là ảnh mờ, ảnh tối và không đủ sáng. Để giải quyết vấn đề này, với mỗi khung ảnh, giải pháp đề xuất sử dụng các bộ lọc ảnh nhằm đánh giá chất lượng của ảnh. Chúng tôi sử dụng các thư viện xử lý ảnh như opencv để loại bỏ ảnh mờ và nhòe dễ dàng hơn. Ngoài ra opencv giúp các khung ảnh sáng hơn tùy thuộc vào thời gian trong ngày.

B. Xác định đối tượng qua của sổ trượt

Thay vì sử dụng toàn bộ hình ảnh, bước đầu của thuật toán là chia nhỏ khung hình thành cả các cửa sổ W có kích thước (l,l) nằm chồng lên nhau và kết hợp với hình ảnh gốc I. Có hai bước chính cho phương pháp này. Thứ nhất, bằng việc chồng các cửa sổ lên nhau với tỉ lệ chồng gamma-δ, chúng ta có thể tránh được các vật thể trung bình và lớn bị xé nhỏ. Vì mỗi cửa sổ được đưa qua cùng một mạng lưới CNN nên chúng sẽ được xác định mà không cần phân tích từ hình ảnh chính. Những đối tượng được tìm sẽ được coi như những kết quả gốc cho mô hình. Bước thứ hai, độ dài của cửa sổ nên có kích thước tương đương với hình ảnh gốc được huấn luyện, từ đó giúp quá trình co dãn ảnh được ổn định và phù hợp với kiến thức của các mô hình đã được lượng tử hóa. Việc chia nhỏ ảnh tránh các hiện tượng xê dịch ô cũng như đảm bảo mô hình được vận hành tốt nhất với nhận thức huấn luyện

$$SLW\ (W,I,l,\ \delta) = W',I' \quad (1)$$

Phương pháp này tuy đơn giản nhưng cho phép hình ảnh đầu vào được đánh giá nhiều lần và kĩ lưỡng với nhiều độ phân giải khác nhau. Do chia nhỏ vùng ảnh trước khi tiên đoán vùng đối tượng, hình ảnh của đối tượng sẽ rõ để biểu hiện cho khoảng bit thấp hơn cũng như tập trung cho các đối tượng nhỏ tốt hơn.

C. Khắc chế số lượng ô cho đối tượng

Bước này được gọi là khắc chế số lượng là do sẽ có rất nhiều vùng ảnh nhỏ được hình thành khi thực hiện cửa sổ trượt và các mảnh của hình

ảnh đối tượng lớn có thể bị xé nhỏ. Đặc tính này có thể được đánh giá tương đương khi một vật có thể được biểu diễn và bao bởi lưới bao quanh (anchor grid) với nhiều độ phân giải khác nhau. Tuy nhiên khác với thuật toán Non-maxima Suppression [11], phương án khắc chế số lượng ở đây nhằm ghép các ô liên đới một vật thể thành ô rộng nhất bao quanh. Để khắc chế, tại bước này, các ô sẽ được đánh giá dựa trên độ phủ của các ô lên cùng một vị trí và so sánh với các ô ở hình ảnh gốc (chưa bị chia ra thành các ô). Các ô nhỏ sẽ được đánh giá là hiếm có thể xuất hiện và được bao chính xác ở hình ảnh gốc và chỉ có thể thu được từ các vùng ảnh cửa sổ. Trong khi đó, các vật lớn sẽ có xu hướng bị chia nhỏ do các ô cửa sổ không thể bao trọn hoặc nằm trong vùng giao giữa các cửa sổ.

Dựa vào giả thiết trên, GreedyPlus nhận vào 2 tham số để khắc chế số ô này. Alpha được dùng để lựa chọn ô cho từng cửa sổ nhỏ và beta là tham số để lựa chọn cho hình ảnh gốc. Thông thường, hai ô sẽ được đánh giá là cùng một vật nếu hệ số giao IoU cao (Intersection over Union là chỉ số đánh giá được sử dụng để đo độ chính xác của Object detector trên tập dữ liệu cụ thể). Tuy nhiên, do các ô lấy từ cửa sổ có kích thước bao một phần của vật thể hơn nên chúng sẽ thường nhỏ hơn ô thực tế của vật, phương pháp sử dụng phép nội giao (self-intersection) để đánh giá bao nhiêu phần trăm một ô hiện tại lệ thuộc ô từ hình ảnh gốc. Phép nội giao được tính bởi diện tích giao giữa 2 ô chia cho chính các ô (2).

$$f = \frac{\text{intersection}}{\text{bounding box area}} > epsilon \qquad (2)$$

Ký hiệu intersection là diện tích giao của hai ô và bounding box area là tổng diện tích hai ô. Để tối ưu tính toán, ta đưa phép nội giao giữa các ô về dạng ma trận. Mỗi hàng của ma trận là phép tự giao f của một đối tượng với các ô ở hình ảnh gốc. Các ô được đánh giá cùng thuộc một đối tượng nếu có chung cột và giá trị lớn hơn epsilon. Chọn epsilon là quan trọng vì giá trị nhỏ có thể khiến cho việc khắc chế không tối ưu và để lại nhiều ô lặp trong khi đó quá lớn sẽ khó hợp nhất các ô.

D. Lựa chọn ô tối ưu

Sau khi loại bỏ các ô nhỏ có tỉ lệ cùng là một vật thể lớn cao, sẽ vẫn còn các ô khác có kích thước vừa và nhỏ nằm rải rác trong khung hình do không có ô để nội giao với ô ở cảnh toàn. Do đó, trong bước cuối, GreedyPlus sẽ thực hiện so sánh giữa các ô này với nhau. Các ô cùng nhắm tới một vật sẽ được nhóm lại thành một nhóm G đại diện cho một đối tượng và là giải pháp cuối cùng để loại bỏ một vật có thể bị lặp nhiều ô.

Từ mỗi nhóm ô G, GreedyPlus sẽ chọn một ô đại diện bằng các chọn tọa độ đỉnh cao nhất và tọa độ đáy xa nhất và có độ tin cậy được đánh giá là trung bình của các ô. Lí do bước này gọi là tối ưu là vì nó sẽ lấy vật thể lớn nhất có thể từ nhóm các ô và tránh lấy chỉ một phần của đối tượng dựa theo độ tin cậy thông thường.

$$x1 = min_X\, G\,, y1 = min_Y\, G,$$

$$x2 = max_X\, G\,, y2 = max_Y\, G$$

$$c = \frac{1}{\mathrm{N}} \sum_G c_g \qquad (3)$$

IV. THỰC NGHIỆM

A. Cài đặt môi trường đánh giá (Benchmark settings)

Mục tiêu của phương pháp là cung cấp một phương pháp có thể giúp cải thiện hiệu suất phát hiện đối tượng cho các mô hình đã được biến đổi thích hợp chạy trên các thiết bị biên và duy trì tổng quát hóa tốt cho các mô hình đó. Do đó, phương pháp đã thực hiện hai thực nghiệm để xác minh hiệu suất của phương pháp: chạy thực nghiệm so sánh phương pháp của chúng tôi với các phương pháp khác trên cùng một dữ liệu (cùng một dataset) và trên nhiều dữ liệu khác nhau (trên nhiều dataset). Chúng tôi chọn Google Coral Board (chỉ hỗ trợ int8) và Jetson Nano (cho FP16 và FP32) là hai thiết bị biên phổ biến nhất hiện nay cho các thử nghiệm. Lưu ý rằng, với việc chạy thực nghiệm trên các tập dữ

BẢNG 1. COCO BENCHMARK CỦA HAI MÔ HÌNH (SSD-MOBILENETV2, RETINA) VỚI PHIÊN BẢN GREEDY CỦA NÓ VỚI ĐỘ CHÍNH XÁC KHÁC NHAU

	300x300-80 đối tượng-Int8		640x640-80 đối tượng-Fp16		640x640-80 đối tượng-Fp32	
Mô hình / Ma trận	Mobinet SSD	Greedy Mobile SSD	Retina	Greedy Retina	Retina	Greedy Retina
AP(0.5) - (S,M,L)	0.21	**0.24**	0.35	**0.41**	0.39	**0.41**
AP(0.5) - S	0.02	**0.06**	0.11	**0.22**	0.14	**0.21**
AP(0.5) - M	0.13	**0.25**	0.39	**0.45**	0.44	**0.46**
AP(0.5) - L	**0.45**	0.42	**0.55**	0.54	**0.55**	0.54
AR(0.5) - (S,M,L)	0.21	**0.26**	0.37	**0.46**	0.41	**0.45**
AR(0.5) - S	0.02	**0.06**	0.11	**0.23**	0.15	**0.23**
AR(0.5) - M	0.14	**0.27**	0.41	**0.51**	0.47	**0.51**
AR(0.5) - L	**0.49**	0.47	0.60	**0.62**	0.60	**0.62**
AP(0.5:0.95) - (S,M,L)	0.14	**0.16**	0.23	**0.26**	**0.28**	**0.28**
AP(0.5:0.95) - S	0.01	**0.03**	0.07	**0.13**	0.09	**0.13**
AP(0.5:0.95) - M	0.07	**0.16**	0.25	**0.29**	0.30	**0.31**
AP(0.5:0.95) - L	**0.31**	0.29	**0.39**	0.38	**0.41**	0.39
AR(0.5:0.95) - S	0.01	**0.03**	0.07	**0.14**	0.10	**0.14**
AR(0.5:0.95) - M	0.08	**0.18**	0.29	**0.35**	0.34	**0.36**
AR(0.5:0.95) - L	**0.36**	0.35	**0.46**	**0.46**	**0.48**	0.47
Tổng điểm	4/15	11/15	3/15	13/15	3/15	12/15

BẢNG 2. COCO VỚI TẬP ĐỐI TƯỢNG LIÊN QUAN ĐẾN GIAO THÔNG. N: NON-GREEDYPLUS. G: GREEDYPLUS

Đối tượng / Ma trận	Người		Xe hơi		Xe tải		xe buýt		Xe máy		Xe đạp	
	N	G	N	G	N	G	N	G	N	G	N	G
SSD-Mobilenet V2												
AP(0.5) - (S,M,L)	0.37	**0.43**	0.16	**0.29**	0.23	**0.27**	0.50	**0.57**	0.34	**0.4**	0.19	**0.30**
AP(0.5:0.95) - M	0.18	**0.28**	0.11	**0.28**	0.03	**0.13**	0.03	**0.21**	0.05	**0.14**	0.08	**0.19**
AP(0.5:0.95) - L	**0.53**	0.51	**0.49**	0.45	**0.40**	0.37	0.65	0.62	**0.47**	0.44	0.47	**0.49**
AR(0.5:0.95) - M	0.21	**0.31**	0.15	**0.34**	0.04	**0.17**	0.03	**0.23**	0.07	**0.18**	0.09	**0.22**
AR(0.5:0.95) - L	**0.60**	0.59	**0.57**	0.53	**0.46**	0.43	**0.69**	0.67	**0.53**	0.50	0.54	**0.56**
Overall Score	2/5	3/5	2/5	3/5	2/5	3/5	1/5	3/5	2/5	3/5	0/5	5/5
Retina												
AP(0.5) - (S,M,L)	**0.57**	**0.57**	0.43	**0.46**	0.31	**0.37**	0.64	**0.68**	0.52	**0.53**	0.38	**0.40**
AP(0.5:0.95) - S	0.15	**0.18**	0.17	**0.20**	0.05	**0.09**	0.14	**0.24**	0.11	**0.15**	0.06	**0.11**
AP(0.5:0.95) - M	**0.47**	0.46	**0.46**	0.45	0.16	**0.25**	0.37	**0.40**	**0.25**	0.24	**0.29**	0.28
AP(0.5:0.95) - L	**0.67**	0.61	**0.54**	0.48	**0.42**	0.37	**0.72**	0.68	**0.52**	0.51	**0.52**	0.51
AR(0.5:0.95) - S	0.17	**0.20**	0.18	**0.22**	0.06	**0.12**	0.15	**0.28**	0.11	**0.18**	0.06	**0.12**
AR(0.5:0.95) - M	**0.52**	0.50	**0.53**	**0.53**	0.21	**0.33**	0.39	**0.44**	**0.32**	0.29	**0.33**	**0.33**
AR(0.5:0.95) - L	**0.73**	0.69	**0.62**	0.58	**0.51**	0.50	**0.74**	0.73	**0.57**	0.55	**0.58**	**0.58**
Tổng điểm	5/7	3/7	4/7	4/7	2/7	4/7	2/7	5/7	4/7	3/7	4/7	5/7

liệu khác nhau, chúng tôi nhấn mạnh tầm quan trọng của phương pháp của chúng tôi là mô hình AI chạy trên môi trường hỗ trợ các dải bit thấp.

Lựa chọn mô hình và tập dữ liệu: Để kiểm tra hiệu suất phát hiện, chúng tôi đưa GreedyPlus so sánh với các mô hình COCO: SSD-MobilenetV2 (6M tham số) và mô hình Retina (32M tham số) trên tập dữ liệu COCO. Đây là những mô hình có những kiến trúc tốt nhất về khả năng phát hiện đối tượng cạnh với khả năng nhanh và nhẹ. Hơn nữa, việc so sánh này cũng nhấn mạnh tính linh hoạt trong các phương pháp tiếp cận của chúng tôi.

Ngoài COCO, tập dữ liệu chúng tôi sử dụng là: KITTI, CrownAI và Autti, để làm tiêu chuẩn xuyên suốt khi đưa các bộ phát hiện chung từ bộ

BẢNG 3. KẾT QUẢ MAP GIỮA GREEDY-SSD SO VỚI SSD-MOBILENETV2 VÀ ĐƯỢC ÁP DUNG TRÊN TẬP DỮ LIÊU KITTI, CROWNAI VÀ AUTTI

Mô hình	dễ (0.5)	trung bình (0.5)	khó (0.5)	dễ (0.7)	trung bình (0.7)	khó (0.7)
KITTI						
Greedy-SSD	**0.5425**	**0.6421***	**0.5924**	**0.5162**	**0.5244**	**0.4204**
SSD-MobilenetV2	0.1632	0.1519	0.1517	0.1566	0.1176	0.1191
CrownAI						
Greedy-SSD	**0.6791***	**-**	**0.5672**	**0.5244**	**-**	**0.4019**
SSD-MobilenetV2	0.1877	-	0.1549	0.1172	-	0.0943
Autti						
Greedy-SSD	**0.6123***	**-**	**0.5772**	**0.5342**	**-**	**0.4446**
SSD-MobilenetV2	0.1784	-	0.1612	0.1123	-	0.1003

dữ liệu COCO vào nhiệm vụ phạm vi hẹp như (phát hiện xe / phát hiện người đi bộ). Đối với các yêu cầu của tập dữ liệu KITTI, chúng tôi theo dõi đánh giá của họ về ba mức độ khó: dễ (to và rõ), trung bình (kích thước trung bình / hơi che khuất) và khó (nhỏ và che khuất nhiều) với tỷ lệ IoU khác nhau. Để áp dụng cài đặt đó vào CrownAI và Autti, chúng tôi xác định mức dễ cho các đối tượng có độ rộng hoặc dài lớn hơn 100pixel và được đánh giá là khó nếu có độ phủ nhỏ hơn 100 pixel.

Lựa chọn siêu tham số (Hyperparameters Selection): khi đánh giá trên tập dữ liệu KITTI, chúng tôi đã tìm ra: kích thước lý tưởng ***k*** cho cửa sổ trượt là khoảng 1/3 chiều rộng của hình ảnh; Độ tin cậy cho mỗi cửa sổ phải là 0,4 và gamma -ẟ cho toàn bộ hình ảnh là 0,7 để đạt độ chính xác cao nhất cho việc phát hiện. Ngoài ra, lý do chúng tôi chọn 0,7 là để cân bằng giữa 0,6 và 0,8 kết quả để tổng quát hóa. Hơn nữa, cài đặt này cũng cho phép phương pháp của chúng tôi bắt kịp tốc độ 70 % của các mô hình gốc. Do đó, chúng tôi sử dụng cấu hình này trong suốt các thí nghiệm của mình. Chuẩn của kết quả phát hiện được tiến hành và đo mAP ở định dạng Pascal VOC.

B. Hiệu suất trên cùng tập dữ liệu COCO

Trong cài đặt này, chúng tôi thực hiện đánh giá xuyên suốt trên mô hình tập dữ liệu COCO được đào tạo trước được xuất bản trong kho lưu trữ mở của Google Coral và Jetson Nano. Chúng tôi bỏ qua cài đặt SSD-Mobinet v2 thử nghiệm trong Jetson Nano do kết quả tương tự trên Coralboard. Chúng tôi cũng loại bỏ tiêu chuẩn của mô hình Retina trên Google Coral vì tốc độ xử lý chậm trên thiết bị này.

Nhìn chung, kết quả tại Bảng 1 (COCO điểm chuẩn cho phép khoảng giá trị của IoU từ 0,5 đến 0,95 và ước tính hiệu suất trên khả năng thu hồi (recall) và độ chính xác (precision). SSD-Mobilenetv2 được thực hiện trên Google Coral Board, trong khi Retina được đo trên Jetson-Nano) cho thấy hiệu suất tăng lên cho mô hình biên bất kể lượng tử hóa int8, giảm xuống FP16 hay giữ nguyên định dạng 32bit trên cả Jetson và Coralboard. Tuy nhiên, phương pháp làm mất đi một chút hiệu suất đối với các đối tượng lớn nhưng lại cải thiện mạnh hiệu quả của các đối tượng nhỏ và kích thước trung bình. Điều này có thể hiểu là chúng ta phân mảnh bức tranh để giải quyết, làm cho các đối tượng nhỏ trở nên chắc chắn hơn nhưng lại vô tình chia các đối tượng lớn thành nhiều mảnh.

Khi kiểm tra một số lớp cụ thể trong Bảng 2, chúng tôi có thể nhận thấy rằng tồn tại lớp có thể duy trì khả năng phát hiện tốt ở kích thước lớn như *"xe máy"* hoặc *"xe buýt"*, trong khi các lớp khác có thể giảm chất lượng. Tuy nhiên, kết quả cho thấy hứa hẹn về hiệu suất độ chính xác khi thu thập các điểm tốt giữa AP (độ chính xác trung bình) và AR (độ thu hồi trung bình) mà GreedyPlus tăng cường mô hình gốc (gần gấp ba lần hiệu suất ~ 12/15 cho các lớp tổng thể và 5/7 trên các lớp cụ thể). Ngoài ra, nó cũng cho thấy

tính linh hoạt của phương pháp của chúng tôi trên các loại mô hình khác nhau (SSD và Retina) và cấu hình biên. Nó cũng chứng minh rằng phương pháp của chúng tôi có thể hỗ trợ mô hình ở nhiều dải bit cấu hình khác nhau.

C. Hiệu suất phát hiện tập dữ liệu bên ngoài

Trong thí nghiệm này này, chúng tôi chỉ giải quyết SSD-Mobilenet như được giới thiệu trong thiết lập ở mục A. Bảng 3 tóm tắt kết quả đánh giá từ ba tập dữ liệu. Như được hiển thị, Greedy-SSD đã hoạt động tốt hơn so với mô hình gốc. Độ chính xác của mAP tăng khoảng năm lần. SSD-MobilenetV2, thông qua lượng tử hóa, đã giảm đáng kể độ chính xác của nó xuống sáu lần so với kết quả gốc trên đánh giá dữ liệu mở KITTI (từ 0,61). Hơn nữa, Greedy-SSD đảm bảo hoạt động ổn định hơn bất kể quy mô đối tượng trong hình ảnh khi khoảng cách giữa tác vụ dễ và khó là gần như tương đương. Do đó, một trong những yếu tố quan trọng nhất đối với phương pháp này là nó có thể giúp mô hình gốc SSD-MobilenetV2 tổng quát hóa kết quả tốt hơn mà không cần thiết kế lại cấu trúc mô hình học sâu mới và cải thiện lượng tử hóa như các cách tiếp cận khác.

IV. KẾT LUẬN

Việc đào tạo các mô hình có thể chạy trên các thiết bị biên là do tính bảo mật. Khi AI được chạy trên thiết bị cạnh, dữ liệu và thông tin nhạy cảm không cần phải được truyền qua mạng hoặc lưu trữ trên các máy chủ trung tâm. Điều này giúp giảm nguy cơ bị tấn công mạng và rò rỉ dữ liệu.

Ngoài ra, chạy AI trên thiết bị cạnh cho phép người dùng có quyền kiểm soát hoàn toàn dữ liệu của họ. Thay vì gửi dữ liệu cá nhân đến các dịch vụ đám mây, người dùng có thể giữ dữ liệu trên thiết bị của họ và quyết định nếu muốn chia sẻ, điều này bảo vệ quyền riêng tư của người dùng.

Trong nghiên cứu này, chúng tôi đã giới thiệu phương pháp GreedyPlus về việc tối ưu hóa hiệu suất phát hiện trên nhiều thiết bị biên. Phương pháp chứng minh rằng khoảng cách về hiệu suất phát hiện đối tượng có và không có GreedyPlus là đáng kể và đảm bảo khả năng tổng quát hóa được cải thiện tốt hơn. Sử dụng GreedyPlus có thể giảm độ phức tạp và nỗ lực tối ưu hóa mô hình bằng tối ưu lượng tử hóa và thay đổi cấu trúc mô hình. Trong tương lai, chúng tôi sẽ cố gắng tích hợp phương pháp này vào quá trình huấn luyện mô hình tính năng đa quy mô để có một giải pháp tốt hơn và nhỏ gọn hơn cho thiết bị biên.

LỜI CẢM ƠN

Nhóm tác giả xin chân thành cảm ơn Trường Đại học Công nghệ Giao thông vận tải đã hỗ trợ kinh phí và trang thiết bị thí nghiệm để thực hiện công trình nghiên cứu này.

TÀI LIỆU THAM KHẢO

[1]. Kim, Jangho, KiYoon Yoo, and Nojun Kwak. "Position-based scaled gradient for model quantization and pruning." Advances in Neural Information Processing Systems 33 (2020): 20415-20426.

[2]. Li, Rundong, et al. "Fully quantized network for object detection." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

[3]. Krishnamoorthi, Raghuraman. "Quantizing deep convolutional networks for efficient inference: A whitepaper." arXiv preprint arXiv:1806.08342 (2018).

[4]. Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems 28 (2015).

[5]. Dai, Jifeng, Yi Li, Kaiming He, and Jian Sun. "R-fcn: Object detection via region-based fully convolutional networks." Advances in neural information processing systems 29 (2016).

[6]. Lin, Tsung-Yi, et al. "Focal loss for dense object detection." Proceedings of the IEEE international conference on computer vision. 2017.

[7]. Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).

[8]. Liu, Wei, et al. "Ssd: Single shot multibox detector." Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016.

[9]. Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen.

"Mobilenetv2: Inverted residuals and linear bottlenecks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520. 2018.

[10]. Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).

[11]. Thuật toán Non maximum suppression https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/.

SƠ LƯỢC VỀ TÁC GIẢ

Lê Chí Luận

Đơn vị công tác: Đại học Công nghệ Giao thông vận tải

Email: luanlc@utt.edu.vn

Quá trình đào tạo: Tốt nghiệp đại học năm 2002 tại trường Đại học Sư phạm Hà Nội ngành tin học. Tốt nghiệp thạc sỹ ngành Công nghệ thông tin tại đại học Công nghệ - Đại học Quốc Gia Hà Nội năm 2009. Năm 2018 hoàn thành luận án tiến sĩ ngành Công nghệ thông tin tại Đại học Công nghệ - Đại học Quốc Gia Hà Nội.

Hướng nghiên cứu hiện nay: Kiểm thử phần mềm, kiểm chứng phần mềm, học sâu, học máy, trí tuệ nhân tạo.

Tô Hải Thiên

Đơn vị công tác: Đại học Công nghệ giao thông vận tải

Email: thienth@utt.edu.vn

Quá trình đào tạo: Tốt nghiệp đại học năm 2010 tại trường Đại học sư phạm kỹ thuật Hưng Yên ngành phần mềm. Tốt nghiệp Đại học tổng hợp Chung Ang, Seoul Hàn Quốc năm 2016 ngành xử lý ảnh. Hiện đang làm ứng viên tiến sỹ tại trường đại học quốc gia Seoul lĩnh vực máy tính và thông tin.

Hướng nghiên cứu hiện nay: thiết bị cạnh thông minh, học sâu, học máy, giải thích mô hình trí tuệ nhân tạo.