

BỘ KHOA HỌC VÀ CÔNG NGHỆ  
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



**BÁO CÁO NGHIÊN CỨU KHOA HỌC SINH VIÊN**

**Đề tài: Ứng dụng OpenVINO vào việc nhận dạng phương tiện giao thông**

**Sinh viên thực hiện:** Cao Duy Thái **N22DCCN076**

Nguyễn Duy Thái **N22DCCN077**

Trần Nguyễn Sơn Thành **N22DCCN078**

**Giảng viên hướng dẫn:** ThS. Huỳnh Trung Trụ

**Lớp:** E22CQCNTT02-N

**Khóa:** 2022-2027

**Hệ:** Đại học chính quy

*Hồ Chí Minh, ngày 16 tháng 10 năm 2025*



## MỤC LỤC

LỜI CẢM ƠN .....	4
NHẬN XÉT CỦA NGƯỜI HƯỚNG DẪN .....	5
TÓM TẮT .....	7
CHƯƠNG 1. GIỚI THIỆU .....	8
1.1. Bối cảnh và Đặt vấn đề .....	8
1.1.1. Thách thức Giao thông Đô thị: Góc nhìn Toàn cầu và Địa phương .....	8
1.1.2. Hạn chế của các Hệ thống Giám sát Giao thông Truyền thống .....	8
1.1.3. Vấn đề về Khả năng Mở rộng: Theo dõi qua các Mạng lưới Camera không giao nhau .....	9
1.1.4. Nút thắt cổ chai về Hiệu năng: Nhu cầu Suy luận AI Hiệu suất cao .....	10
1.2. Tổng quan Tình hình Nghiên cứu và Khoảng trống Nghiên cứu .....	10
1.2.1. Tình hình Nghiên cứu Hiện tại về Theo dõi Đa Đối tượng (MOT) và Theo dõi Đa Đối tượng qua Nhiều Camera (MC-MOT) .....	10
1.2.2. Tổng quan về các Kỹ thuật Tái nhận dạng Phương tiện (Re-ID) .....	11
1.2.3. Vai trò của các Framework Tối ưu hóa Mô hình AI .....	12
1.2.4. Xác định Khoảng trống Nghiên cứu .....	12
1.3. Mục tiêu và Phạm vi Nghiên cứu .....	13
1.3.1. Mục tiêu Chính .....	13
1.3.2. Mục tiêu Cụ thể .....	13
1.4. Phạm vi Nghiên cứu .....	14
1.4.1. Đối tượng Nghiên cứu: .....	14
1.4.2. Phạm vi Dữ liệu: .....	15
1.4.3. Phạm vi Kỹ thuật: .....	17
1.5. Phương pháp Nghiên cứu .....	17
1.5.1. Nghiên cứu Lý thuyết: .....	17
1.5.2. Nghiên cứu Thực nghiệm: .....	18
1.5.3. Phân tích So sánh: .....	18
1.6. Đóng góp về Khoa học và Thực tiễn .....	18
1.6.1. Đóng góp về Khoa học: .....	18
1.6.2. Đóng góp về Thực tiễn: .....	18
1.7. Cấu trúc Luận văn .....	19
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT VÀ CÁC CÔNG NGHỆ CỐT LÕI .....	19



<b>2.1. Các Khái niệm Nền tảng trong Thị giác Máy tính cho Hệ thống Giao thông Thông minh (ITS).....</b>	<b>19</b>
<b>2.1.1. Phát hiện Đối tượng: Sự tiến hóa từ Cửa sổ trượt đến các Bộ phát hiện Một lần.....</b>	<b>19</b>
<b>2.1.2. Theo dõi Đối tượng: Nguyên tắc của Theo dõi-bằng-Phát hiện.....</b>	<b>20</b>
<b>2.1.3. Học sâu theo Hệ mét để Tái nhận dạng.....</b>	<b>20</b>
<b>2.2. Quy trình các Mô hình AI: Phân tích Chuyên sâu .....</b>	<b>22</b>
<b>2.2.1. YOLOv12 cho Phát hiện: Động cơ Nhận thức .....</b>	<b>22</b>
<b>2.2.2. Transformer Tích chập Gọn nhẹ (CCT) để Nhận dạng .....</b>	<b>23</b>
<b>2.2.3. Học biểu diễn Đa nhánh để Tái nhận dạng (Re-ID).....</b>	<b>24</b>
<b>2.2.4. BoT-SORT để Theo dõi: Đảm bảo sự Mạch lạc theo Thời gian .....</b>	<b>26</b>
<b>2.3. Bộ công cụ Intel® OpenVINO™: Trình bày Kỹ thuật.....</b>	<b>27</b>
<b>2.3.2. Chi tiết các Thành phần Chính .....</b>	<b>28</b>
<b>2.3.3. Các Kỹ thuật Tối ưu hóa Cơ bản: Một cái nhìn Cận cảnh .....</b>	<b>29</b>
<b>CHƯƠNG 3. PHÂN TÍCH, THIẾT KẾ VÀ TRIỂN KHAI HỆ THỐNG .....</b>	<b>30</b>
<b>3.1. Đặc tả Yêu cầu Hệ thống .....</b>	<b>30</b>
<b>3.1.1. Yêu cầu Chức năng.....</b>	<b>30</b>
<b>3.1.2. Yêu cầu Phi chức năng.....</b>	<b>31</b>
<b>3.2. Kiến trúc Tổng thể của Hệ thống.....</b>	<b>31</b>
<b>3.3. Mô-đun 1: Thu thập và Chuẩn bị Dữ liệu .....</b>	<b>32</b>
<b>3.3.1. Nguồn Dữ liệu: Một Cách tiếp cận Đa diện .....</b>	<b>33</b>
<b>3.3.2. Tự động Thu thập Dữ liệu (data_capture.py).....</b>	<b>33</b>
<b>3.3.3. Xử lý và Hợp nhất Dữ liệu .....</b>	<b>34</b>
<b>3.4. Mô-đun 2: Phát hiện Phương tiện và Theo dõi trong một Camera .....</b>	<b>34</b>
<b>3.4.1. Huấn luyện Bộ phát hiện Phương tiện YOLOv12.....</b>	<b>35</b>
<b>3.4.2. Tích hợp BoT-SORT để Theo dõi Ổn định trong một Camera .....</b>	<b>35</b>
<b>3.5. Mô-đun 3: Nhận dạng Biển số xe (LPR) .....</b>	<b>36</b>
<b>3.5.1. Tinh chỉnh YOLOv12 để Phát hiện Biển số xe .....</b>	<b>37</b>
<b>3.5.2. Triển khai CCT để Nhận dạng Ký tự Quang học (OCR).....</b>	<b>38</b>
<b>3.6. Mô-đun 4: Tái nhận dạng qua Nhiều Camera (MC-MOT) .....</b>	<b>39</b>
<b>3.6.1. Huấn luyện Mô hình Học biểu diễn Đa nhánh .....</b>	<b>39</b>
<b>3.6.2. Cơ sở dữ liệu ID Toàn cầu và Logic Đối sánh .....</b>	<b>40</b>
<b>3.7. Mô-đun 5: Tối ưu hóa Hệ thống với OpenVINO™ .....</b>	<b>43</b>
<b>3.7.1. Quy trình Chuyển đổi Hai giai đoạn.....</b>	<b>43</b>

<b>3.7.2. Triển khai Quy trình Suy luận bằng OpenVINO™.....</b>	<b>44</b>
<b>3.8. Môi trường Phát triển và Nền tảng Phần mềm .....</b>	<b>44</b>
<b>3.8.1. Thông số Kỹ thuật Phần cứng:.....</b>	<b>45</b>
<b>3.8.2. Nền tảng Phần mềm: .....</b>	<b>45</b>
<b>CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM VÀ BÀN LUẬN .....</b>	<b>45</b>
<b>4.1. Kết quả Huấn luyện và Xác thực Mô hình .....</b>	<b>45</b>
<b>4.1.1. Hiệu năng của Bộ phát hiện Biển số xe.....</b>	<b>46</b>
<b>4.1.2. Bàn luận về Độ chính xác của Mô hình .....</b>	<b>46</b>
<b>4.2. Đo lường Hiệu năng Hệ thống.....</b>	<b>47</b>
<b>4.2.1. Phương pháp Đo lường Hiệu năng .....</b>	<b>47</b>
<b>4.2.2. Kết quả Định lượng: Một Bước đột phá về Hiệu năng .....</b>	<b>47</b>
<b>4.2.3. Phân tích Trực quan về Mức tăng Hiệu năng .....</b>	<b>48</b>
<b>4.3. Kết quả Hệ thống về mặt Định tính.....</b>	<b>49</b>
<b>4.3.1. Trình diễn Phát hiện và Theo dõi trên một Camera.....</b>	<b>49</b>
<b>4.3.2. Trình diễn Tái nhận dạng qua Nhiều Camera .....</b>	<b>50</b>
<b>4.4. Bàn luận và Phân tích .....</b>	<b>51</b>
<b>4.4.1. Diễn giải Kết quả: Giải thích cho Bước đột phá về Hiệu năng .....</b>	<b>51</b>
<b>4.4.2. Các điểm mạnh của Hệ thống.....</b>	<b>52</b>
<b>4.4.3. Những Thách thức và Hạn chế Gặp phải.....</b>	<b>53</b>
<b>CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>53</b>
<b>5.1. Kết luận .....</b>	<b>53</b>
<b>5.1.1. Tóm tắt các Kết quả Đạt được .....</b>	<b>54</b>
<b>5.1.2. Đánh giá lại các Mục tiêu.....</b>	<b>54</b>
<b>5.1.3. Phát biểu cuối cùng về các Đóng góp .....</b>	<b>54</b>
<b>5.2. Hướng Phát triển và các Cải tiến Tiềm năng .....</b>	<b>54</b>
<b>5.2.1. Cải tiến Mô hình và Dữ liệu: .....</b>	<b>54</b>
<b>5.2.2. Mở rộng Chức năng: .....</b>	<b>55</b>
<b>5.2.3. Triển khai Phần cứng và Điện toán Biên: .....</b>	<b>55</b>
<b>5.2.4. Tích hợp Hệ thống và Giao diện Người dùng:.....</b>	<b>55</b>

## LỜI CẢM ƠN



Để hoàn thành đề tài nghiên cứu khoa học này, bên cạnh sự nỗ lực cống hiến của mỗi thành viên trong nhóm, chúng tôi đã may mắn nhận được rất nhiều sự chỉ bảo, hỗ trợ và động viên từ thầy hướng dẫn, quý thầy cô, gia đình và bạn bè.

Trước hết, chúng tôi xin được gửi lời cảm ơn sâu sắc và chân thành nhất đến **Thầy Huỳnh Trung Trụ**. Với vai trò là người hướng dẫn khoa học, Thầy đã tận tình chỉ bảo, chia sẻ những kiến thức chuyên môn sâu rộng cùng kinh nghiệm thực tiễn vô cùng quý báu. Những định hướng kịp thời và lời khuyên sáng suốt của Thầy chính là kim chỉ nam giúp chúng tôi vượt qua những khó khăn và hoàn thành tốt đê tài này. Niềm đam mê nghiên cứu và tinh thần khoa học nghiêm túc của Thầy đã trở thành nguồn động lực, truyền cảm hứng cho cả nhóm chúng tôi trong suốt quá trình thực hiện.

Chúng tôi cũng xin trân trọng gửi lời cảm ơn đến Ban Giám đốc Học viện Công nghệ Bưu chính Viễn thông – Cơ sở tại Thành phố Hồ Chí Minh, cùng toàn thể Quý Thầy/Cô thuộc Khoa Công nghệ Thông tin II. Chúng tôi biết ơn vì đã được học tập trong một môi trường tuyệt vời, với cơ sở vật chất hiện đại và được trang bị những kiến thức nền tảng, tạo tiền đề vững chắc để chúng tôi xây dựng và phát triển nghiên cứu này.

Đề tài này là kết quả của sự nỗ lực không mệt mỏi và sự hợp tác ăn ý của cả nhóm. Xin cảm ơn tất cả các thành viên vì đã luôn đoàn kết, tận tụy và kiên trì để cùng nhau vượt qua mọi trở ngại:

1. **Nguyễn Duy Thái (N22DCCN077)** – Nhóm trưởng, phụ trách điều phối công việc, đảm nhiệm chính các công việc phát triển mô hình, định hướng kỹ thuật và duy trì tinh thần làm việc hiệu quả trong nhóm.
2. **Trần Nguyễn Sơn Thành (N22DCCN078)** – Thành viên nhóm, cùng đảm nhiệm các công việc phát triển mô hình, xử lý dữ liệu và đánh giá kết quả, đóng góp quan trọng vào việc hoàn thiện và đảm bảo độ chính xác của hệ thống
3. **Cao Duy Thái (N22DCCN076)** – Thành viên nhóm, cùng đảm nhiệm các công việc phát triển mô hình, xử lý dữ liệu và đánh giá kết quả, đóng góp quan trọng vào việc hoàn thiện và đảm bảo độ chính xác của hệ thống.

Cuối cùng, chúng tôi xin dành lời cảm ơn đặc biệt đến gia đình và bạn bè. Mọi người đã luôn là hậu phương vững chắc, không ngừng cổ vũ và tạo mọi điều kiện tốt nhất để chúng tôi có thể tập trung nghiên cứu và hoàn thành tốt đẹp đê tài này.

Mặc dù nhóm đã rất nỗ lực và cố gắng, tuy nhiên do những hạn chế về kiến thức và kinh nghiệm thực tiễn, báo cáo không thể tránh khỏi những thiếu sót nhất định. Chúng tôi rất mong nhận được những ý kiến đóng góp và chỉ bảo từ Quý Thầy/Cô và các bạn để đê tài được hoàn thiện hơn nữa.

## NHẬN XÉT CỦA NGƯỜI HƯỚNG DẪN

**Tên đề tài:** Phát triển Hệ thống Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT) để Giám sát và Tái nhận dạng Phương tiện với Tối ưu hóa Suy luận bằng Intel OpenVINO™

**Sinh viên thực hiện:**



1. Nguyễn Duy Thái (N22DCCN077)
2. Trần Nguyễn Sơn Thanh (N22DCCN078)
3. Cao Duy Thái (N22DCCN076)

**Lớp:** E22CQCNTT02-N

**Đơn vị:** Học viện Công nghệ Bưu chính Viễn thông, Cơ sở tại Thành phố Hồ Chí Minh

Với tư cách là giảng viên hướng dẫn khoa học cho đề tài nghiên cứu này, tôi đã có dịp hướng dẫn và theo dõi quá trình làm việc của nhóm sinh viên trong suốt vòng đời của dự án, từ giai đoạn lên ý tưởng cho đến khi triển khai và đánh giá cuối cùng.

Nhóm đã lựa chọn một đề tài vừa có tính thực tiễn cao, vừa có độ thách thức lớn về mặt kỹ thuật: phát triển một hệ thống MC-MOT hoàn chỉnh để giám sát giao thông đô thị. Quyết định của nhóm không chỉ tập trung vào việc xây dựng một quy trình AI có chức năng, mà còn giải quyết bài toán thực tế quan trọng về hiệu năng triển khai đã cho thấy một sự thấu hiểu sâu sắc về lĩnh vực này.

Trong suốt quá trình nghiên cứu, các sinh viên đã luôn thể hiện một tinh thần chủ động, năng lực chuyên môn vững vàng và khả năng làm việc nhóm hiệu quả. Nhóm đã áp dụng một phương pháp luận có hệ thống, bắt đầu bằng việc tổng quan tài liệu một cách kỹ lưỡng về các mô hình và thuật toán tiên tiến, sau là một quy trình tỉ mỉ từ thu thập dữ liệu, huấn luyện mô hình đến tích hợp hệ thống. Sự cẩn thận và tận tụy của nhóm trong việc giải quyết các vấn đề phức tạp gặp phải là rất đáng khen ngợi.

Đóng góp quan trọng nhất của đề tài nằm ở sự ứng dụng một cách nghiêm ngặt và thành công Bộ công cụ Intel® OpenVINO™ để tối ưu hóa một quy trình AI phức tạp, đa giai đoạn cho việc triển khai trên nền tảng CPU. Mức tăng hiệu năng được ghi nhận vào khoảng **8858%** (từ 0.17 FPS lên 15.23 FPS) là một thành tựu hết sức ấn tượng, giúp biến đổi hệ thống từ một nguyên lý thuyết thành một ứng dụng khả thi, có khả năng hoạt động trong thời gian thực. Kết quả này không chỉ xác thực cách tiếp cận kỹ thuật của nhóm mà còn cung cấp một nghiên cứu tình huống (case study) đầy thuyết phục về giá trị của việc tối ưu hóa suy luận chuyên dụng.

Báo cáo nghiên cứu cuối cùng có cấu trúc tốt, được trình bày rõ ràng và ghi lại đầy đủ phương pháp luận, chi tiết triển khai, và các kết quả thực nghiệm của nhóm. Đây là một bản trình bày trung thực và toàn diện cho công trình xuất sắc của các em.

Tóm lại, các sinh viên đã thể hiện năng lực ở mức độ cao trong cả nghiên cứu và ứng dụng thực tế. Đề tài đã hoàn thành xuất sắc tất cả các yêu cầu của một đề tài nghiên cứu khoa học ở cấp độ này. Tôi tự hào về công trình của nhóm và tự tin đề cử báo cáo nghiên cứu này để được xem xét và đánh giá với kết quả cao.

**Trân trọng,**

---

**TS. Huỳnh Trung Trụ**

Giảng viên hướng dẫn



Khoa Công nghệ Thông tin II

Học viện Công nghệ Bưu chính Viễn thông, Cơ sở TP.HCM

---

## TÓM TẮT

Việc quản lý hiệu quả giao thông đô thị tại các khu vực đô thị phát triển nhanh như Thành phố Hồ Chí Minh đặt ra một thách thức lớn, chủ yếu do khó khăn trong việc theo dõi phương tiện qua các mạng lưới camera rộng lớn, không giao nhau. Mặc dù học sâu đã mang lại những giải pháp mạnh mẽ cho việc phát hiện và theo dõi đối tượng, việc triển khai chúng trên quy mô toàn thành phố trong thực tế thường bị cản trở bởi chi phí tính toán không lồ, thường đòi hỏi phần cứng GPU đắt tiền và tiêu tốn nhiều năng lượng. Nghiên cứu này giải quyết nút thắt cổ chai quan trọng về hiệu năng này bằng cách trình bày việc thiết kế, triển khai và tối ưu hóa một cách nghiêm ngặt một hệ thống Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT) hoàn chỉnh, được thiết kế để suy luận hiệu năng cao trên các nền tảng CPU phổ biến.

Hệ thống của chúng tôi tích hợp một quy trình phức tạp gồm các mô hình AI tiên tiến nhất để đạt được chức năng toàn diện từ đầu đến cuối. Quy trình này bao gồm: mô hình **YOLOv12** để phát hiện phương tiện và biển số xe một cách mạnh mẽ; mô hình **Transformer Tích chập Gọn nhẹ (CCT)** để nhận dạng chính xác ký tự biển số xe; mô hình **Học biểu diễn Đa nhánh** để tạo ra các vector nhúng phương tiện có tính phân biệt cao phục vụ cho việc Tái nhận dạng (Re-ID); và thuật toán **BoT-SORT** để theo dõi ổn định trong phạm vi một camera.

Đóng góp cốt lõi của công trình này nằm ở việc tối ưu hóa một cách có hệ thống toàn bộ quy trình suy luận này bằng cách sử dụng **Bộ công cụ Intel® OpenVINO™**. Một bài đo lường hiệu năng so sánh nghiêm ngặt đã được tiến hành trên một CPU Intel tiêu chuẩn, đo lường hiệu năng của các mô hình PyTorch cơ sở so với các phiên bản tương ứng đã được tối ưu hóa bằng OpenVINO™. Kết quả cho thấy một bước đột phá mang tính chuyển đổi về hiệu năng: thông lượng hệ thống đã được tăng tốc từ mức không khả thi là **0.17 Khung hình trên Giây (FPS)** lên mức có khả năng hoạt động thời gian thực là **15.23 FPS**, tương đương với mức **tăng khoảng 8858%**. Tương ứng, độ trễ trên mỗi khung hình đã giảm mạnh ~**94.3%**, từ 5720.48 ms xuống còn 327.33 ms.

Nghiên cứu này chứng minh một cách thuyết phục rằng bằng cách tận dụng một bộ công cụ tối ưu hóa chuyên biệt và nhận biết phần cứng, có thể thu hẹp khoảng cách hiệu năng đáng kể giữa CPU và GPU đối với các tác vụ thị giác máy tính phức tạp, đa giai đoạn. Dự án cung cấp một bản thiết kế (blueprint) đã được kiểm chứng, hiệu năng cao và hiệu quả về chi phí cho việc phát triển các Hệ thống Giao thông Thông minh (ITS) có khả năng mở rộng, giúp cho việc triển khai các hệ thống giám sát giao thông tiên tiến dựa trên AI trở thành một nỗ lực dễ tiếp cận và khả thi hơn về mặt kinh tế cho các sáng kiến thành phố thông minh.

Chỉ số (Metric)	Mô hình gốc PyTorch (trên CPU)	Mô hình OpenVINO (trên CPU)	Mức độ cải thiện
<b>Thông lượng (Throughput)</b>	<b>0.17 FPS</b>	<b>15.23 FPS</b>	<b>~8858%</b>
<b>Độ trễ trung bình (Latency)</b>	<b>5720.48 ms</b>	<b>327.33 ms</b>	<b>Giảm ~94.3%</b>

**Từ khóa:** Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT), Tái nhận dạng Phương tiện (Re-ID), Intel OpenVINO, Tối ưu hóa Suy luận, Theo dõi Đối tượng, YOLOv12, Hệ thống Giao thông Thông minh (ITS), Hiệu năng CPU.

## CHƯƠNG 1. GIỚI THIỆU

### 1.1. Bối cảnh và Đặt vấn đề

#### 1.1.1. Thách thức Giao thông Đô thị: Góc nhìn Toàn cầu và Địa phương

Các mạng lưới đường bộ và cao tốc phức tạp, vốn tạo nên hệ thống tuần hoàn của các thành phố hiện đại, đang ngày càng chịu nhiều áp lực. Các trung tâm đô thị trên toàn thế giới, với vai trò là đầu mối của các hoạt động kinh tế và xã hội, đang phải vật lộn với một cuộc khủng hoảng giao thông ngày càng leo thang. Vấn đề này đặc biệt nghiêm trọng tại các siêu đô thị đang phát triển nhanh chóng ở Đông Nam Á, và Thành phố Hồ Chí Minh (TP.HCM), Việt Nam, là một ví dụ điển hình. Sự tăng trưởng nồng độ của thành phố đã dẫn đến sự gia tăng chưa từng có về mật độ phương tiện, gây ra tình trạng ùn tắc giao thông kinh niên, ô nhiễm môi trường gia tăng và tỷ lệ tai nạn giao thông cao hơn. Những thách thức này mang lại những hệ quả kinh tế - xã hội sâu sắc, bao gồm tổn thất đáng kể về năng suất kinh tế do chậm trễ trong di chuyển, tăng chi phí vận hành cho hoạt động logistics và ảnh hưởng đến an toàn công cộng. Mật độ và sự phức tạp của luồng giao thông đã khiến cho việc giám sát thủ công và các phương pháp quản lý truyền thống trở nên không còn đủ hiệu quả, tạo ra một nhu cầu cấp bách và không thể phủ nhận đối với các giải pháp thông minh, tự động và có khả năng mở rộng.

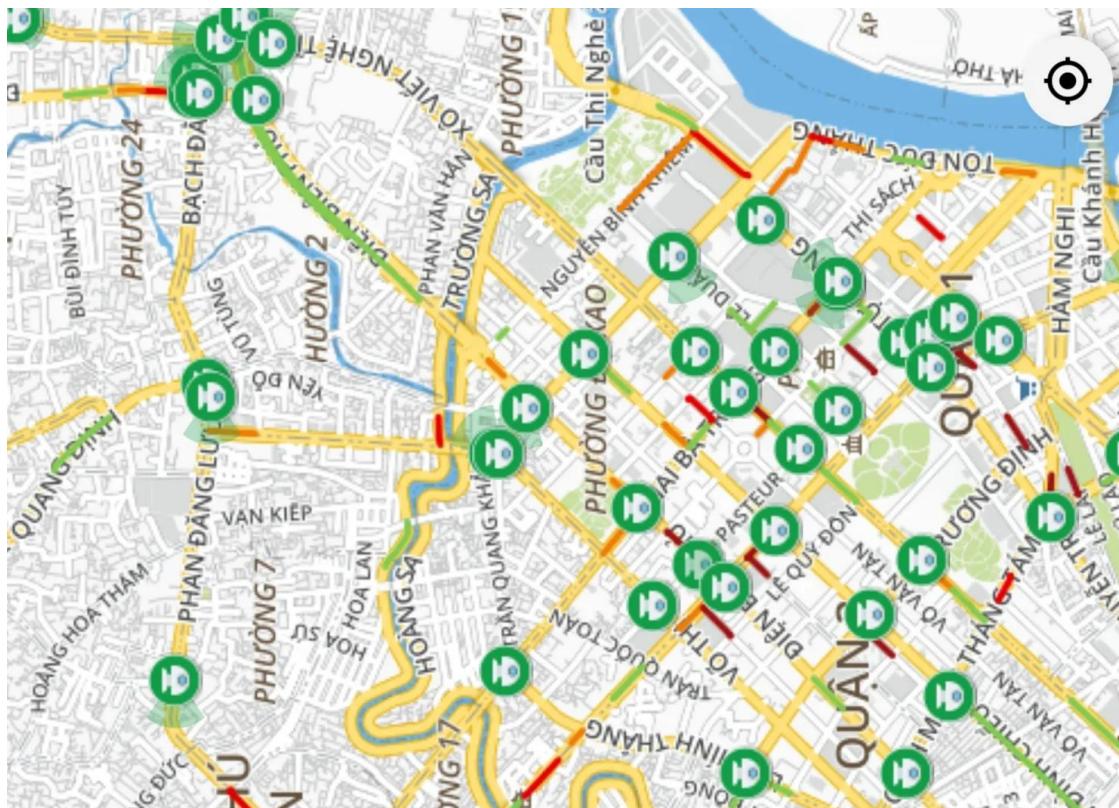
#### 1.1.2. Hạn chế của các Hệ thống Giám sát Giao thông Truyền thống

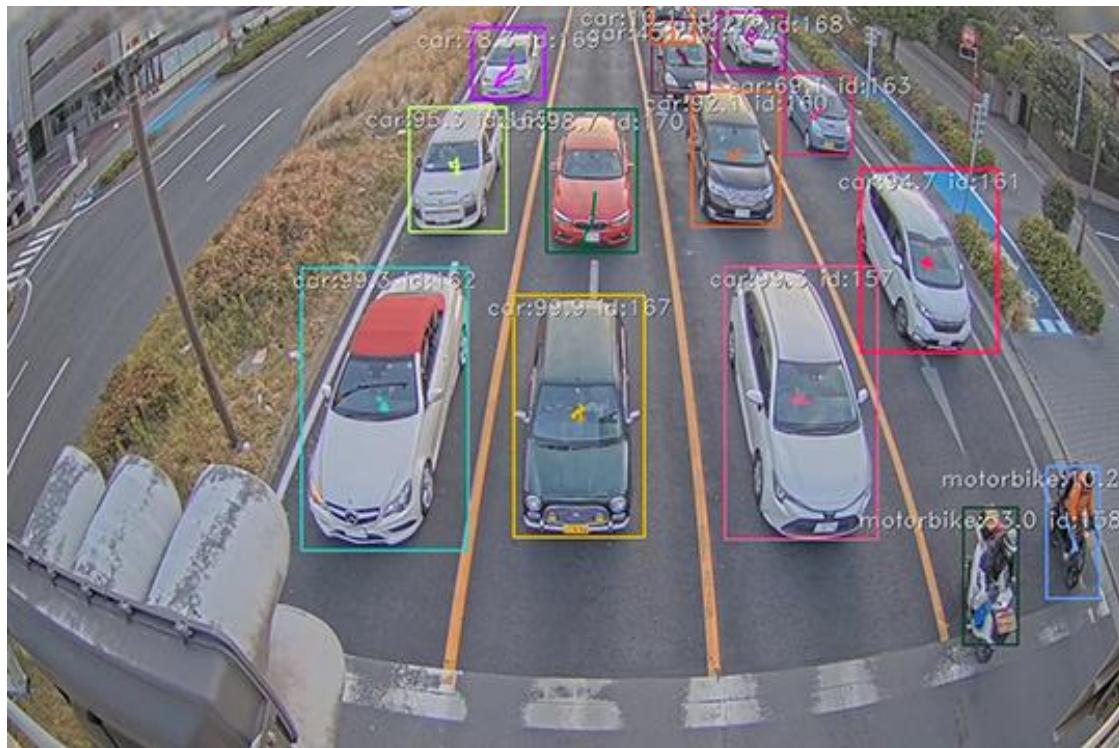
Trong lịch sử, việc giám sát giao thông đã dựa vào sự kết hợp của việc quan sát thủ công bởi người điều phối giao thông, các bộ dò vòng cảm ứng đặt dưới lòng đường, và các mạng lưới camera quan sát (CCTV) được theo dõi bởi con người. Mặc dù các phương pháp này đã cung cấp một mức độ chức năng cơ bản, chúng lại bộc lộ nhiều hạn chế trong bối cảnh đô thị hiện đại. Các vòng cảm ứng có thể đếm phương tiện nhưng không thể xác định danh tính, phân loại hay theo dõi hành trình của chúng. Việc giám sát CCTV do con người thực hiện không có khả năng mở rộng; nó tốn nhiều nhân lực, dễ bị mệt mỏi và mắc lỗi của con người, và về cơ bản là mang tính phản ứng thay vì chủ động. Một người vận hành không thể giám sát hiệu quả hàng chục, chứ chưa nói đến hàng trăm, nguồn cấp dữ liệu camera cùng một lúc, khiến cho việc theo dõi các phương tiện cụ thể hoặc hiểu được các quy luật giao thông trên toàn thành phố trong thời gian thực là điều không thể.

### **1.1.3. Vấn đề về Khả năng Mở rộng: Theo dõi qua các Mạng lưới Camera không giao nhau**

Sự ra đời của Trí tuệ Nhân tạo (AI) và thị giác máy tính đã mang lại những công cụ mạnh mẽ để tự động phát hiện và theo dõi phương tiện trong tầm quan sát của một camera duy nhất. Tuy nhiên, một thách thức phức tạp hơn nhiều xuất hiện trong môi trường đô thị điển hình: đó là bài toán Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT). Các mạng lưới camera toàn thành phố về bản chất là rời rạc, với những khoảng trống đáng kể và không có vùng phủ giao nhau giữa hầu hết các nút camera. Một phương tiện có thể được ghi lại bởi một camera, biến mất trong vài phút, và sau đó xuất hiện trở lại trong tầm quan sát của một camera khác cách đó vài cây số.

Điều này tạo ra một bài toán liên kết dữ liệu quan trọng: làm thế nào một hệ thống có thể xác định với độ tin cậy cao rằng chiếc xe được nhìn thấy bởi Camera A chính là chiếc xe được nhìn thấy sau đó bởi Camera C? Giải quyết được vấn đề này là cốt lõi của thách thức Tái nhận dạng Phương tiện (Re-ID). Nó đòi hỏi hệ thống phải vượt ra ngoài việc theo dõi vị trí đơn giản và học được một “dấu vân tay” hay vector nhúng (embedding) độc nhất dựa trên ngoại hình cho mỗi phương tiện, mà vẫn nhất quán qua các góc nhìn, điều kiện ánh sáng và độ phân giải khác nhau. Nếu không có khả năng Tái nhận dạng mạnh mẽ, một hệ thống giám sát chỉ có thể cung cấp những hình ảnh tức thời, riêng lẻ về giao thông, chứ không phải là một câu chuyện mạch lạc về hành trình của phương tiện trên toàn thành phố.





#### 1.1.4. Nút thắt cổ chai về Hiệu năng: Nhu cầu Suy luận AI Hiệu suất cao

Các mô hình học sâu cần thiết để giải quyết bài toán phát hiện, theo dõi và tái nhận dạng đòi hỏi năng lực tính toán rất lớn. Các mô hình tiên tiến nhất, mặc dù chính xác, yêu cầu sức mạnh xử lý đáng kể cho quá trình suy luận (inference)—quá trình áp dụng một mô hình đã được huấn luyện vào dữ liệu mới, trực tiếp. Giải pháp thông thường cho nút thắt cổ chai về hiệu năng này là triển khai các Bộ xử lý Đồ họa (GPU) đắt đỏ và tiêu thụ nhiều năng lượng. Mặc dù hiệu quả, cách tiếp cận này tạo ra những rào cản đáng kể cho việc triển khai quy mô lớn trên toàn thành phố do chi phí phần cứng cao, mức tiêu thụ năng lượng lớn và sự phức tạp trong việc bảo trì máy chủ. Do đó, một câu hỏi quan trọng được đặt ra: liệu có thể đạt được hiệu năng thời gian thực cần thiết cho một hệ thống MC-MOT quy mô thành phố mà không cần dựa vào phần cứng GPU chuyên dụng không? Thách thức này nhấn mạnh sự cần thiết của các kỹ thuật tối ưu hóa mô hình tiên tiến có thể khai phá toàn bộ tiềm năng của các Bộ xử lý Trung tâm (CPU) phổ biến và hiệu quả hơn về chi phí, vốn đã có sẵn trong cơ sở hạ tầng máy chủ hiện tại.

### 1.2. Tổng quan Tình hình Nghiên cứu và Khoảng trống Nghiên cứu

Để hiểu một cách toàn diện về hệ thống được đề xuất, cần phải xem xét các nghiên cứu đã được công bố trong các lĩnh vực cấu thành của nó: theo dõi đa đối tượng, tái nhận dạng phương tiện, và tối ưu hóa mô hình AI. Phần tổng quan này sẽ khảo sát các công nghệ tiên tiến nhất, xác định những hạn chế có hứa của chúng, và qua đó xác định khoảng trống nghiên cứu chính xác mà dự án này hướng đến giải quyết.

#### 1.2.1. Tình hình Nghiên cứu Hiện tại về Theo dõi Đa Đối tượng (MOT) và Theo dõi Đa Đối tượng qua Nhiều Camera (MC-MOT)

Mô hình chủ đạo trong Theo dõi Đa Đối tượng hiện đại là Theo dõi-bằng-Phát hiện (Tracking-by-Detection). Cách tiếp cận này đầu tiên sử dụng một bộ phát hiện đối tượng mạnh mẽ để xác định tất cả các mục tiêu tiềm năng trong mỗi khung hình, sau đó sử dụng một thuật toán liên kết riêng biệt để nối các phát hiện này theo thời gian. Các phương pháp thành công ban đầu, chẳng hạn như SORT (Simple Online and Realtime Tracking), đã sử dụng Bộ lọc Kalman để dự đoán chuyển động và thuật toán Hungarian để liên kết các phát hiện dựa trên độ trùng lặp của hộp giới hạn (IoU). Mặc dù hiệu quả, SORT gặp khó khăn với các trường hợp che khuất và chuyển đổi ID do chỉ dựa hoàn toàn vào các tín hiệu chuyển động.

Những tiến bộ sau đó, như DeepSORT, đã cải tiến khuôn khổ này bằng cách tích hợp một mô hình học sâu dựa trên ngoại hình để trích xuất đặc trưng từ các đối tượng được phát hiện, làm cho quá trình liên kết trở nên mạnh mẽ hơn trước các trường hợp biến mất tạm thời. Thuật toán BoT-SORT, được sử dụng trong dự án này, đại diện cho một bước tiến hóa tiếp theo trong dòng phát triển này. Nó tích hợp một bộ phát hiện hiệu năng cao (như YOLO), một Bộ lọc Kalman tinh vi hơn bao gồm cả Bù trừ Chuyển động Camera (CMC) để xử lý các camera không tĩnh, và một chiến lược liên kết dựa trên Tái nhận dạng (Re-ID) kết hợp cả tín hiệu ngoại hình và chuyển động để đạt được sự ổn định theo dõi vượt trội. Mặc dù các phương pháp này xuất sắc trong việc theo dõi trong phạm vi một camera (intra-camera tracking), chúng không giải quyết được vấn đề tái nhận dạng một đối tượng khi nó rời khỏi tầm nhìn của camera này và đi vào một camera khác. Đây là lĩnh vực của MC-MOT, vốn phụ thuộc một cách cốt lõi vào sức mạnh của thành phần Tái nhận dạng Phương tiện bên trong nó.

### 1.2.2. Tổng quan về các Kỹ thuật Tái nhận dạng Phương tiện (Re-ID)

Tái nhận dạng Phương tiện (Re-ID) nhằm mục đích đối sánh hình ảnh của cùng một phương tiện được ghi lại từ các camera khác nhau vào những thời điểm khác nhau. Các cách tiếp cận ban đầu dựa trên các đặc trưng thủ công như SIFT, SURF, và biểu đồ màu, nhưng những phương pháp này rất nhạy cảm với các biến thể về góc nhìn, ánh sáng và tỷ lệ.

Công nghệ tiên tiến nhất hiện nay bị chi phối bởi Học sâu theo Hệ mét (Deep Metric Learning), trong đó một Mạng Nơ-ron Tích chập (CNN) được huấn luyện để chiều các hình ảnh vào một không gian nhúng (embedding space) nhiều chiều. Trong không gian này, hình ảnh của cùng một phương tiện được gom cụm lại gần nhau, trong khi hình ảnh của các phương tiện khác nhau bị đẩy ra xa nhau. Các kiến trúc dựa trên ResNet và các biến thể của nó (chẳng hạn như IBN-Net được sử dụng trong dự án này) thường được sử dụng làm mạng xương sống (backbone) để trích xuất đặc trưng. Để tăng cường hơn nữa sức mạnh phân biệt, các nhà nghiên cứu đã tích hợp các kỹ thuật tiên tiến như:

- Cơ chế Chú ý (Attention Mechanisms):** Cho phép mô hình tập trung vào các vùng hình ảnh nổi bật nhất của một phương tiện (ví dụ: logo, các dấu hiệu độc nhất, cánh lướt gió) trong khi loại bỏ nhiễu nền.
- Kiến trúc Đa nhánh (Multi-Branche Architectures):** Như đã được chứng minh trong bài báo “Strength in Diversity: Multi-Branche Representation Learning for Vehicle Re-Identification,” (Sức mạnh trong sự Đa dạng: Học biểu diễn Đa nhánh để Tái nhận dạng Phương tiện) mà từ đó mô hình Re-ID của chúng tôi được phát

triển, việc sử dụng nhiều nhánh song song trong mạng có thể mang lại hiệu quả cao. Bằng cách buộc các nhánh khác nhau học các loại đặc trưng khác nhau—ví dụ, một đặc trưng toàn cục dựa trên kết cấu (texture) từ một khối ResNet và một đặc trưng cục bộ dựa trên các bộ phận từ một khối chú ý (BotNet)—vector nhúng kết hợp cuối cùng trở nên mạnh mẽ và có tính mô tả cao hơn đáng kể. Chiến lược Phân chia Nhánh theo Hàm mất mát (Loss-Branch-Split - LBS) là một yếu tố then chốt cho phép điều này, buộc mỗi nhánh phải chuyên môn hóa và đóng góp thông tin độc nhất, bổ sung cho nhau.

### 1.2.3. Vai trò của các Framework Tối ưu hóa Mô hình AI

Việc triển khai các mô hình học sâu phức tạp này trong các ứng dụng thực tế thường bị hạn chế bởi yêu cầu tính toán của chúng. Để thu hẹp khoảng cách giữa nghiên cứu và ứng dụng thực tiễn, một số framework tối ưu hóa mô hình đã được phát triển. TensorRT của NVIDIA vượt trội trong việc tối ưu hóa các mô hình cho GPU độc quyền của họ thông qua các kỹ thuật như hợp nhất nhân (kernel fusion) và hiệu chỉnh độ chính xác. Đối với các triển khai trên thiết bị di động và thiết bị biến, TensorFlow Lite và PyTorch Mobile cung cấp các công cụ để chuyển đổi và lượng tử hóa mô hình để chạy hiệu quả trên các bộ xử lý dựa trên ARM.

Bộ công cụ Intel® OpenVINO™ (Open Visual Inference and Neural Network Optimization) tạo ra một thị trường ngách riêng biệt và quan trọng: tối đa hóa hiệu năng suy luận trên hệ sinh thái rộng lớn của phần cứng Intel, bao gồm CPU, GPU tích hợp (iGPU) và Bộ xử lý Tầm nhìn (VPU). Nó đạt được điều này bằng cách chuyển đổi các mô hình từ các framework phổ biến thành một Biểu diễn Trung gian (Intermediate Representation - IR) và sau đó sử dụng một Bộ máy Suy luận (Inference Engine) được tối ưu hóa cao để thực thi chúng, tận dụng các tập lệnh phần cứng cấp thấp (ví dụ: AVX2, AVX512) mà các framework học sâu tiêu chuẩn thường không tận dụng hết.

**Figure 1.2 - Comparison of AI Optimization Frameworks**

Highlighting OpenVINO's suitability for Intel-based project objectives

Feature	TensorRT	TensorFlow Lite	OpenVINO
Primary Target Hardware	NVIDIA GPUs	Mobile & Embedded (CPUs, GPUs, DSPs)	Intel Hardware (CPUs, iGPUs, VPUs)
Core Optimization Approach	<ul style="list-style-type: none"> <li>- Layer &amp; Tensor Fusion</li> <li>- Precision Calibration (FP16, INT8)</li> <li>- Kernel Auto-Tuning</li> <li>- Memory Optimization</li> </ul>	<ul style="list-style-type: none"> <li>- Quantization (Lượng tử hóa)</li> <li>- Weight Pruning (Tỉa trọng số)</li> <li>- Model Topology Transforms</li> <li>- Tối ưu hóa cho thiết bị hạn chế</li> </ul>	<ul style="list-style-type: none"> <li>- Model Conversion to IR</li> <li>- Quantization &amp; Pruning</li> <li>- Hardware-aware Scheduling</li> <li>- Asynchronous Execution</li> </ul>

### 1.2.4. Xác định Khoảng trống Nghiên cứu

Các tài liệu nghiên cứu đã đưa ra các giải pháp mạnh mẽ cho các thành phần riêng lẻ của một hệ thống giám sát: các bộ theo dõi trên một camera hiệu quả như BoT-SORT, các mô hình Tái nhận dạng mạnh mẽ dựa trên học đa nhánh, và các bộ công cụ tối ưu hóa hiệu quả như OpenVINO. Tuy nhiên, một khoảng trống nghiên cứu đáng kể tồn tại ở giao điểm của các lĩnh vực này. Cụ thể, hai lĩnh vực chưa được khám phá sâu:

1. **Tích hợp Hệ thống Toàn diện (End-to-End):** Có sự khan hiếm các nghiên cứu trình bày chi tiết việc tích hợp hoàn chỉnh các thành phần tiên tiến, riêng biệt này thành một quy trình MC-MOT duy nhất, gắn kết và hoạt động được, được thiết kế cho một môi trường đô thị phức tạp trong thế giới thực.
2. **Bước đột phá về Hiệu năng được Định lượng trên Phần cứng Phổ thông:** Mặc dù lợi ích của OpenVINO đã được ghi nhận rõ ràng đối với các bài kiểm tra hiệu năng trên một mô hình duy nhất, vẫn có sự thiếu hụt rõ rệt các nghiên cứu tình huống toàn diện mà có thể chứng minh một cách định lượng tác động của nó đối với một quy trình AI phức tạp, đa giai đoạn (tức là phát hiện nối tiếp với nhận dạng và Tái nhận dạng). Tiềm năng đạt được sự tăng hiệu năng mang tính chuyển đổi, gần như thời gian thực từ một đến hai bậc độ lớn (ví dụ: từ <1 FPS lên >15 FPS) trên cơ sở hạ tầng CPU tiêu chuẩn vẫn là một lĩnh vực quan trọng, chưa được khám phá đầy đủ.

Dự án này trực tiếp giải quyết khoảng trống quan trọng này. Nó vượt ra ngoài việc phân tích các thành phần riêng lẻ để xây dựng, và quan trọng nhất là tối ưu hóa và đo lường hiệu năng của toàn bộ một hệ thống từ đầu đến cuối. Đóng góp khoa học chính là sự chứng minh nghiêm ngặt, định lượng về khả năng của OpenVINO trong việc nâng cấp một quy trình MC-MOT vốn không khả thi về mặt tính toán trở thành một hệ thống hiệu năng cao, hoạt động được trên phần cứng CPU phổ biến, qua đó cung cấp một bản thiết kế thực tiễn và có khả năng mở rộng cho các hệ thống giao thông thông minh trong tương lai.

### 1.3. Mục tiêu và Phạm vi Nghiên cứu

Dựa trên khoảng trống nghiên cứu đã xác định, dự án này được định hướng bởi một mục tiêu chính rõ ràng, được chia nhỏ thành một loạt các nhiệm vụ cụ thể, đo lường được, có thể đạt được, phù hợp và có giới hạn thời gian (SMART). Phạm vi được xác định cẩn thận để đảm bảo dự án luôn tập trung và các kết quả của nó được xác thực một cách thấu đáo.

#### 1.3.1. Mục tiêu Chính

Mục tiêu chính của nghiên cứu này là thiết kế, triển khai, và đánh giá một cách nghiêm ngặt một hệ thống Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT) hoàn chỉnh có khả năng phát hiện, theo dõi và tái nhận dạng phương tiện qua một mạng lưới các camera không giao nhau, được thiết kế để đạt được hiệu năng suy luận thời gian thực trên phần cứng CPU tiêu chuẩn thông qua việc tối ưu hóa có hệ thống bằng Bộ công cụ Intel® OpenVINO™. Công trình này nhằm mục đích phục vụ như một bằng chứng về khái niệm toàn diện cho một giải pháp giám sát giao thông thông minh có khả năng mở rộng và hiệu quả về chi phí.

#### 1.3.2. Mục tiêu Cụ thể

Để đạt được mục tiêu chính, các nhiệm vụ nghiên cứu và phát triển cụ thể sau đây sẽ được thực hiện:

1. **Nghiên cứu, Huấn luyện và Tinh chỉnh một Quy trình gồm các Mô hình AI Chuyên dụng:**

1. **Phát hiện Phương tiện và Biển số xe:** Huấn luyện và xác thực một cách có hệ thống mô hình YOLOv12 để đạt được độ chính xác cao trong việc phát hiện cả phương tiện (ô tô, xe tải, xe buýt) và biển số xe tương ứng trong các bối cảnh đô thị phức tạp.
  2. **Nhận dạng Ký tự Biển số xe:** Triển khai và huấn luyện mô hình Compact Convolutional Transformer (CCT) có khả năng phiên âm chính xác các ký tự chữ và số từ các hình ảnh biển số xe đã được cắt ra, xử lý được các biến thể về góc độ và ánh sáng.
  3. **Tái nhận dạng Phương tiện:** Triển khai và huấn luyện một mô hình Học biểu diễn Đa nhánh (Multi-Branh Representation Learning) tinh vi, tận dụng các mạng xương sống ResNet50 và BotNet, để tạo ra một vector nhúng đặc trưng có tính phân biệt cao cho mỗi phương tiện được phát hiện, cho phép đổi sánh mạnh mẽ qua các góc nhìn camera khác biệt.
2. **Triển khai một Thuật toán Theo dõi Ôn định và Hiệu quả:**
1. Tích hợp bộ phát hiện phương tiện YOLOv12 đã được huấn luyện với thuật toán BoT-SORT để thiết lập và duy trì các ID theo dõi ổn định, nhất quán cho tất cả các phương tiện trong tầm quan sát của bất kỳ camera đơn lẻ nào.
3. **Tối ưu hóa một cách có Hệ thống Toàn bộ Quy trình Suy luận:**
1. Phát triển một quy trình làm việc có thể lặp lại để chuyển đổi tất cả các mô hình PyTorch đã được huấn luyện sang định dạng Open Neural Network Exchange (ONNX).
  2. Sử dụng Bộ tối ưu hóa Mô hình (Model Optimizer) của OpenVINO™ để chuyển đổi các mô hình ONNX sang định dạng Biểu diễn Trung gian (IR) hiệu năng cao của framework, đặc biệt nhằm đến độ chính xác FP32 để đạt độ chính xác tối đa.
  3. Tái cấu trúc logic suy luận của hệ thống để chỉ sử dụng độc quyền API của Bộ máy Suy luận (Inference Engine) OpenVINO™ cho tất cả các hoạt động thực thi mô hình trên CPU.
4. **Đo lường và Phân tích Định lượng Hiệu năng Hệ thống:**
- Thiết kế và thực hiện một thí nghiệm đo lường hiệu năng có kiểm soát để đo lường và so sánh một cách tỉ mỉ hiệu năng từ đầu đến cuối của mô-đun nhận dạng biển số xe.
- Các chỉ số chính để so sánh sẽ là **Thông lượng** (đo bằng **Khung hình trên giây - FPS**) và **Độ trễ** (đo bằng **mili giây - ms**).
- Việc so sánh sẽ được thực hiện giữa hệ thống cơ sở chạy trên framework PyTorch gốc và hệ thống đã được tối ưu hóa hoàn toàn chạy qua OpenVINO™, trên cùng một phần cứng CPU, để định lượng chính xác bước đột phá về hiệu năng đã đạt được.

#### 1.4. Phạm vi Nghiên cứu

Để đảm bảo các mục tiêu nghiên cứu được đáp ứng với sự chặt chẽ khoa học và trong một khuôn khổ có thể quản lý được, phạm vi của dự án này được xác định như sau:

##### 1.4.1. Đối tượng Nghiên cứu:

Đối tượng cốt lõi của nghiên cứu này là các mô hình và thuật toán AI cấu thành nên quy trình MC-MOT, cụ thể là: kiến trúc YOLOv12 để phát hiện, Compact Convolutional Transformer để nhận dạng, kiến trúc Học biểu diễn Đa nhánh (Multi-Branch Representation Learning) để Tái nhận dạng, và thuật toán BoT-SORT để theo dõi. Trọng tâm công nghệ chính là việc ứng dụng và đánh giá Bộ công cụ Intel® OpenVINO™ như một framework tối ưu hóa và suy luận chủ đạo.

#### **1.4.2. Phạm vi Dữ liệu:**

Bộ dữ liệu chính để phát triển hệ thống và đánh giá định tính bao gồm dữ liệu hình ảnh thực tế được thu thập từ một mạng lưới khoảng 200 camera giao thông công cộng tại Thành phố Hồ Chí Minh. Trọng tâm địa lý là bán kính 4 kilômét xung quanh Dinh Độc Lập, cung cấp một môi trường đô thị dày đặc và đa dạng. Dữ liệu thực tế này được bổ sung trong các giai đoạn huấn luyện mô hình bởi các bộ dữ liệu công khai đã được công nhận, bao gồm Vehicle-1M, VRIC, và nhiều bộ dữ liệu khác có nguồn gốc từ Roboflow Universe để đảm bảo tính mạnh mẽ và khả năng tổng quát hóa của mô hình.

Nguồn dữ liệu bao gồm (Nguồn dữ liệu đều được accept agreement và được thông qua, bạn có thể đến trang chủ được đề cập đến và truy cập dữ liệu, tôi sẽ không cung cấp dữ liệu chuẩn)

<https://app.roboflow.com/finetune1/detectors-license-plate>

PKU-VehicleID: <https://www.pkuml.org/resources/pku-vehicleid.html>

1. → The “VehicleID” dataset contains data captured during daytime by multiple real-world surveillance cameras distributed in a small city in China. There are 26267 vehicles(221763 images in total) in the entire dataset. Each image is attached with an id label corresponding to its identity in real world. In addition, we manually labeled 10319 vehicles(90196 images in total) of their vehicle model information(i.e.“MINI-cooper”, “Audi A6L” and “BWM 1 Series”).



2.  
3.

- Vehicle-1M: <https://nlpr.ia.ac.cn/iva/homepage/jqwang/Vehicle1M.htm>
4. → The Vehicle-1M dataset is constructed by National Laboratory of Pattern Recognition, Institute of Automation, University of Chinese Academy of Sciences (NLPR, CASIA). This dataset involves vehicle images captured across day and night, from head or rear, by multiple surveillance cameras installed in several cities in China. There are totally 936,051 images from 55,527 vehicles and 400 vehicle models in the dataset. Each image is attached with a vehicle ID label denoting its identity in real world as well as a vehicle model label indicating the make, model and year of the vehicle(i.e. “Audi-A6-2013”).



5.

6.

#### **1.4.3. Phạm vi Kỹ thuật:**

Việc triển khai hệ thống được thực hiện bằng ngôn ngữ lập trình Python, sử dụng framework học sâu PyTorch để huấn luyện mô hình. Việc đánh giá hiệu năng và mục tiêu triển khai cuối cùng được giới hạn rõ ràng trên các nền tảng CPU của Intel. Nghiên cứu này không mở rộng sang việc đo lường hiệu năng trên GPU hoặc các bộ tăng tốc phần cứng chuyên dụng khác (ví dụ: FPGA, ASIC), vì giả thuyết trung tâm xoay quanh việc tối đa hóa hiệu năng trên phần cứng CPU phổ thông. Việc tối ưu hóa tập trung vào mức độ chính xác FP32 (dầu phẩy động 32-bit) để duy trì độ chính xác cao nhất có thể trong khi vẫn chứng minh được những lợi ích về mặt kiến trúc và phần mềm từ OpenVINO.

### **1.5. Phương pháp Nghiên cứu**

Dự án này áp dụng một phương pháp luận kết hợp giữa nghiên cứu lý thuyết và nghiên cứu thực nghiệm ứng dụng để đảm bảo một kết quả vững chắc và có cơ sở.

#### **1.5.1. Nghiên cứu Lý thuyết:**

Giai đoạn đầu tiên bao gồm việc tổng quan toàn diện các tài liệu hiện có. Điều này bao gồm việc nghiên cứu các bài báo khoa học nền tảng về phát hiện đối tượng (họ YOLO), theo dõi đối tượng (SORT, DeepSORT, BoT-SORT), và học sâu theo hệ mét (deep metric learning) cho Tái nhận dạng Phương tiện. Giai đoạn này cũng bao gồm việc phân tích sâu tài liệu kỹ thuật chính thức của Bộ công cụ Intel® OpenVINO™ để hiểu đầy đủ về kiến trúc, công cụ và các nguyên tắc tối ưu hóa cơ bản của nó.

### **1.5.2. Nghiên cứu Thực nghiệm:**

Đây là giai đoạn cốt lõi của dự án và tuân theo một quy trình phát triển có cấu trúc, lặp đi lặp lại:

1. **Thiết kế Hệ thống:** Xây dựng kiến trúc cho quy trình phần mềm dạng mô-đun.
2. **Triển khai:** Viết mã nguồn cho việc thu thập dữ liệu, tích hợp mô hình, và logic hệ thống.
3. **Huấn luyện & Tinh chỉnh Mô hình:** Huấn luyện các mô hình AI khác nhau trên các bộ dữ liệu kết hợp cho đến khi đạt được độ chính xác thỏa đáng.
4. **Tối ưu hóa:** Chuyển đổi các mô hình đã huấn luyện và tích hợp chúng vào quy trình suy luận của OpenVINO™.
5. **Kiểm thử & Đo lường Hiệu năng:** Đo lường một cách nghiêm ngặt hiệu năng của hệ thống để thu thập dữ liệu định lượng cho việc phân tích.

### **1.5.3. Phân tích So sánh:**

Giai đoạn cuối cùng của phương pháp luận tập trung vào một sự so sánh trực tiếp, dựa trên bằng chứng để xác thực giả thuyết chính của dự án. Các chỉ số hiệu năng (FPS và Độ trễ) của hệ thống cơ sở (các mô hình chạy nguyên bản trong PyTorch trên CPU) được ghi lại. Các bài kiểm tra tương tự sau đó được thực hiện trên hệ thống đã được tối ưu hóa (các mô hình chạy thông qua Bộ máy Suy luận OpenVINO™ trên cùng một CPU). Dữ liệu thu được sau đó được phân tích để tính toán phần trăm cải thiện và rút ra các kết luận dứt khoát về hiệu quả của việc tối ưu hóa bằng OpenVINO™.

## **1.6. Đóng góp về Khoa học và Thực tiễn**

Nghiên cứu này dự kiến sẽ mang lại những đóng góp đáng kể cho cả lĩnh vực học thuật và thực tiễn của AI ứng dụng và các hệ thống giao thông thông minh.

### **1.6.1. Đóng góp về Khoa học:**

Đóng góp khoa học chính của công trình này là việc tạo ra một nghiên cứu tình huống chi tiết, có thể tái tạo, nhằm xác thực một cách định lượng tác động mang tính chuyên đổi của OpenVINO™ đối với một quy trình thị giác máy tính phức tạp, đa mô hình. Bằng cách chứng minh sự tăng hiệu năng gần hai bậc độ lớn (từ ~0.17 FPS lên >15 FPS), nghiên cứu này cung cấp cho cộng đồng học thuật một dữ liệu thuyết phục và một phương pháp luận đã được xác thực để triển khai các hệ thống AI tinh vi trên cơ sở hạ tầng CPU phổ biến, thách thức quan niệm rằng các tác vụ như vậy chỉ thuộc về lĩnh vực của các GPU cao cấp.

### **1.6.2. Đóng góp về Thực tiễn:**

Về mặt thực tiễn, dự án này mang lại một bằng chứng về khái niệm có hiệu năng cao, hiệu quả về chi phí cho một hệ thống giám sát giao thông tiên tiến. Nó cung cấp một bản thiết kế hữu hình mà các cơ quan nhà nước (như các sở giao thông vận tải thành phố) và các doanh nghiệp tư nhân có thể sử dụng để phát triển và triển khai các giải pháp thành phố thông minh có khả năng mở rộng. Khả năng thực hiện theo dõi và tái nhận dạng thời gian thực trên phần cứng máy chủ hiện có hoặc phô thông làm giảm đáng kể rào cản gia nhập, khiến cho việc triển khai quản lý giao thông thông minh quy mô thành phố trở thành một nỗ lực khả thi và hiệu quả hơn về mặt kinh tế.

## 1.7. Cấu trúc Luận văn

Báo cáo này được tổ chức thành năm chương, mỗi chương đề cập đến một khía cạnh cụ thể của nghiên cứu:

1. **Chương 1** cung cấp một phần giới thiệu toàn diện về vấn đề nghiên cứu, tổng quan các tài liệu hiện có, xác định các mục tiêu và phạm vi, phác thảo phương pháp luận, và nêu bật những đóng góp của dự án.
2. **Chương 2** đi sâu vào các nền tảng lý thuyết và công nghệ cốt lõi, cung cấp giải thích kỹ thuật chi tiết về các mô hình AI, thuật toán theo dõi, và Bộ công cụ OpenVINOTM tạo nên cơ sở của hệ thống.
3. **Chương 3** mô tả quá trình hoàn chỉnh của việc phân tích, thiết kế và triển khai hệ thống, chi tiết hóa kiến trúc của từng mô-đun từ thu thập dữ liệu đến tối ưu hóa cuối cùng.
4. **Chương 4** trình bày và phân tích các kết quả của dự án, bao gồm hiệu năng huấn luyện mô hình, kết quả của các bài đo lường hiệu năng quan trọng, và các ví dụ định tính về hệ thống khi hoạt động.
5. **Chương 5** kết luận báo cáo bằng cách tóm tắt những phát hiện chính, thảo luận về những hạn chế của công trình hiện tại, và đề xuất các hướng đi cụ thể cho nghiên cứu và phát triển trong tương lai.

## CHƯƠNG 2. CƠ SỞ LÝ THUYẾT VÀ CÁC CÔNG NGHỆ CỐT LÕI

Một hệ thống tinh vi, như hệ thống được phát triển trong nghiên cứu này, được xây dựng dựa trên nền tảng của các khái niệm lý thuyết đã được công nhận và một chặng công nghệ được lựa chọn cẩn thận. Chương này trình bày chi tiết về các thành phần cốt lõi đó. Chương bắt đầu bằng việc phác thảo các bài toán thị giác máy tính cơ bản liên quan đến giám sát giao thông, sau đó đi sâu vào các kiến trúc cụ thể của các mô hình AI được chọn cho từng tác vụ, và kết thúc bằng một phân tích kỹ thuật về Bộ công cụ Intel® OpenVINOTM, yếu tố then chốt cho phép hệ thống đạt được hiệu năng cao.

### 2.1. Các Khái niệm Nền tảng trong Thị giác Máy tính cho Hệ thống Giao thông Thông minh (ITS)

Khả năng tự động hiểu dữ liệu hình ảnh từ camera giao thông dựa trên việc giải quyết một số bài toán thị giác máy tính cơ bản. Hệ thống của chúng tôi giải quyết các bài toán này theo một quy trình tuần tự.

#### 2.1.1. Phát hiện Đối tượng: Sự tiến hóa từ Cửa sổ trượt đến các Bộ phát hiện Một lần

Phát hiện đối tượng là tác vụ xác định và định vị các thực thể của những đối tượng cụ thể (ví dụ: ô tô, xe tải, người đi bộ) trong một hình ảnh bằng cách vẽ một hộp giới hạn (bounding box) xung quanh chúng. Các phương pháp ban đầu sử dụng phương pháp “cửa sổ trượt”, trong đó một bộ phân loại sẽ được áp dụng cho hàng nghìn vùng và tỷ lệ khác nhau của một hình ảnh, một quy trình chậm đến mức không thể chấp nhận được. Kỷ nguyên hiện đại của việc phát hiện đối tượng đã được mở ra bởi học sâu, đặc biệt là các Mạng Nơ-ron Tích chập (CNN).

Các phương pháp này thường được chia thành hai loại:

1. **Các bộ phát hiện Hai giai đoạn:** (ví dụ: R-CNN, Fast R-CNN, Faster R-CNN). Các phương pháp này đầu tiên đề xuất một tập hợp các “vùng quan tâm” (regions of interest) nơi một đối tượng có khả năng xuất hiện, và sau đó một mạng ở giai đoạn thứ hai sẽ phân loại và tinh chỉnh hộp giới hạn cho mỗi đề xuất. Mặc dù có độ chính xác cao, quy trình hai giai đoạn thường gây ra chi phí tính toán đáng kể, khiến việc ứng dụng trong thời gian thực trở nên khó khăn.
2. **Các bộ phát hiện Một lần (Single-Shot Detectors - SSDs):** (ví dụ: SSD, YOLO). Các phương pháp này đã cách mạng hóa việc phát hiện đối tượng trong thời gian thực bằng cách coi nó như một bài toán hồi quy duy nhất. Mạng chỉ nhìn vào hình ảnh một lần và dự đoán trực tiếp một tập hợp các hộp giới hạn và xác suất lớp cho các hộp đó. Kiến trúc hợp nhất này làm giảm đáng kể chi phí tính toán, làm cho các mô hình thuộc họ YOLO (You Only Look Once) trở thành lựa chọn lý tưởng cho các ứng dụng như giám sát giao thông, nơi thông lượng cao (FPS) là một yêu cầu quan trọng.

### 2.1.2. Theo dõi Đối tượng: Nguyên tắc của Theo dõi-bằng-Phát hiện

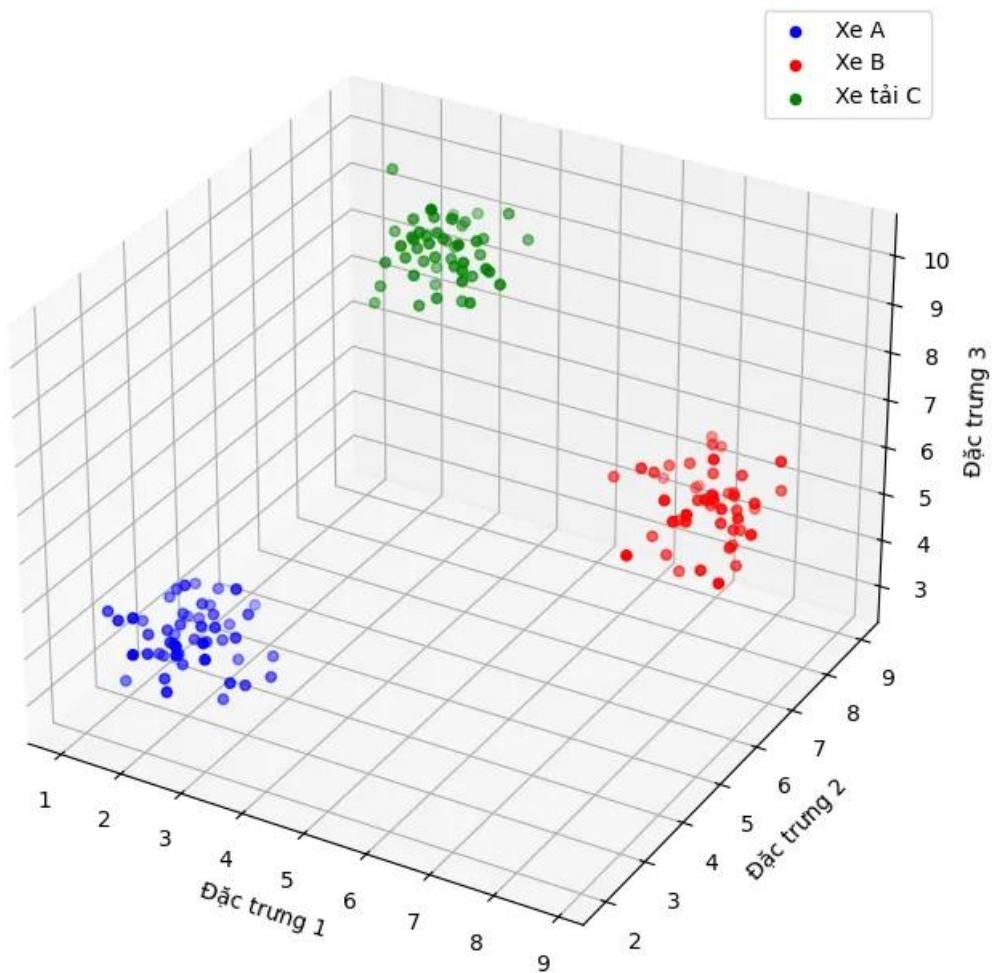
Trong khi phát hiện xác định các đối tượng trong một khung hình duy nhất, theo dõi đối tượng nhằm mục đích liên kết các phát hiện này qua một chuỗi các khung hình, gán một ID nhất quán và duy nhất cho mỗi đối tượng theo thời gian. Phương pháp chủ đạo là Theo dõi-bằng-Phát hiện. Quá trình này bao gồm:

1. **Dự đoán:** Đối với mỗi đối tượng đang được theo dõi, một mô hình chuyển động sẽ dự đoán vị trí mới của nó trong khung hình hiện tại. **Bộ lọc Kalman** là một công cụ được sử dụng rộng rãi và mạnh mẽ cho mục đích này. Nó là một thuật toán ước tính toàn phương tuyến tính sử dụng một chuỗi các phép đo được quan sát theo thời gian (chứa nhiều thông kê và các sai số khác) để đưa ra các ước tính về các biến chưa biết mà có xu hướng chính xác hơn so với các ước tính chỉ dựa trên một phép đo duy nhất. Trong theo dõi, nó dự đoán vị trí hộp giới hạn tiếp theo dựa trên vận tốc và gia tốc trong quá khứ của đối tượng.
2. **Liên kết:** Các phát hiện từ khung hình hiện tại được đối sánh với các vị trí đã được dự đoán của các đối tượng đang theo dõi. Điều này thường được xem như một bài toán gán (assignment problem), được giải quyết bằng các phương pháp như thuật toán Hungarian để tối thiểu hóa một ma trận chi phí, vốn có thể dựa trên độ trùng lặp của hộp giới hạn (IoU), độ tương đồng về ngoại hình, hoặc sự kết hợp của cả hai.

### 2.1.3. Học sâu theo Hệ mét để Tái nhận dạng

Thách thức cốt lõi của Tái nhận dạng Phương tiện (Re-ID) là học một hàm có thể xác định xem hai hình ảnh từ các camera khác nhau có mô tả cùng một phương tiện hay không. Điều này đạt được thông qua **Học sâu theo Hệ mét (Deep Metric Learning)**. Mục tiêu không phải là phân loại một phương tiện vào một danh mục được xác định trước, mà là huấn luyện một mạng nơ-ron sâu để ánh xạ một hình ảnh phương tiện vào một không gian vector nhiều chiều, được gọi là **không gian nhúng (embedding space)**.

Biểu đồ phân tán 3D các đặc trưng của phương tiện



Mạng được huấn luyện để đảm bảo rằng khoảng cách Euclidean hoặc Cosine giữa các vector nhúng của cùng một phương tiện được tối thiểu hóa, trong khi khoảng cách giữa các vector nhúng của các phương tiện khác nhau được tối đa hóa. Điều này thường được thực hiện bằng cách sử dụng **hàm mất mát Triplet (Triplet Loss)**, xem xét ba mẫu cùng một lúc: một “mỏ neo” (anchor - một hình ảnh của một phương tiện), một “dương tính” (positive - một hình ảnh khác của cùng phương tiện đó), và một “âm tính” (negative - một hình ảnh của một phương tiện khác). Hàm mất mát sẽ phạt mô hình nếu khoảng cách giữa anchor và negative không lớn hơn khoảng cách giữa anchor và positive một khoảng biên (margin) nhất định. Các vector đặc trưng (embeddings) thu được sẽ đóng vai trò như một “dấu vân tay hình ảnh” mạnh mẽ cho mỗi phương tiện.

## 2.2. Quy trình các Mô hình AI: Phân tích Chuyên sâu

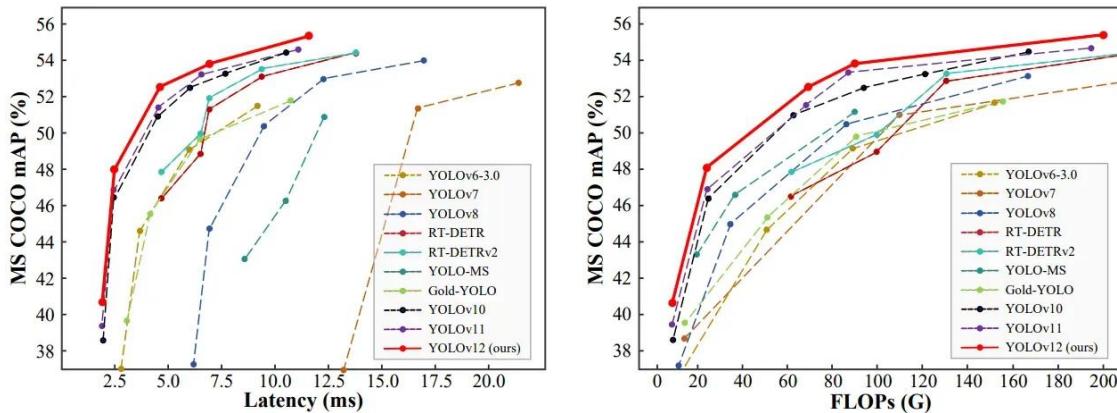
Chức năng của hệ thống được hiện thực hóa thông qua một quy trình phức tạp gồm các mô hình học sâu được kết nối với nhau, trong đó đầu ra của một giai đoạn đóng vai trò là đầu vào cho giai đoạn tiếp theo. Mỗi mô hình được lựa chọn hoặc thiết kế để trở thành công nghệ tiên tiến nhất cho tác vụ cụ thể của nó, cân bằng giữa độ chính xác và hiệu quả tính toán. Phần này sẽ phân tích chi tiết kiến trúc và lý do đằng sau mỗi mô hình cốt lõi này.

### 2.2.1. YOLOv12 cho Phát hiện: Động cơ Nhận thức

Đối với nhiệm vụ chính là phát hiện đối tượng—xác định cả phương tiện và biển số xe—dự án này sử dụng kiến trúc YOLOv12. YOLO (You Only Look Once) đại diện cho đỉnh cao của các bộ phát hiện một lần, nổi tiếng với sự cân bằng vượt trội giữa tốc độ và độ chính xác.

1. **Phân tích Kiến trúc:** Giống như các phiên bản tiền nhiệm, YOLOv12 bao gồm ba phần chính:
  1. **Mạng Xương sống (Backbone):** Đây là một Mạng Nơ-ron Tích chập (CNN) sâu chịu trách nhiệm trích xuất các bản đồ đặc trưng (feature maps) phong phú từ hình ảnh đầu vào ở nhiều tỷ lệ khác nhau. Nó tạo thành nền tảng cho khả năng hiểu hình ảnh của mô hình.
  2. **Phần Cổ (Neck):** Thành phần này, thường sử dụng các cấu trúc như Mạng Lưới Kim tự tháp Đặc trưng (Feature Pyramid Networks - FPN) và Mạng Lưới Tích hợp Đường dẫn (Path Aggregation Networks - PANet), chịu trách nhiệm hợp nhất và kết hợp các bản đồ đặc trưng từ mạng xương sống. Quá trình này cho phép mô hình phát hiện hiệu quả các đối tượng có kích thước khác nhau—from một chiếc xe tải lớn chiếm phần lớn khung hình đến một biển số xe nhỏ ở xa.
  3. **Phần Đầu (Head):** Phần đầu là giai đoạn cuối cùng thực hiện việc phát hiện thực tế. Nó lấy các bản đồ đặc trưng đã được hợp nhất từ phần cổ và áp dụng các lớp tích chập để dự đoán các hộp giới hạn cuối cùng, điểm đối tượng (độ tin cậy), và xác suất lớp cho mỗi đối tượng.
2. **Ưu điểm chính và Lý do Lựa chọn:**
  1. **Tốc độ:** Là một bộ phát hiện một lần, kiến trúc hợp nhất của YOLO vốn đã nhanh hơn các bộ phát hiện hai giai đoạn, khiến nó trở thành ứng cử viên hàng đầu cho việc xử lý video thời gian thực.
  2. **Độ chính xác:** YOLOv12, với tư cách là một sự tiến hóa của dòng YOLO, tích hợp các cải tiến kiến trúc gần đây giúp đẩy độ chính xác trung bình trung bình (mAP) của nó lên mức cạnh tranh, và trong nhiều trường hợp vượt qua, các mô hình phức tạp hơn, như được chứng minh qua hiệu năng cao trên các bộ dữ liệu chuẩn như MS COCO. Hiệu năng của mô hình YOLOv12m được tinh chỉnh của chúng tôi là một minh chứng cho khả năng này.
  3. **Khả năng mở rộng:** Dòng YOLO cung cấp một loạt các kích thước mô hình (ví dụ: nano n, small s, medium m). Điều này cho phép nhóm của chúng tôi chọn phiên bản YOLOv12m làm mô hình cơ sở mạnh mẽ để phát

hiện phương tiện và tinh chỉnh một phiên bản nhỏ hơn, chuyên dụng hơn cho nhiệm vụ có phạm vi hẹp hơn là phát hiện biển số xe, tối ưu hóa tỷ lệ hiệu năng trên chi phí cho từng tác vụ.

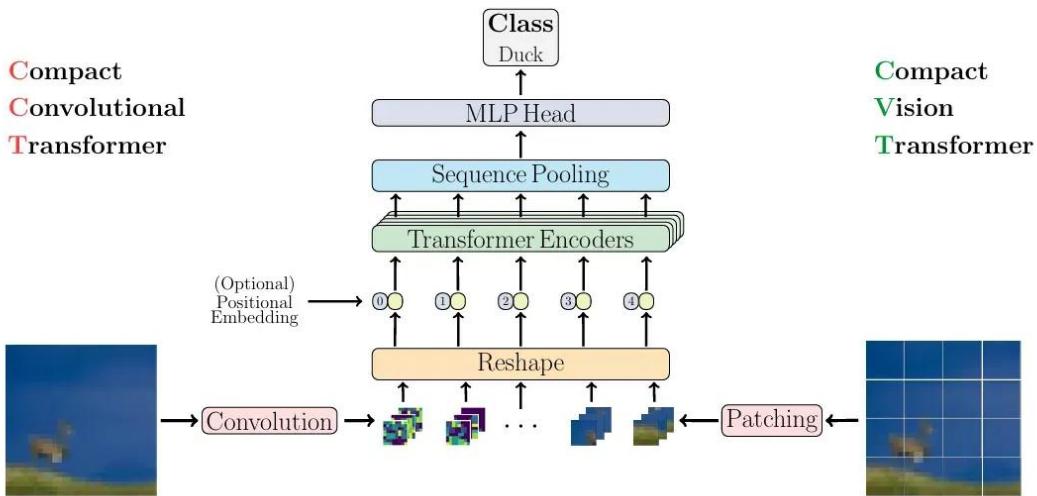


### 2.2.2. Transformer Tích chập Gọn nhẹ (CCT) để Nhận dạng

Một khi biển số xe được phát hiện và cắt ra, chuỗi các ký tự chữ và số phải được nhận dạng. Đây là một tác vụ kết hợp giữa việc trích xuất đặc trưng không gian và phân tích tuần tự. Đối với nhiệm vụ này, dự án sử dụng mô hình Transformer Tích chập Gọn nhẹ (Compact Convolutional Transformer - CCT). Kiến trúc lai (hybrid) này kết hợp một cách xuất sắc thế mạnh của hai mô hình học sâu chủ đạo:

- Phản đầu Tích chập (Convolutional Front-End):** Không giống như các mô hình Vision Transformer (ViT) tiêu chuẩn chia một hình ảnh thành các mảng ván (patch) thô, CCT đầu tiên xử lý hình ảnh đầu vào bằng một loạt các lớp tích chập. Đây là một lựa chọn thiết kế quan trọng. Các mạng CNN sở hữu một “thiên vị quy nạp” (inductive bias) mạnh mẽ đối với dữ liệu hình ảnh—chúng vốn đã thành thạo trong việc học các cấu trúc phân cấp không gian của các đặc trưng như cạnh, góc và kết cấu. Bộ token hóa tích chập này xử lý trước hình ảnh một cách hiệu quả, trích xuất các đặc trưng cục bộ có ý nghĩa và chuyển chúng thành một chuỗi các token.
- Phản cuối Transformer (Transformer Back-End):** Chuỗi các token giàu đặc trưng này sau đó được đưa vào một bộ mã hóa Transformer tiêu chuẩn. Cơ chế cốt lõi của Transformer, lớp tự chú ý (self-attention layer), vượt trội trong việc học các phụ thuộc tầm xa và các mối quan hệ ngữ cảnh trong một chuỗi. Điều này cho phép nó hiểu được mối quan hệ giữa các ký tự khác nhau trên biển số, cải thiện độ chính xác nhận dạng, đặc biệt đối với các ký tự trông giống nhau hoặc bị che khuất một phần.

Bằng cách tận dụng phản đầu CNN, CCT có thể được huấn luyện hiệu quả trên các bộ dữ liệu nhỏ hơn so với các mô hình ViT truyền thống, vốn đòi hỏi lượng dữ liệu khổng lồ để học các mẫu hình ảnh cơ bản. Điều này làm cho nó trở thành một lựa chọn rất thực tế và hiệu quả cho các tác vụ chuyên biệt như Nhận dạng Biển số xe (LPR).



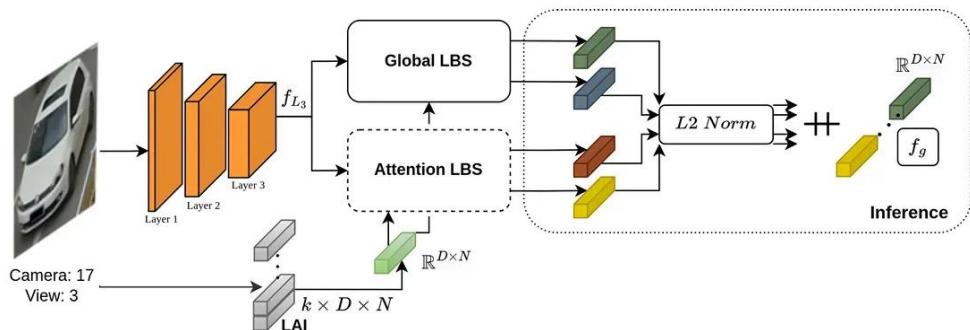
### 2.2.3. Học biểu diễn Đa nhánh để Tái nhận dạng (Re-ID)

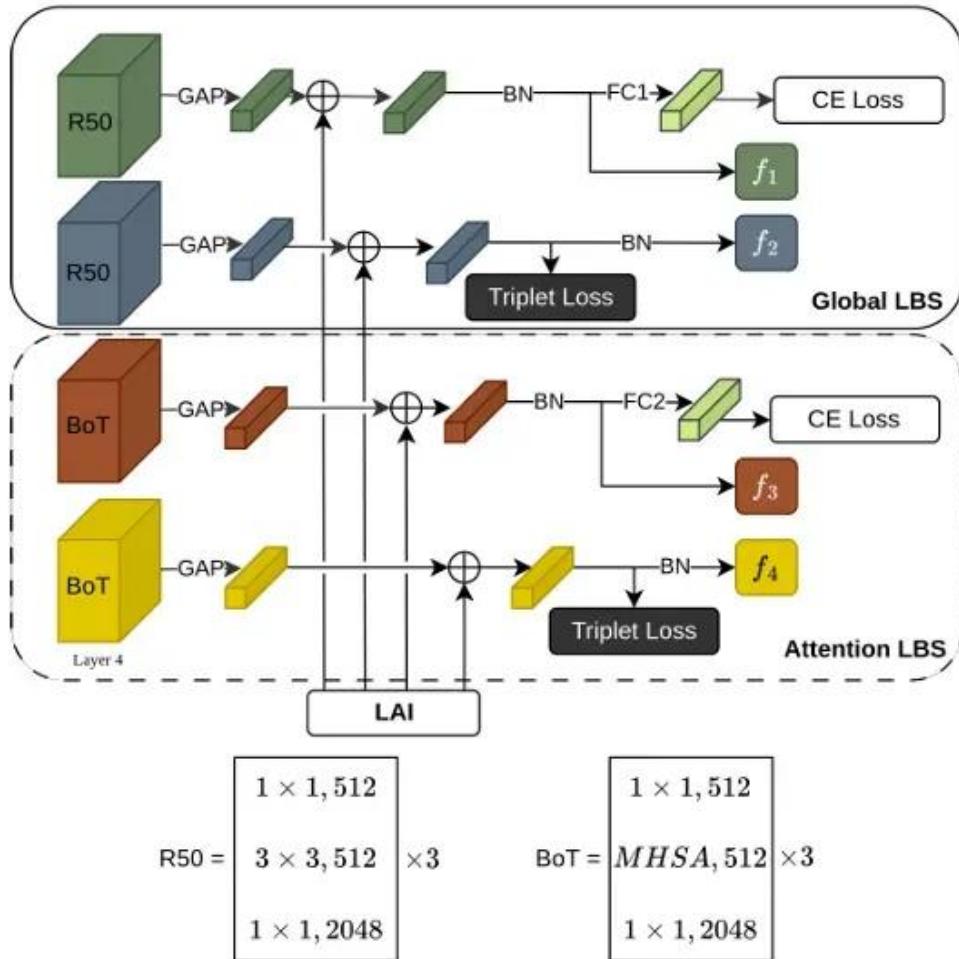
Nhiệm vụ Tái nhận dạng Phương tiện (Re-ID) là nền tảng cốt lõi cho khả năng hoạt động trên nhiều camera của hệ thống. Mục tiêu là tạo ra một vector đặc trưng (embedding) độc nhất và mạnh mẽ đến mức nó có thể hoạt động như một “dấu vân tay hình ảnh” đáng tin cậy. Để đạt được điều này, dự án triển khai một kiến trúc Học biểu diễn Đa nhánh (Multi-Branched Representation Learning) tiên tiến, như được đề xuất trong công trình “Strength in Diversity,” (Sức mạnh trong sự Đa dạng) vốn được thiết kế để nắm bắt một sự am hiểu phong phú và đa chiều về ngoại hình của mỗi phương tiện.

- **Triết lý Kiến trúc: Sức mạnh trong sự Đa dạng**
- 3. Giả thuyết trung tâm của kiến trúc này là một nhánh mạng duy nhất, dù sâu đến đâu, cũng sẽ không thể tránh khỏi việc phát triển một thiên kiến (bias) đối với một số loại đặc trưng nhất định. Để khắc phục điều này, mô hình sử dụng một thiết kế đa nhánh, song song, trong đó mỗi nhánh được khuyến khích học các biểu diễn khác biệt, bổ sung cho nhau của cùng một hình ảnh đầu vào. Vector nhúng cuối cùng là sự kết hợp (concatenation) của các đầu ra từ tất cả các nhánh, tạo ra một vector mô tả phong phú và có tính phân biệt cao hơn bất kỳ nhánh đơn lẻ nào có thể tự tạo ra.
- **Thiết kế Nhánh: Một Cách tiếp cận Lai**
- 4. Việc triển khai cụ thể của chúng tôi sử dụng sự kết hợp lai mạnh mẽ của hai phong cách kiến trúc riêng biệt cho các nhánh của nó:
  - **Nhánh Đặc trưng Toàn cục (ResNet50-IBN-a):** Nhánh này dựa trên kiến trúc ResNet50 đã được công nhận rộng rãi, được cải tiến với Chuẩn hóa Lô-Thực thể (Instance-Batch Normalization - IBN-a). Nó xử lý hình ảnh thông qua một chồng sâu các khối dư (residual blocks) để học các đặc trưng toàn cục, mạnh mẽ liên quan đến hình dáng tổng thể, màu sắc và kết cấu của phương tiện. Điều này tạo thành biểu diễn ngoại hình nền tảng.
  - **Nhánh Dựa trên Cơ chế Chú ý (BotNet):** Nhánh này thay thế một số lớp tích chập tiêu chuẩn trong một khối ResNet bằng các cơ chế Tự chú ý Đa đầu (Multi-Head Self-Attention - MHSA), lấy cảm hứng từ kiến trúc Transformer. Cơ chế tự chú ý cho phép mô hình cân nhắc tầm quan trọng

của các vùng khác nhau trong hình ảnh so với nhau. Điều này cho phép nhánh học một biểu diễn “dựa trên các bộ phận” hoặc “dựa trên mối quan hệ”, tập trung vào các chi tiết cục bộ có tính phân biệt cao như đèn pha, lưỡi tản nhiệt, logo hoặc các dấu hiệu độc nhất, và hiểu được các mối quan hệ không gian của chúng.

- **Chiến lược Phân chia Nhánh theo Hàm măt măt (LBS)**
5. Để đảm bảo các nhánh thực sự học được các đặc trưng đa dạng thay vì hội tụ về các giải pháp tương tự, chiến lược huấn luyện Phân chia Nhánh theo Hàm măt măt (Loss-Branch-Split - LBS) được sử dụng. Thay vì sử dụng một hàm măt măt kết hợp duy nhất ở cuối mạng, mỗi nhánh được gán một hàm măt măt chuyên biệt riêng trong quá trình huấn luyện. Ví dụ:
1. Một nhánh có thể được huấn luyện với **Măt măt Phân loại** (ví dụ: Cross-Entropy với Làm mịn Nhăń), thúc đẩy nó học các đặc trưng tốt cho việc xác định ID phương tiện cụ thể.
  2. Một nhánh khác có thể được huấn luyện với **Măt măt theo Hệ mét** (ví dụ: Măt măt Triplet), thúc đẩy nó học các đặc trưng vượt trội trong việc tối đa hóa khoảng cách giữa các lớp và tối thiểu hóa khoảng cách trong cùng một lớp trong không gian nhúng.
  3. Sự tách biệt chiến lược này trong các mục tiêu học tập buộc mỗi nhánh phải chuyên môn hóa, đảm bảo rằng vector đặc trưng được kết hợp cuối cùng là một tổng thể của các thông tin đa dạng, bổ sung cho nhau và có tính phân biệt cao.





#### 2.2.4. BoT-SORT để Theo dõi: Đảm bảo sự Mạch lạc theo Thời gian

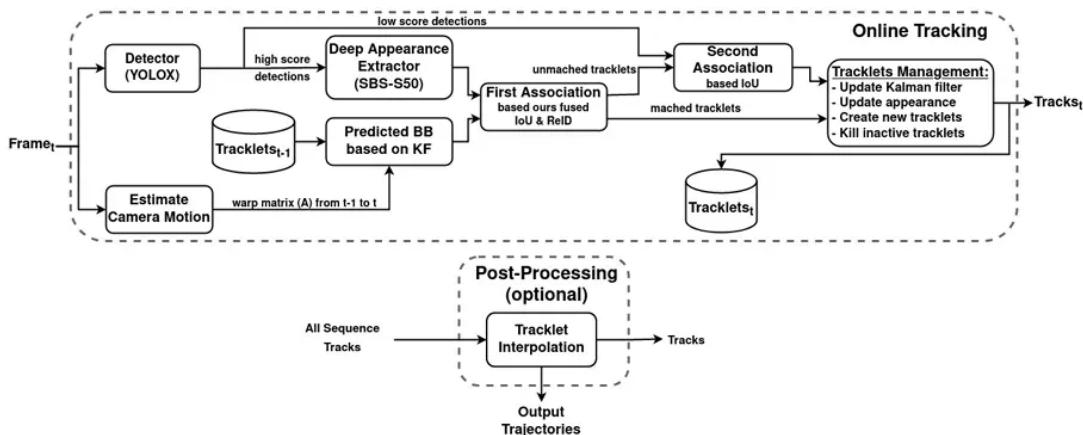
BoT-SORT (Bag of Tricks for SORT) là thuật toán chịu trách nhiệm cho thành phần “theo dõi” trong phạm vi quan sát của mỗi camera đơn lẻ. Nó được xây dựng dựa trên mô hình Theo dõi-bằng-Phát hiện (Tracking-by-Detection) nền tảng của SORT và DeepSORT nhưng tích hợp một số cải tiến quan trọng giúp nó trở nên mạnh mẽ và chính xác hơn đáng kể cho các kịch bản thực tế như giám sát giao thông.

1. **Bộ phát hiện Hiệu năng cao:** Chất lượng của việc theo dõi phụ thuộc một cách cơ bản vào chất lượng của các phát hiện. BoT-SORT được thiết kế để tích hợp liền mạch với các bộ phát hiện tiên tiến nhất. Trong hệ thống của chúng tôi, các phát hiện có độ tin cậy cao từ mô hình YOLOv12 đóng vai trò là đầu vào cho bộ theo dõi, cung cấp một điểm khởi đầu vững chắc cho quá trình liên kết.
2. **Bộ lọc Kalman với Bù trừ Chuyển động Camera (CMC):** Các Bộ lọc Kalman tiêu chuẩn giả định camera là tĩnh, chỉ mô hình hóa chuyển động của đối tượng. Tuy nhiên, camera giao thông có thể bị rung lắc hoặc có những chuyển động nhỏ. CMC giải quyết vấn đề này bằng cách đầu tiên ước tính chuyển động toàn cục của toàn bộ khung hình giữa thời điểm  $t-1$  và  $t$  (thường bằng cách theo dõi các đặc trưng nền ổn định). Chuyển động toàn cục này sau đó được trừ đi khỏi chuyển động quan sát được của các đối tượng được phát hiện, dẫn đến một ước tính chính xác hơn.

xác hơn về chuyển động thực của đối tượng. Điều này cải thiện đáng kể độ chính xác dự đoán của Bộ lọc Kalman, dẫn đến ít lỗi liên kết hơn.

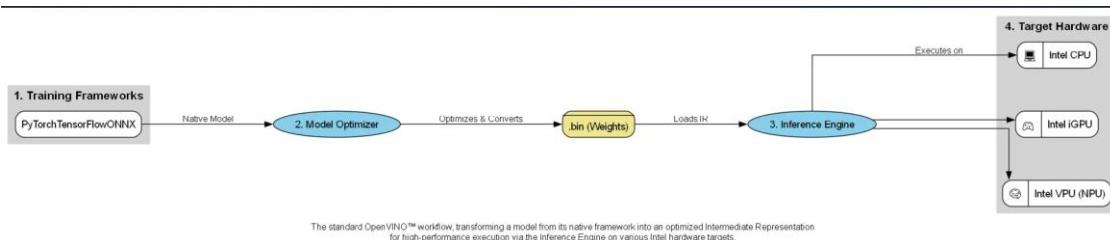
3. **Chiến lược Liên kết Tái nhận dạng Vượt trội:** Đây là cải tiến quan trọng nhất. BoT-SORT sử dụng một quy trình đối sánh hai giai đoạn một cách thông minh, kết hợp các tín hiệu chuyển động và ngoại hình:

1. **Giai đoạn 1 (Đối sánh Tái nhận dạng):** Các phát hiện có độ tin cậy cao đầu tiên được đối sánh với các đối tượng đang được theo dõi bằng cách sử dụng các đặc trưng ngoại hình được trích xuất bởi mô hình Re-ID (trong trường hợp của chúng tôi là mô hình Đa nhánh). Điều này rất hiệu quả để tái nhận diện một phương tiện sau một khoảng thời gian bị che khuất ngắn (ví dụ, đi qua phía sau một chiếc xe tải).
2. **Giai đoạn 2 (Đối sánh IoU):** Đối với bất kỳ đối tượng theo dõi và phát hiện nào còn lại chưa được đối sánh, BoT-SORT sẽ quay lại sử dụng phương pháp đối sánh IoU (Intersection over Union) cổ điển dựa trên độ trùng lặp của hộp giới hạn. Điều này “cứu” các phát hiện mà đặc trưng ngoại hình có thể bị sai lệch tạm thời (ví dụ, do lóa sáng hoặc chỉ nhìn thấy một phần) nhưng dự đoán chuyển động vẫn đáng tin cậy.
3. Chiến lược liên kết đa giai đoạn, thông minh này làm cho BoT-SORT có khả năng chống chọi đặc biệt tốt với những thách thức của các cảnh giao thông trong thế giới thực, dẫn đến ít lần chuyển đổi ID hơn và theo dõi ổn định, dài hạn hơn.



### 2.3. Bộ công cụ Intel® OpenVINO™: Trình bày Kỹ thuật

Trong khi các mô hình AI đã thảo luận trước đó cung cấp “trí thông minh” cho hệ thống, thì Bộ công cụ Intel® OpenVINO™ (Open Visual Inference and Neural Network Optimization) cung cấp “động cơ” cho nó. Đây là một bộ công cụ phần mềm toàn diện được thiết kế để giải quyết một thách thức duy nhất và cực kỳ quan trọng trong việc triển khai học sâu: tối ưu hóa các mô hình mạng nơ-ron để suy luận hiệu năng cao trên toàn bộ dải phần cứng của Intel. Triết lý của nó là cho phép các nhà phát triển huấn luyện một mô hình trong bất kỳ framework phổ biến nào và sau đó triển khai nó với hiệu suất tối đa, mà không cần phải viết mã cấp thấp, đặc thù cho phần cứng một cách thủ công.



### 2.3.2. Chi tiết các Thành phần Chính

Bộ công cụ OpenVINO™ đạt được mục tiêu của mình thông qua hai thành phần chính hoạt động song song: Bộ tối ưu hóa Mô hình (Model Optimizer) và Bộ máy Suy luận (Inference Engine).

#### 1. Bộ tối ưu hóa Mô hình (Model Optimizer):

Model Optimizer là một công cụ dòng lệnh đóng vai trò là cầu nối giữa môi trường huấn luyện của một mô hình và môi trường thực thi của OpenVINO™. Chức năng chính của nó là chuyển đổi một mô hình đã được huấn luyện từ framework gốc của nó (chẳng hạn như PyTorch, TensorFlow, hoặc ONNX) thành một định dạng được tiêu chuẩn hóa và tối ưu hóa được gọi là **Biểu diễn Trung gian (Intermediate Representation - IR)**. Đây không chỉ đơn thuần là một sự thay đổi định dạng tệp; Model Optimizer thực hiện một số tối ưu hóa quan trọng ở cấp độ đồ thị trong quá trình này:

- Chuyển đổi Độc lập với Framework:** Nó phân tích kiến trúc và trọng số của mô hình đầu vào, dịch chúng thành một cấu trúc đồ thị phổ quát.
- Tạo Đồ thị Tính:** Nó tạo ra một biểu diễn tĩnh, cố định của đồ thị tính toán. Điều này cho phép Bộ máy Suy luận biết trước toàn bộ quy trình làm việc, cho phép các tối ưu hóa sâu hơn so với các đồ thị động được sử dụng trong nhiều framework huấn luyện.
- Cắt tỉa và Hợp nhất Đồ thị:** Nó phân tích đồ thị một cách thông minh để xác định và loại bỏ các hoạt động không cần thiết cho việc suy luận (ví dụ: các hoạt động chỉ dùng trong huấn luyện như dropout hoặc tính toán gradient). Quan trọng hơn, nó thực hiện **Hợp nhất Đồ thị (Graph Fusion)**, nơi các chuỗi hoạt động (ví dụ: một lớp Tích chập theo sau bởi một lớp Chuẩn hóa Lô rồi đến một hàm kích hoạt ReLU) được kết hợp về mặt toán học thành một nhân (kernel) hợp nhất, hiệu quả hơn. Điều này làm giảm đáng kể chi phí bộ nhớ và số lượng lệnh gọi hàm, dẫn đến tăng tốc đáng kể.

Đầu ra của Model Optimizer là một cặp tệp tạo nên IR:

- tệp .xml:** Mô tả cấu trúc liên kết của mạng, bao gồm các lớp, các tham số của chúng, và các kết nối giữa chúng.
- tệp .bin:** Chứa trọng số và độ lệch của mô hình ở định dạng nhị phân nhỏ gọn.

#### 2. Bộ máy Suy luận (Inference Engine):

Inference Engine là thư viện C++ hiệu năng cao (với các ràng buộc Python - Python bindings) nhận các tệp IR và thực thi mô hình trên phần cứng đích. Đây là nơi diễn ra các

tối ưu hóa cấp thấp, đặc thù cho phần cứng. Khi một mô hình được tải lên một thiết bị đích (ví dụ: CPU), Inference Engine sẽ:

1. **Lựa chọn các Nhân (Kernel) Tối ưu:** Nó duy trì một thư viện các hàm cấp thấp (kernel) được tối ưu hóa cao cho mỗi loại lớp mạng nơ-ron. Nó tự động chọn kernel hiệu quả nhất cho phần cứng cụ thể được phát hiện.
2. **Tận dụng Vector hóa:** Đối với CPU của Intel, tính năng mạnh mẽ nhất của nó là việc sử dụng sâu các tập lệnh vector tiên tiến, chẳng hạn như AVX2 (Advanced Vector Extensions 2) và AVX512. Các tập lệnh này cho phép CPU thực hiện cùng một phép toán (ví dụ: phép nhân) trên nhiều điểm dữ liệu cùng một lúc (Một Lệnh, Nhiều Dữ liệu - SIMD). Các phép nhân ma trận và tích chập ở trung tâm của học sâu hoàn toàn phù hợp với loại hình song song hóa này. Các framework tiêu chuẩn như PyTorch có thể không khai thác hết các tập lệnh này để suy luận trên CPU, nhưng Inference Engine được thiết kế rõ ràng để làm điều đó, đây là nguồn gốc chính cho lợi thế hiệu năng vượt trội của nó trên CPU.
3. **Quản lý Thực thi Bất đồng bộ:** Nó cung cấp một API cho việc suy luận bất đồng bộ, cho phép ứng dụng gửi một yêu cầu suy luận mới mà không cần chờ yêu cầu trước đó hoàn thành. Điều này cho phép phát triển các ứng dụng dạng ống (pipelined) hiệu quả cao có thể khai thác triệt để phần cứng bằng cách chồng chéo các công việc tiền xử lý dữ liệu, suy luận và hậu xử lý.

### 2.3.3. Các Kỹ thuật Tối ưu hóa Cơ bản: Một cái nhìn Cận cảnh

Ngoài các thành phần cấp cao, lợi ích về hiệu năng của OpenVINO bắt nguồn từ các kỹ thuật cụ thể. Mặc dù dự án của chúng tôi tập trung vào độ chính xác FP32, việc hiểu các kỹ thuật này là rất quan trọng để đánh giá đúng sức mạnh của bộ công cụ.

1. **Hợp nhất Đồ thị (Graph Fusion):** Như đã đề cập, đây là một tối ưu hóa tĩnh quan trọng. Hãy xem xét một chuỗi phổ biến trong một mạng CNN: Conv -> Bias -> ReLU. Trong một framework tiêu chuẩn, điều này có thể liên quan đến ba lệnh gọi tính toán riêng biệt. Model Optimizer có thể hợp nhất chúng thành một hoạt động FusedConvBiasReLU duy nhất. Điều này làm giảm chi phí khởi chạy nhiều kernel và cải thiện tính cục bộ của dữ liệu, giữ các kết quả trung gian trong các cấp bộ nhớ đệm nhanh hơn của CPU.
2. **Giảm độ chính xác (Lượng tử hóa - Quantization):** Mặc dù không phải là trọng tâm của việc đo lường hiệu năng trong dự án này, lượng tử hóa là một trong những tính năng mạnh mẽ nhất của OpenVINO để tối ưu hóa sâu hơn. Nó liên quan đến việc chuyển đổi trọng số và các giá trị kích hoạt của mô hình từ số dấu phẩy động 32-bit (FP32) sang số nguyên 8-bit (INT8) có độ chính xác thấp hơn.
  1. **Tại sao nó nhanh hơn:** Các phép toán số nguyên nhanh hơn đáng kể và tiêu thụ ít năng lượng hơn trên hầu hết các bộ xử lý so với các phép toán dấu phẩy động.
  2. **Cách thức hoạt động:** Quá trình này bao gồm việc xác định phạm vi (giá trị min/max) của dữ liệu FP32 cho mỗi lớp và ánh xạ phạm vi này tới 256 giá trị có thể có của một số INT8. Điều này đòi hỏi một bước hiệu chỉnh (calibration) sử dụng một bộ dữ liệu đại diện.

3. **Lợi ích:** Điều này dẫn đến việc giảm 4 lần kích thước mô hình về mặt lý thuyết (giảm dung lượng bộ nhớ và yêu cầu băng thông) và có thể cung cấp thêm tốc độ suy luận nhanh hơn 2-3 lần, thường chỉ với một sự sụt giảm tối thiểu về độ chính xác của mô hình.

Bằng cách chọn OpenVINO™, dự án này đã tận dụng một cách chiến lược một bộ công cụ được thiết kế đặc biệt để khai thác hiệu suất tính toán tối đa từ các CPU Intel phổ thông, cung cấp một giải pháp trực tiếp và mạnh mẽ cho vấn đề nút thắt cổ chai về hiệu năng đã được xác định trong Chương 1.

### CHƯƠNG 3. PHÂN TÍCH, THIẾT KẾ VÀ TRIỂN KHAI HỆ THỐNG

Chương này chi tiết hóa quá trình kỹ thuật đằng sau việc phát triển hệ thống MC-MOT. Nó chuyển đổi các khái niệm lý thuyết từ Chương 2 thành một thiết kế kiến trúc cụ thể và một triển khai có chức năng. Chúng tôi sẽ bắt đầu bằng việc định nghĩa chính thức các yêu cầu của hệ thống, sau đó trình bày kiến trúc cấp cao, và tiếp theo là phân tích chi tiết việc triển khai của từng mô-đun cốt lõi, từ thu thập dữ liệu đến quy trình tối ưu hóa bằng OpenVINO™ cuối cùng.

#### 3.1. Đặc tả Yêu cầu Hệ thống

Một hệ thống mạnh mẽ bắt đầu từ việc định nghĩa rõ ràng các mục tiêu và ràng buộc của nó. Các yêu cầu cho dự án này được phân thành hai loại riêng biệt: yêu cầu chức năng và yêu cầu phi chức năng.

##### 3.1.1. Yêu cầu Chức năng

Yêu cầu chức năng định nghĩa những gì hệ thống phải làm. Các chức năng cốt lõi bắt buộc đối với hệ thống MC-MOT này là:

1. **FR-1: Phát hiện Phương tiện:** Hệ thống sẽ phát hiện và định vị các phương tiện (cụ thể là ô tô, xe tải và xe buýt) từ một khung hình video hoặc hình ảnh đầu vào cho trước.
2. **FR-2: Phát hiện Biển số xe:** Đối với mỗi phương tiện được phát hiện, hệ thống phải có khả năng phát hiện vị trí của biển số xe.
3. **FR-3: Nhận dạng Biển số xe:** Hệ thống sẽ thực hiện Nhận dạng Ký tự Quang học (OCR) trên vùng biển số xe đã được phát hiện để phiên âm nó thành một chuỗi văn bản.
4. **FR-4: Theo dõi trong một Camera (Intra-Camera):** Hệ thống sẽ gán và duy trì một ID theo dõi cụ bộ nhất quán cho mỗi phương tiện duy nhất miễn là nó vẫn còn trong tầm quan sát của một camera.
5. **FR-5: Trích xuất Đặc trưng để Tái nhận dạng:** Hệ thống sẽ tạo ra một vector đặc trưng (embedding) nhiều chiều, đóng vai trò như một chữ ký hình ảnh độc nhất cho mỗi phương tiện được phát hiện.
6. **FR-6: Tái nhận dạng giữa các Camera (Inter-Camera):** Hệ thống sẽ sử dụng các vector đặc trưng đã trích xuất để đối sánh các phương tiện qua các góc nhìn camera khác nhau, không giao nhau, gán một ID toàn cầu nhất quán cho mỗi phương tiện duy nhất trên toàn bộ mạng lưới.

### 3.1.2. Yêu cầu Phi chức năng

Yêu cầu phi chức năng định nghĩa các phẩm chất và ràng buộc mà hệ thống phải hoạt động theo. Đây là những yêu cầu quan trọng để thiết lập các tiêu chí thành công của dự án.

7. **NFR-1: Hiệu năng:** Hệ thống cuối cùng, đã được tối ưu hóa, phải đạt được thông lượng suy luận cao. Mục tiêu là vượt quá **15 Khung hình trên Giây (FPS)** cho quy trình phát hiện và nhận dạng từ đầu đến cuối khi chạy trên một CPU Intel tiêu chuẩn, chứng tỏ khả năng tồn tại của nó cho các ứng dụng thời gian thực.
8. **NFR-2: Độ chính xác:** Mỗi thành phần mô hình AI phải đạt được mức độ chính xác cao trong tác vụ tương ứng của nó. Điều này bao gồm độ chính xác trung bình trung bình (mAP) cao cho các mô hình phát hiện và độ chính xác nhận dạng ký tự cao cho mô hình OCR.
9. **NFR-3: Tính mạnh mẽ:** Hệ thống phải đủ mạnh mẽ để xử lý các thách thức của dữ liệu giao thông thực tế, bao gồm các biến thể về ánh sáng, điều kiện thời tiết, mức độ che khuất vừa phải, và các góc camera khác nhau.
10. **NFR-4: Tính di động & Hiệu quả về Chi phí:** Hệ thống phải được thiết kế để chạy hiệu quả trên phân cứng CPU phổ thông, loại bỏ sự phụ thuộc vào các GPU chuyên dụng, đắt tiền và qua đó đảm bảo chi phí triển khai và vận hành thấp hơn.

### 3.2. Kiến trúc Tổng thể của Hệ thống

Để đáp ứng các yêu cầu đã đặc tả, một kiến trúc dạng ống (pipeline) đa giai đoạn, theo mô-đun đã được thiết kế. Thiết kế này đảm bảo một luồng dữ liệu logic và cho phép phát triển, kiểm thử và tối ưu hóa độc lập từng thành phần. Toàn bộ quá trình có thể được hình dung như một cái phễu, nơi dữ liệu hình ảnh thô dần được tinh chỉnh thành thông tin có cấu trúc và có thể hành động.

Quy trình làm việc của hệ thống bao gồm các giai đoạn tuần tự sau:

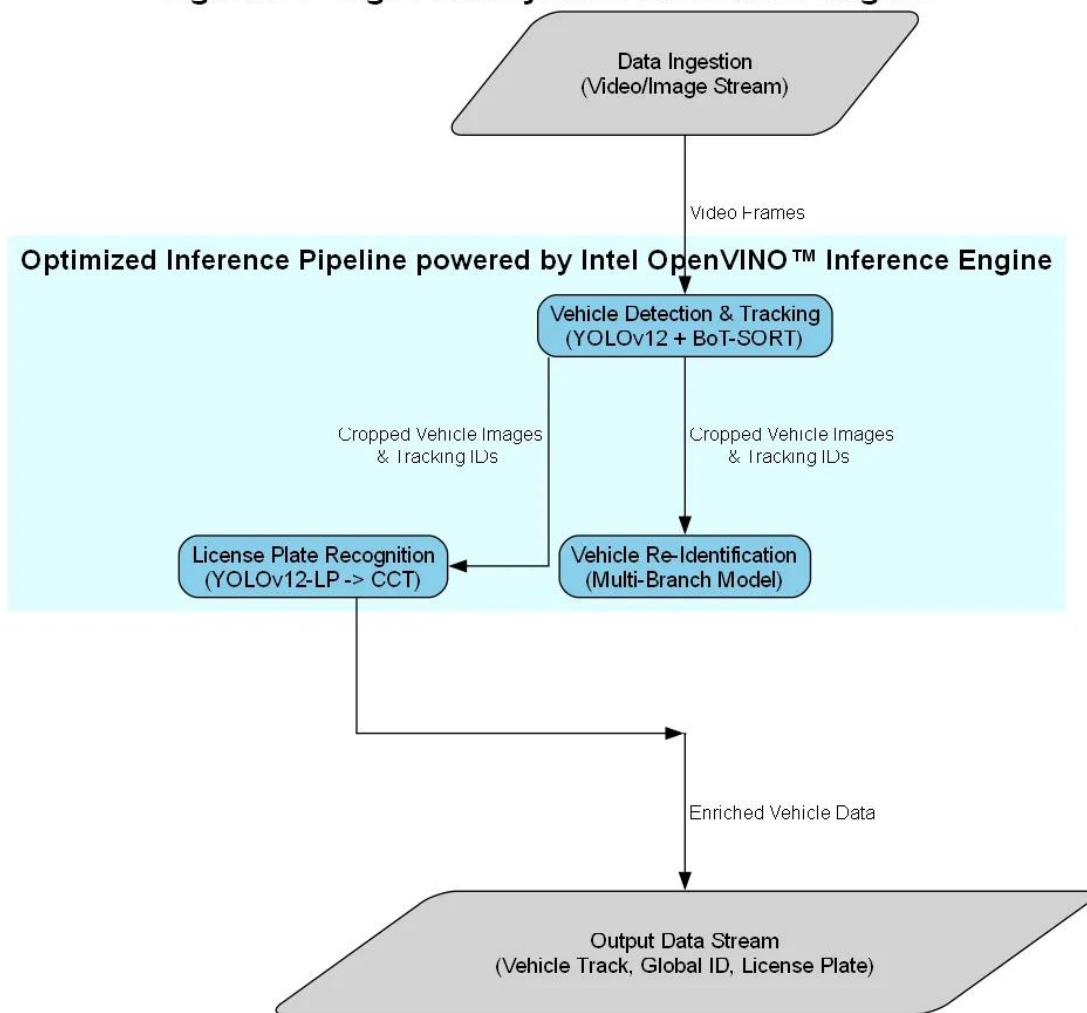
1. **Tiếp nhận Dữ liệu:** Hệ thống nhận đầu vào thô, có thể là một hình ảnh duy nhất, một tệp video đã ghi trước, hoặc một luồng hình ảnh trực tiếp được thu thập từ mạng lưới camera của TP.HCM.
2. **Phát hiện & Theo dõi Phương tiện:** Ở giai đoạn này, một bộ phát hiện chính (YOLOv12) xác định tất cả các phương tiện trong khung hình. Một bộ theo dõi (BoT-SORT) được áp dụng ngay lập tức để gán và quản lý các ID theo dõi cục bộ (trong một camera).
3. **Phát hiện & Nhận dạng Biển số xe:** Đối với mỗi phương tiện được theo dõi, một bộ phát hiện thứ cấp (YOLOv12-LP) định vị biển số xe. Vùng biển số được cắt ra sau đó được chuyển đến mô hình OCR (CCT) để đọc văn bản. Văn bản được nhận dạng sẽ được liên kết với ID theo dõi của phương tiện.
4. **Tái nhận dạng Phương tiện:** Đồng thời, hình ảnh được cắt ra của mỗi phương tiện đang được theo dõi được đưa qua mô hình Tái nhận dạng Đa nhánh để tạo ra một vector đặc trưng duy nhất.
5. **Liên kết Đa Camera:** Vector đặc trưng, cùng với văn bản biển số xe đã nhận dạng, được sử dụng để truy vấn một cơ sở dữ liệu trung tâm. Sử dụng Độ tương đồng Cosine và đối sánh dựa trên luật, hệ thống xác định xem phương tiện này đã

được nhìn thấy trước đó bởi một camera khác hay chưa. Nếu tìm thấy một kết quả khớp, nó sẽ được gán ID Toàn cầu hiện có; nếu không, một ID Toàn cầu mới sẽ được tạo ra.

6. **Dữ liệu Đầu ra:** Đầu ra cuối cùng là một luồng dữ liệu có cấu trúc chứa hộp giới hạn của phương tiện, ID theo dõi cục bộ, ID Toàn cầu vĩnh viễn, và văn bản biển số xe đã được nhận dạng.

Toàn bộ quy trình suy luận (Giai đoạn 2, 3, và 4) được thực thi thông qua **Bộ máy Suy luận Intel OpenVINO™** để đạt được các yêu cầu hiệu năng quan trọng.

**Figure 3.1 - High-Level System Architecture Diagram**



### 3.3. Mô-đun 1: Thu thập và Chuẩn bị Dữ liệu

Hiệu năng của bất kỳ hệ thống học sâu nào cũng gắn bó chặt chẽ với chất lượng và số lượng dữ liệu được sử dụng để huấn luyện nó. Dự án này đã tận dụng một chiến lược dữ liệu hai mũi nhọn: thứ nhất, bằng cách tìm nguồn cung cấp các bộ dữ liệu công khai không lò để xây dựng các mô hình nền tảng mạnh mẽ, và thứ hai, bằng cách phát triển một công cụ tùy chỉnh để thu thập dữ liệu thực tế, phù hợp với bối cảnh trực tiếp từ môi trường triển khai mục tiêu—mạng lưới camera giao thông của Thành phố Hồ Chí Minh.

### 3.3.1. Nguồn Dữ liệu: Một Cách tiếp cận Đa diện

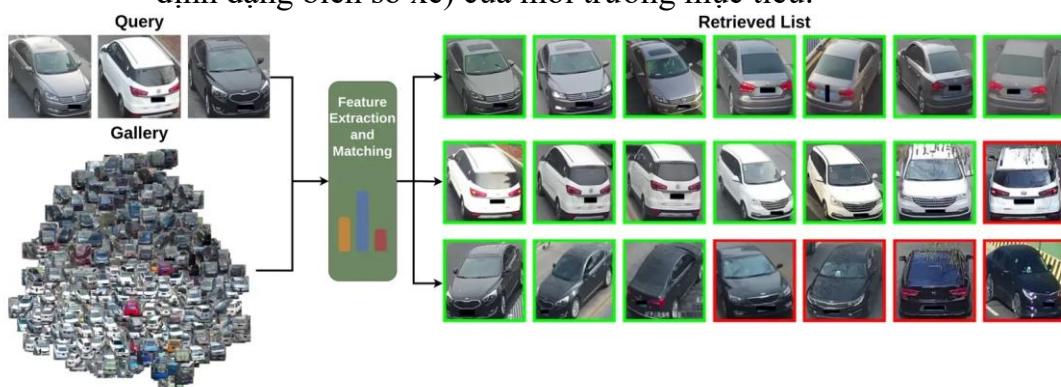
Để đảm bảo các mô hình được huấn luyện trên một tập hợp các ví dụ đa dạng và toàn diện, dữ liệu đã được tổng hợp từ một số nguồn chính:

#### 1. Các Bộ dữ liệu Đánh giá Công khai:

1. **Vehicle-1M:** Một bộ dữ liệu khổng lồ chứa hơn một triệu hình ảnh của gần 50.000 thực thể phương tiện, cung cấp một nền tảng vững chắc để huấn luyện mô hình Tái nhận dạng phương tiện với rất nhiều loại xe, màu sắc và góc nhìn khác nhau.
2. **VRIC (Vehicle Re-Identification in Context):** Bộ dữ liệu này cực kỳ quan trọng vì nó cung cấp hình ảnh từ một mạng lưới camera quy mô thành phố trong thực tế, mang lại các biến thể thực tế về ánh sáng, độ phân giải và góc độ, là những yếu tố cần thiết để huấn luyện một mô hình Tái nhận dạng mạnh mẽ.
3. **Roboflow Universe:** Một bộ sưu tập các bộ dữ liệu chuyên biệt, do người dùng đóng góp đã được sử dụng, đặc biệt cho nhiệm vụ phát hiện biển số xe. Các bộ dữ liệu như license-plate-w8chc và license-plate-7zrdm đã cung cấp một nguồn phong phú các biển số xe đã được gán nhãn từ nhiều vùng và điều kiện khác nhau, rất cần thiết để tinh chỉnh một bộ phát hiện có độ chính xác cao.

#### 2. Mạng lưới Camera Giao thông Thành phố Hồ Chí Minh:

1. Dữ liệu phù hợp nhất với bối cảnh được lấy trực tiếp từ cổng thông tin công khai của Sở Giao thông Vận tải TP.HCM. Mạng lưới này bao gồm hàng trăm camera được đặt một cách chiến lược tại các giao lộ quan trọng trên toàn thành phố. Dữ liệu này không chỉ quan trọng cho việc kiểm thử và xác thực mà còn có tiềm năng để tinh chỉnh các mô hình cho phù hợp với các đặc điểm hình ảnh cụ thể (ví dụ: ánh sáng, loại camera, loại phương tiện, định dạng biển số xe) của môi trường mục tiêu.



### 3.3.2. Tự động Thu thập Dữ liệu (data\_capture.py)

Để thu thập dữ liệu một cách có hệ thống từ mạng lưới camera TP.HCM, một kịch bản (script) thu thập dữ liệu bất đồng bộ tùy chỉnh (data\_capture.py) đã được phát triển. Công cụ này được thiết kế để đạt hiệu quả và khả năng mở rộng.

1. **Chức năng Cốt lõi:** Kịch bản đọc một danh sách siêu dữ liệu camera từ một tệp Excel (DANH\_SACH\_CAMERA\_DAY\_DU\_FINAL.xlsx), chứa ID camera, tọa độ GPS và URL ảnh chụp nhanh (snapshot).
2. **Thu thập có Mục tiêu:** Một tính năng chính của kịch bản là khả năng thực hiện thu thập dữ liệu có mục tiêu. Thay vì tải xuống hình ảnh từ tất cả các camera có sẵn, nó đầu tiên xác định một điểm địa lý quan tâm (trong trường hợp này là tọa độ của Dinh Độc Lập). Sử dụng công thức haversine để tính toán khoảng cách cung tròn lớn, nó lựa chọn một cách có lập trình 200 camera gần nhất với điểm trung tâm này, đảm bảo dữ liệu thu thập được gom cụm về mặt địa lý và phù hợp cho các kịch bản theo dõi đa camera.
3. **Tải xuống Bất đồng bộ:** Để tối đa hóa tốc độ và hiệu quả tải xuống, kịch bản sử dụng các thư viện asyncio và aiohttp của Python. Điều này cho phép nó gửi hàng trăm yêu cầu HTTP đến các URL ảnh chụp nhanh của camera một cách đồng thời, thay vì chờ từng lượt tải xuống hoàn thành một cách tuần tự. Cách tiếp cận bất đồng bộ này rất quan trọng để có được một ảnh chụp gần như đồng thời của toàn bộ khu vực mục tiêu theo các khoảng thời gian đều đặn (ví dụ: mỗi 7 giây).
4. **Tổ chức Dữ liệu:** Các hình ảnh được chụp sẽ được lưu trong một cấu trúc thư mục có tổ chức: captured\_images/[YYYY-MM-DD\_HH-MM-SS]/[CameraID].jpg. Cách tổ chức theo thứ tự thời gian và được đánh chỉ mục theo nguồn này là cân thiết cho việc phân tích sau này và để tái hiện lại các kịch bản giao thông cho việc kiểm thử hệ thống.

### 3.3.3. Xử lý và Hợp nhất Dữ liệu

Sau khi được tải xuống, dữ liệu thô cần được xử lý thêm trước khi có thể sử dụng để huấn luyện. Các kịch bản phụ trợ đã được tạo ra để quản lý quy trình này:

1. **Thống nhất Nhãn:** Các bộ dữ liệu từ các nguồn khác nhau thường có các nhãn lớp khác nhau (ví dụ: ‘car’, ‘truck’, ‘sedan’). Một kịch bản (data/VehicleDataset/process\_data.py) đã được sử dụng để duyệt qua các tệp nhãn và chuẩn hóa chúng, ví dụ, bằng cách ánh xạ tất cả các lớp liên quan đến phương tiện về một lớp vehicle duy nhất để huấn luyện một bộ phát hiện phương tiện tổng quát.
2. **Gộp các Bộ dữ liệu:** Các bộ dữ liệu khác nhau cho việc phát hiện phương tiện đã được gộp lại thành một bộ dữ liệu huấn luyện thống nhất, tạo ra một cấu trúc thư mục duy nhất với các thư mục con train/, valid/, và test/, mỗi thư mục chứa các thư mục images/ và labels/ theo định dạng YOLO tiêu chuẩn. Bộ dữ liệu đã được hợp nhất này sau đó được sử dụng để huấn luyện mô hình một cách mạnh mẽ.

Cách tiếp cận tỉ mỉ và đa diện này đối với việc thu thập và chuẩn bị dữ liệu đã đảm bảo rằng giai đoạn huấn luyện mô hình tiếp theo được xây dựng trên một nền tảng vững chắc của dữ liệu chất lượng cao, đa dạng và phù hợp với bối cảnh thực tế.

### 3.4. Mô-đun 2: Phát hiện Phương tiện và Theo dõi trong một Camera

Mô-đun này tạo thành lớp nền tảng cho các khả năng nhận thức của hệ thống. Trách nhiệm chính của nó là trả lời hai câu hỏi cơ bản cho mỗi khung hình của một video: “Có những đối tượng nào?” và “Những đối tượng cụ thể này đã ở đâu trước đây?” Điều này

được thực hiện bằng cách kết hợp một bộ phát hiện đối tượng tiên tiến nhất với một thuật toán theo dõi mạnh mẽ.

### 3.4.1. Huấn luyện Bộ phát hiện Phương tiện YOLOv12

Cốt lõi của hệ thống phát hiện là một mô hình YOLOv12, cụ thể là biến thể kích thước trung bình (yolo12m), được chọn vì sự cân bằng tuyệt vời giữa độ chính xác cao và chi phí tính toán có thể quản lý được. Mặc dù các mô hình YOLOv12 đã được huấn luyện trước có khả năng rất cao, việc tinh chỉnh trên một bộ dữ liệu tùy chỉnh, chuyên biệt theo lĩnh vực là rất quan trọng để đạt được hiệu năng tối đa trong một môi trường cụ thể.

1. **Bộ dữ liệu Huấn luyện:** Mô hình được huấn luyện trên bộ dữ liệu hợp nhất được mô tả trong Mục 3.3.3, bao gồm sự kết hợp đa dạng của các hình ảnh từ các bộ dữ liệu công khai và, có thể, các mẫu từ mạng lưới camera của TP.HCM. Sự đa dạng này đảm bảo mô hình học cách nhận dạng phương tiện dưới một loạt các điều kiện, bao gồm ánh sáng, thời tiết, góc camera khác nhau, và các trường hợp bị che khuất một phần.
2. **Kịch bản Huấn luyện và các Siêu tham số (models/modelVehicle/train.py):**
3. Quá trình huấn luyện được quản lý bởi một kịch bản chuyên dụng, tận dụng framework Ultralytics. Các siêu tham số chính đã được cấu hình để tối ưu hóa quá trình học:
  1. model: yolo12m.pt - Chỉ định các trọng số đã được huấn luyện trước để bắt đầu.
  2. data: Đường dẫn đến tệp data.yaml của bộ dữ liệu phương tiện hợp nhất.
  3. epochs: 20 - Số lượng epoch huấn luyện đủ để mô hình hội tụ trên bộ dữ liệu tùy chỉnh.
  4. batch: 16 - Kích thước lô được chọn để tối đa hóa việc sử dụng VRAM trên phần cứng có sẵn.
  5. imgsz: 640 - Kích thước hình ảnh đầu vào được thay đổi thành 640x640 pixel, một kích thước tiêu chuẩn mang lại sự cân bằng tốt giữa chi tiết và tốc độ xử lý.
  6. cache: True - Tùy chọn này lưu trữ hình ảnh vào RAM, giúp tăng tốc đáng kể quá trình huấn luyện sau epoch đầu tiên bằng cách giảm các nút thắt cỗ chai I/O của đĩa cứng.
  7. export\_onnx: True - Quan trọng nhất, cờ này được đặt để tự động xuất các trọng số của mô hình có hiệu năng tốt nhất sang định dạng ONNX sau khi hoàn thành huấn luyện. Tệp ONNX này là sản phẩm thiết yếu cần thiết cho việc chuyển đổi sang định dạng IR của OpenVINO™ sau này.
4. **Môi trường Huấn luyện:** Quá trình huấn luyện được thực hiện trên một GPU hiệu năng cao (ví dụ: NVIDIA RTX 4090, như được chỉ ra trong tài liệu) để hoàn thành quá trình tính toán nặng trong một khung thời gian hợp lý. Mô hình kết quả (best.pt và best.onnx) là một bộ phát hiện chuyên dụng được tối ưu hóa cao cho nhiệm vụ phát hiện phương tiện trong các cảnh giao thông đô thị.

### 3.4.2. Tích hợp BoT-SORT để Theo dõi Ốn định trong một Camera

Một khi mô hình YOLOv12 có thể phát hiện chính xác các phương tiện trong một khung hình duy nhất, thuật toán BoT-SORT được sử dụng để liên kết các phát hiện này qua các khung hình liên tiếp, thiết lập việc theo dõi ổn định.

- **Tích hợp với Thư viện Supervision:** Việc triển khai logic theo dõi được tinh giản hóa thông qua việc sử dụng thư viện supervision của Python. Thư viện này cung cấp các lớp trừu tượng hóa cấp cao cho các tác vụ thị giác máy tính phổ biến, bao gồm một giao diện gọn gàng cho bộ theo dõi BoT-SORT.
- **Quy trình Theo dõi:** Quá trình này, như được thấy trong kịch bản output/Vehicle\_and\_License.py, tuân theo mô hình Theo dõi-bằng-Phát hiện cổ điển:
  - **Tiếp nhận Khung hình:** Một khung hình được đọc từ nguồn video.
  - **Phát hiện:** Mô hình YOLOv12 đã được tinh chỉnh được sử dụng để tạo ra một tập hợp các phát hiện hộp giới hạn cho khung hình hiện tại.
  - **Định dạng các Phát hiện:** Các kết quả thu từ mô hình được chuyển đổi thành một đối tượng sv.Detections, một cấu trúc dữ liệu được tiêu chuẩn hóa do thư viện supervision cung cấp.
  - **Cập nhật Bộ theo dõi:** Bước cốt lõi liên quan đến việc chuyển đổi từ sv.Detections sang phương thức byte\_tracker.update\_with\_detections(). Thuật toán BoT-SORT bên trong phương thức này thực hiện logic liên kết quan trọng:
    1. Nó sử dụng Bộ lọc Kalman nội bộ để dự đoán vị trí mới của các đối tượng đang được theo dõi.
    2. Nó thực hiện quy trình đối sánh hai giai đoạn (đầu tiên bằng Re-ID/ngoài hình, sau đó bằng IoU) để liên kết các phát hiện mới với các đối tượng đã được dự đoán.
    3. Nó quản lý vòng đời của các đối tượng theo dõi: tạo đối tượng mới cho các phát hiện chưa được đối sánh, cập nhật đối tượng hiện có với vị trí mới, và xóa các đối tượng đã bị mất dấu trong một số lượng khung hình nhất định (được xác định bởi track\_buffer).
  - **Đầu ra:** Phương thức update\_with\_detections() trả về một đối tượng sv.Detections mới, trong đó mỗi phát hiện giờ đây được bổ sung một tracker\_id duy nhất. tracker\_id này giữ nguyên cho cùng một phương tiện miễn là nó còn hiển thị trước camera.

Sự kết hợp giữa một bộ phát hiện có độ chính xác cao và một thuật toán theo dõi mạnh mẽ tạo ra một mô-đun mạnh mẽ, có khả năng xác định và theo dõi đáng tin cậy mọi phương tiện trong tầm quan sát của một camera duy nhất, cung cấp nền tảng thiết yếu mà trên đó tất cả các phân tích đa camera sau này được xây dựng.

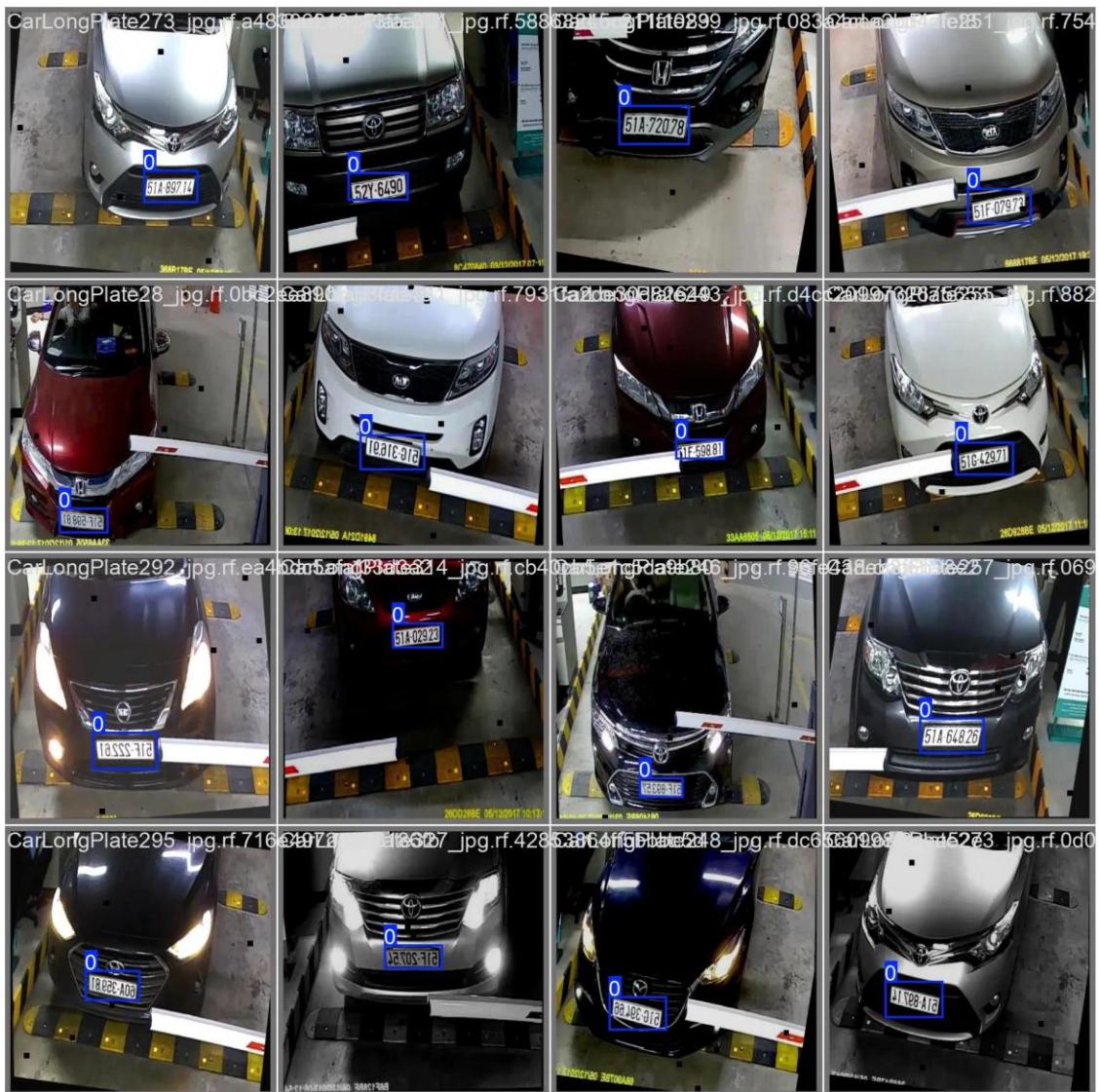
### 3.5. Mô-đun 3: Nhận dạng Biển số xe (LPR)

Một khi một phương tiện đã được theo dõi một cách đáng tin cậy, bước hợp lý tiếp theo là trích xuất định danh duy nhất của nó. Mô-đun Nhận dạng Biển số xe (LPR) là một quy trình phụ chuyên biệt gồm hai giai đoạn được thiết kế chính xác cho mục đích này. Nó đầu tiên định vị biển số xe trên phương tiện và sau đó phiên âm các ký tự trong vùng đó.

### 3.5.1. Tinh chỉnh YOLOv12 để Phát hiện Biển số xe

Trong khi một bộ phát hiện phương tiện tổng quát được huấn luyện để tìm ô tô, cần có một bộ phát hiện chuyên dụng để tìm chính xác vùng biển số xe nhỏ hơn và phức tạp hơn nhiều. Vì mục đích này, một mô hình YOLOv12n (nano) riêng biệt đã được huấn luyện. Việc lựa chọn biến thể ‘nano’ nhỏ hơn là một sự tối ưu hóa có chủ đích; phát hiện biển số xe là một tác vụ ít phức tạp hơn so với việc phát hiện các loại phương tiện đa dạng, và một mô hình nhỏ hơn cung cấp tốc độ suy luận nhanh hơn với độ chính xác đủ dùng.

1. **Bộ dữ liệu Huấn luyện:** Mô hình được tinh chỉnh trên một bộ dữ liệu được tuyển chọn, tổng hợp từ Roboflow Universe, như được chỉ định trong tệp config/global\_config.yml. Bộ dữ liệu này bao gồm hàng nghìn hình ảnh với các hộp giới hạn được gán nhãn chặt chẽ dành riêng cho biển số xe, bao gồm nhiều góc độ, điều kiện ánh sáng và thiết kế biển số khác nhau.
2. **Quy trình Huấn luyện (models/modelLicensePlate/train.py):**
3. Quy trình huấn luyện tương tự như quy trình của bộ phát hiện phương tiện, sử dụng một kịch bản tương tự và framework Ultralytics. Sự khác biệt chính là bộ dữ liệu, được trỏ đến tệp data.yaml chuyên dụng cho biển số xe.
  1. model: yolo12n.pt - Bắt đầu với mô hình đã được huấn luyện trước nhỏ nhất để đạt hiệu quả tối đa.
  2. epochs: 100 - Số lượng epoch cao hơn đã được sử dụng để cho phép mô hình chuyên môn hóa và đạt được độ chính xác cao trong tác vụ chi tiết này.
  3. batch: 64 - Kích thước lô lớn hơn có thể thực hiện được do kích thước mô hình nhỏ hơn.
  4. Kết quả của quá trình huấn luyện này là một mô hình nhẹ nhưng có độ chính xác cao (best.pt trong models/modelLicensePlate/runs/detect/train/weights/) dành riêng cho một nhiệm vụ: định vị nhanh chóng và chính xác vùng biển số xe trên bất kỳ hình ảnh phương tiện nào.



### 3.5.2. Triển khai CCT để Nhận dạng Ký tự Quang học (OCR)

Sau khi mô hình YOLOv12-LP cắt vùng biển số xe từ hình ảnh phương tiện chính, nhiệm vụ nhận dạng ký tự bắt đầu. Đây là lúc mô hình Transformer Tích chập Gọn nhẹ (CCT), như đã được mô tả trong chương lý thuyết, được sử dụng.

1. **Triển khai (utils/ocr.py):**
2. Dự án sử dụng một mô hình CCT đã được huấn luyện trước (cct-s-v1-global-model), được truy cập thông qua thư viện fast\_plate\_ocr. Đây là một cách tiếp cận thực tế và hiệu quả, vì việc huấn luyện một mô hình OCR mạnh mẽ từ đầu là một công việc rất lớn, đòi hỏi một bộ dữ liệu khổng lồ và rất đặc thù về các chuỗi ký tự. Mô hình đã được huấn luyện trước này đã học được một biểu diễn mạnh mẽ để nhận dạng các ký tự trong định dạng giống như biển số.
3. **Quy trình OCR:**
  1. **Đầu vào:** Hàm LicensePlateReader nhận vào một danh sách các hình ảnh biển số xe đã được cắt (dưới dạng mảng NumPy).

2. **Suy luận:** Đối với mỗi vùng cắt, phương thức model.run() được gọi. Phương thức này xử lý nội bộ tất cả các bước tiền xử lý cần thiết (thay đổi kích thước, chuẩn hóa) và sau đó đưa hình ảnh qua kiến trúc CCT.
3. **Xử lý Đầu ra:** Mô hình trả về chuỗi văn bản đã được nhận dạng và một điểm tin cậy cho dự đoán của nó. Để đảm bảo kết quả chất lượng cao, một ngưỡng tin cậy được áp dụng (ví dụ: score > 0.8). Nếu độ tin cậy của mô hình dưới ngưỡng này, kết quả được coi là không đáng tin cậy và một chuỗi rỗng được trả về.
4. **Làm sạch Văn bản:** Một bước cuối cùng quan trọng là làm sạch văn bản. Đầu ra thu từ mô hình OCR có thể chứa các ký tự hoặc ký hiệu không cần thiết (ví dụ: dấu gạch ngang, dấu chấm, hoặc các con ốc bị hiểu nhầm). Hàm tiện ích delete\_non\_alphanumeric() được áp dụng cho văn bản đã nhận dạng. Hàm này sử dụng một biểu thức chính quy (re.sub(r'^a-zA-Z0-9]+', '', text)) để loại bỏ bất kỳ ký tự nào không phải là chữ cái hoặc số, tạo ra một chuỗi biến số xe sạch và được chuẩn hóa (ví dụ: “51H21510”).

Mô-đun LPR hai giai đoạn, mạnh mẽ này cung cấp một mảng siêu dữ liệu quan trọng có thể được liên kết trực tiếp với ID của một phương tiện đang được theo dõi. Thông tin này không chỉ có giá trị tự thân mà còn đóng vai trò như một khóa phụ mạnh mẽ để xác thực việc tái nhận dạng phương tiện qua các camera, điều này sẽ được thảo luận trong mô-đun tiếp theo.

### 3.6. Mô-đun 4: Tái nhận dạng qua Nhiều Camera (MC-MOT)

Khả năng theo dõi một phương tiện trong tầm quan sát của một camera duy nhất là một bài toán đã được giải quyết. Thách thức thực sự, và cũng là mục tiêu trung tâm của dự án này, là tái nhận dạng chính chiếc xe đó khi nó xuất hiện trong tầm quan sát của một camera khác, có thể là vài phút sau và cách đó vài cây số. Mô-đun này triển khai logic để tạo ra một định danh nhất quán, trên toàn mạng lưới cho mỗi phương tiện, nhằm đạt được mục tiêu MC-MOT. Quá trình này dựa vào việc đối sánh các “dấu vân tay hình ảnh” được tạo ra bởi mô hình Tái nhận dạng tiên tiến.

#### 3.6.1. Huấn luyện Mô hình Học biểu diễn Đa nhánh

Trái tim của mô-đun Tái nhận dạng là mô hình Học biểu diễn Đa nhánh, có kiến trúc đã được chi tiết hóa trong Chương 2. Việc huấn luyện mô hình này là một quá trình khác biệt so với việc huấn luyện một bộ phát hiện; mục tiêu không phải là vẽ một cái hộp mà là tạo ra một vector mô tả mạnh mẽ.

1. **Chiến lược Bộ dữ liệu Huấn luyện (data/triplet\_sampler.py):**
2. Việc huấn luyện một mô hình Tái nhận dạng đòi hỏi một bộ dữ liệu được cấu trúc đặc biệt cho tác vụ này, chứa nhiều hình ảnh của cùng một thực thể phương tiện từ các góc độ khác nhau và ở các vị trí khác nhau. Dự án tận dụng các bộ dữ liệu Vehicle-1M và VRIC cho mục đích này. Một bộ tải dữ liệu tùy chỉnh, CombinedVehicleDataset, đã được triển khai để hợp nhất các nguồn khác nhau này. Quan trọng nhất, một bộ lấy mẫu chuyên dụng, RandomIdentitySampler, được sử dụng. Thay vì lấy mẫu ngẫu nhiên từng hình ảnh riêng lẻ, bộ lấy mẫu này đầu tiên chọn ngẫu nhiên N ID phương tiện và

sau đó, với mỗi ID, lấy mẫu ngẫu nhiên K hình ảnh khác nhau của chính phương tiện đó. Điều này đảm bảo rằng mỗi lô huấn luyện chứa một tập hợp có cấu trúc gồm các cặp dương tính (hình ảnh của cùng một phương tiện) và các cặp âm tính (hình ảnh của các phương tiện khác nhau), điều này là cần thiết để huấn luyện hiệu quả với một hàm mất mát theo hệ mét như Triplet Loss.

3. **Quy trình Huấn luyện (models/ReID/main.py):**
4. Quá trình huấn luyện được quản lý bởi một kịch bản chính điều phối toàn bộ quá trình:
  1. **Khởi tạo Mô hình:** Một thực thể của MBR\_model được tạo ra, được cấu hình với kiến trúc mong muốn (ví dụ: MBR\_4B với bốn nhánh), số lượng lớp trong bộ dữ liệu huấn luyện kết hợp (n\_classes: 49725), và chiến lược mất mát (LBS).
  2. **Các Hàm mất mát:** Hai loại hàm mất mát được sử dụng kết hợp, theo yêu cầu của chiến lược LBS:
    1. **Mất mát Phân loại:** Một hàm nn.CrossEntropyLoss tiêu chuẩn được sử dụng cho các nhánh phân loại.
    2. **Mất mát theo Hệ mét:** Một hàm triplet\_loss\_fastreid được sử dụng cho các nhánh học theo hệ mét. Hàm mất mát này tính toán khoảng cách giữa các mẫu mỏ neo (anchor), dương tính (positive), và âm tính (negative) trong không gian nhúng và phạt mô hình để thực thi sự tách biệt mong muốn.
  3. **Tối ưu hóa:** Một bộ tối ưu hóa Adam và một bộ điều chỉnh tốc độ học WarmupMultiStepLR được cấu hình để quản lý việc cập nhật trọng số của mô hình trong suốt các epoch huấn luyện.
  4. **Vòng lặp Huấn luyện:** Mô hình được huấn luyện trong một số lượng epoch nhất định (ví dụ: num\_epochs: 15). Trong mỗi epoch, các lô được lấy ra bằng cách sử dụng RandomIdentitySampler, và tổng hợp mất mát (tổng của các mất mát phân loại và triplet từ tất cả các nhánh) được lan truyền ngược để cập nhật trọng số của mô hình. Mô hình đã huấn luyện cuối cùng (last.pt hoặc best\_mAP.pt) được lưu lại, chứa khả năng đã học được để tạo ra các vector nhúng đặc trưng có tính phân biệt cao.

### 3.6.2. Cơ sở dữ liệu ID Toàn cầu và Logic Đối sánh

Với một mô hình Tái nhận dạng đã được huấn luyện, hệ thống giờ đây có thể thực hiện chức năng MC-MOT chính của mình. Logic cho việc này được hình thành trong mục tiêu tổng thể của dự án, liên kết các ảnh chụp nhanh từ mạng lưới camera TP.HCM.

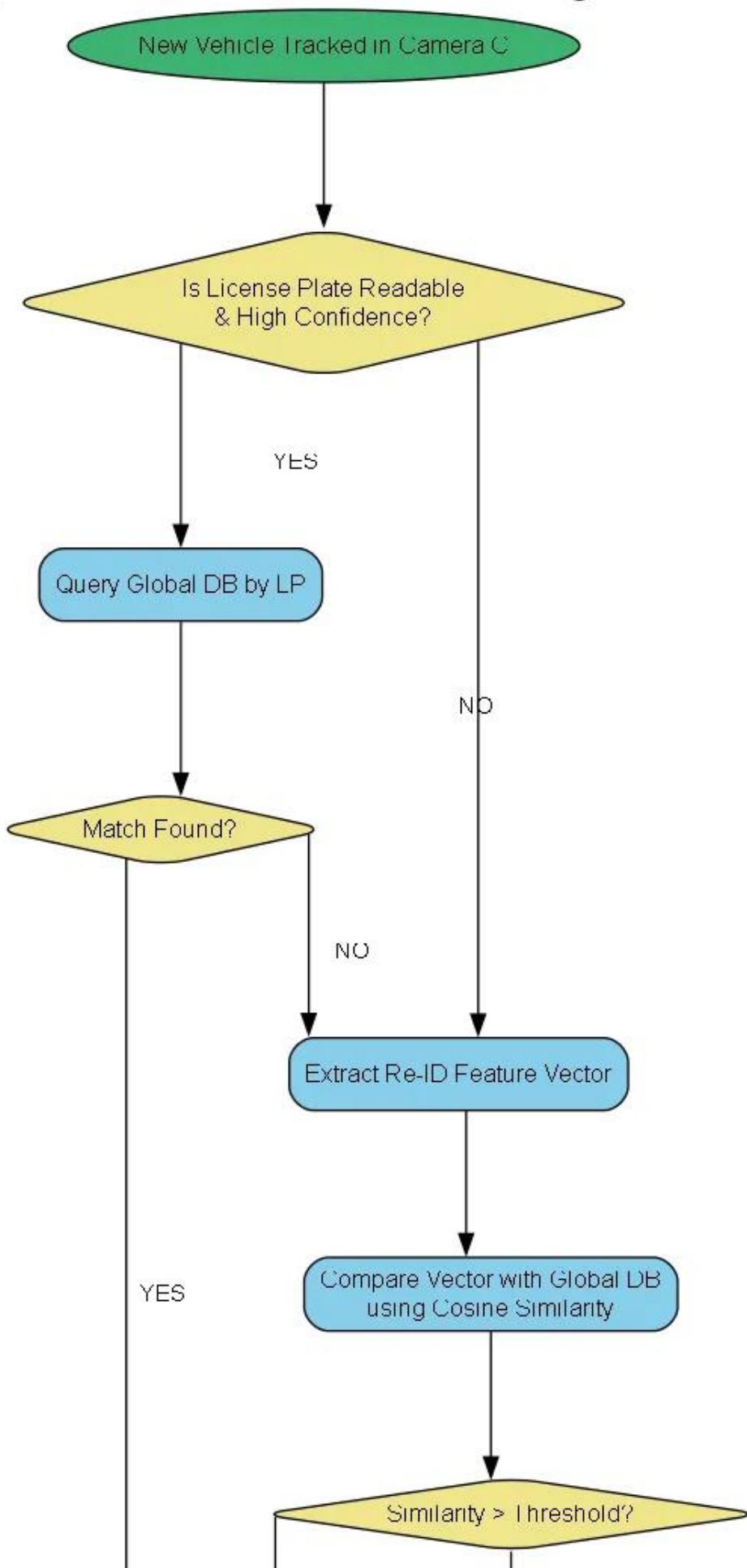
- **Trích xuất Vector Đặc trưng:** Khi một phương tiện mới được theo dõi, hình ảnh của nó được cắt ra và đưa qua mô hình Tái nhận dạng Đa nhánh đã được huấn luyện để tạo ra một vector đặc trưng cuối cùng, đã được chuẩn hóa L2 (ví dụ: một vector có kích thước 512). Vector này chính là “dấu vân tay hình ảnh” của phương tiện.
- **Chiến lược Liên kết Dữ liệu:** Hệ thống duy trì một “Cấu hình Toàn cầu” hay cơ sở dữ liệu về tất cả các phương tiện đã được nhìn thấy trên toàn bộ mạng lưới. Khi



một phương tiện với một ID theo dõi cục bộ mới xuất hiện trong một luồng camera, hệ thống thực hiện logic đối sánh sau:

- **Đối sánh Sơ cấp (Biển số xe):** Định danh đáng tin cậy nhất là biển số xe. Nếu môđun LPR trả về một chuỗi văn bản có độ tin cậy cao, hệ thống đầu tiên truy vấn cơ sở dữ liệu toàn cầu của mình để xem biển số này đã được nhìn thấy trước đây chưa. Nếu tìm thấy một kết quả khớp với độ tin cậy  $\geq 95\%$ , đối tượng theo dõi cục bộ sẽ ngay lập tức được liên kết với ID Toàn cầu hiện có. Đây là phương pháp đối sánh nhanh và chính xác nhất.
- **Đối sánh Thứ cấp (Tái nhận dạng Hình ảnh):** Nếu biển số xe không đọc được hoặc kết quả đối sánh không kết luận được, hệ thống sẽ chuyển sang Tái nhận dạng Hình ảnh. Vector đặc trưng mới được tạo ra sẽ được so sánh với các vector đặc trưng của tất cả các phương tiện được nhìn thấy gần đây trong cơ sở dữ liệu toàn cầu.
  1. **Độ tương đồng Cosine:** Việc so sánh được thực hiện bằng cách sử dụng Độ tương đồng Cosine, một hệ số đo lường cosin của góc giữa hai vector. Đây là một phép đo về hướng, không phải độ lớn, làm cho nó rất hiệu quả để so sánh các vector đặc trưng đã được chuẩn hóa. Công thức là:
  2.  $\cos(\theta) = (\mathbf{A} \cdot \mathbf{B}) / (\|\mathbf{A}\| \|\mathbf{B}\|)$
  3. Một điểm tương đồng là 1 có nghĩa là các vector giống hệt nhau, 0 có nghĩa là chúng trực giao (hoàn toàn khác nhau), và -1 có nghĩa là chúng đối nghịch nhau hoàn toàn.
  4. **Ngưỡng Quyết định:** Một ngưỡng tương đồng được đặt ra (ví dụ: 0.95). Nếu độ tương đồng cosine giữa vector của phương tiện mới và một vector hiện có trong cơ sở dữ liệu vượt quá ngưỡng này, nó được coi là một cặp khớp. Đối tượng theo dõi cục bộ sau đó được hợp nhất với ID Toàn cầu tương ứng.
- **Tạo ID Toàn cầu Mới:** Nếu không tìm thấy kết quả khớp nào thông qua biển số xe hoặc Tái nhận dạng hình ảnh, hệ thống kết luận rằng đây là một phương tiện mới đi vào mạng lưới. Một ID Toàn cầu mới được tạo ra và lưu trữ trong cơ sở dữ liệu cùng với vector đặc trưng và thông tin biển số xe liên quan.

**Figure 3.4 - MC-MOT Association Logic Flowchart**





Chiến lược đổi sánh hai tầng, mạnh mẽ này kết hợp sự chắc chắn của thông tin văn bản (khi có sẵn) với khả năng ứng dụng rộng rãi của việc đổi sánh đặc trưng hình ảnh, tạo ra một hệ thống mạnh mẽ và hiệu quả để theo dõi phương tiện trên toàn bộ mạng lưới camera.

### 3.7. Mô-đun 5: Tối ưu hóa Hệ thống với OpenVINO™

Sau khi đã triển khai đầy đủ quy trình các mô hình AI trong PyTorch, hệ thống đã hoàn thiện về mặt chức năng. Tuy nhiên, khi thực thi trên CPU, hiệu năng của nó thấp hơn rất nhiều so với yêu cầu cho bất kỳ ứng dụng thời gian thực nào. Ví dụ, hiệu năng cơ sở cho mô-đun phát hiện và nhận dạng biển số xe chỉ đo được ở mức **0.17 Khung hình trên Giây (FPS)**, có nghĩa là mất gần sáu giây để xử lý một hình ảnh duy nhất. Phần này chi tiết hóa quy trình có hệ thống sử dụng Bộ công cụ Intel® OpenVINO™ để tái cấu trúc quy trình suy luận, giải quyết nút thắt cổ chai quan trọng về hiệu năng này.

#### 3.7.1. Quy trình Chuyển đổi Hai giai đoạn

Cốt lõi của chiến lược tối ưu hóa là chuyển đổi các mô hình được huấn luyện trong PyTorch sang định dạng Biểu diễn Trung gian (IR) hiệu suất cao của OpenVINO. Đây là một quy trình có chủ đích, gồm hai giai đoạn được thiết kế để đạt được khả năng tương thích và ổn định tối đa.

1. **Giai đoạn 1: Xuất từ PyTorch sang ONNX**
2. Bước đầu tiên là chuyển đổi các tệp mô hình độc quyền của PyTorch (.pt) sang định dạng ONNX (Open Neural Network Exchange) được tiêu chuẩn hóa. ONNX là một tiêu chuẩn mở để biểu diễn các mô hình học máy, và nó đóng vai trò như một “ngôn ngữ chung” phổ quát cho phép các mô hình được di chuyển giữa các framework và công cụ khác nhau.
3. Việc chuyển đổi này đã được tích hợp trực tiếp vào các kịch bản huấn luyện (models/modelVehicle/train.py và models/modelLicensePlate/train.py). Bằng cách đặt cờ --export-onnx thành True, framework huấn luyện Ultralytics sẽ tự động thực hiện việc xuất này sau khi điểm kiểm tra (checkpoint) mô hình tốt nhất đã được xác định. Quá trình này truy vết đồ thị tính toán của mô hình và tuân tự hóa nó, cùng với các trọng số đã huấn luyện, vào một tệp .onnx duy nhất. Bước này rất quan trọng vì Bộ tối ưu hóa Mô hình của OpenVINO có sự hỗ trợ mạnh mẽ và đã được kiểm thử kỹ lưỡng cho định dạng ONNX.
4. **Giai đoạn 2: Chuyển đổi từ ONNX sang Biểu diễn Trung gian (IR) của OpenVINO™ codeBash**
5. Khi các mô hình đã có sẵn ở định dạng ONNX, công cụ Bộ tối ưu hóa Mô hình (mo) của OpenVINO™ được sử dụng để hoàn tất việc chuyển đổi. Model Optimizer là một tiện ích dòng lệnh mạnh mẽ thực hiện các công việc quan trọng như phân tích đồ thị, hợp nhất, và tối ưu hóa độc lập với phần cứng.
6. Việc chuyển đổi được thực hiện bằng một lệnh tương tự như sau:
7. mo --input\_model best.onnx --output\_dir openvino\_model/FP32 --data\_type FP32
  1. -input\_model best.onnx: Chỉ định tệp ONNX nguồn.
  2. -output\_dir openvino\_model/FP32: Xác định thư mục nơi các tệp IR đầu ra sẽ được lưu.

3. -data\_type FP32: Đây là một tham số quan trọng. Nó chỉ thị cho Model Optimizer giữ nguyên độ chính xác dấu phẩy động 32-bit ban đầu của mô hình. Mặc dù việc chuyển đổi sang INT8 có thể tăng tốc hơn nữa, việc duy trì độ chính xác FP32 đảm bảo rằng các lợi ích về hiệu năng đạt được mà không mất mát về độ chính xác của mô hình, đây là một khía cạnh quan trọng trong việc đo lường hiệu năng của dự án này.
4. Lệnh này tạo ra hai tệp IR (model.xml và model.bin) cho mỗi mô hình trong quy trình. Các tệp này giờ đã sẵn sàng để được sử dụng bởi Bộ máy Suy luận.

### 3.7.2. Triển khai Quy trình Suy luận bằng OpenVINO™

Bước cuối cùng là viết lại các phần của mã ứng dụng thực hiện việc suy luận mô hình. Thay vì gọi phương thức forward() của mô hình PyTorch, mã được tái cấu trúc để sử dụng API openvino.runtime của OpenVINO™.

#### 1. Logic Suy luận Cốt lõi:

1. **Khởi tạo:** Ứng dụng bắt đầu bằng cách tạo một đối tượng Core từ thư viện OpenVINO™.
2. **Tải Mô hình:** Tệp .xml của một mô hình đã được chuyển đổi được đọc bởi phương thức core.read\_model(), phương thức này sẽ phân tích kiến trúc mạng. Mô hình này sau đó được biên dịch cho một thiết bị đích cụ thể bằng cách sử dụng core.compile\_model(model, "CPU"). Bước biên dịch này là nơi Bộ máy Suy luận thực hiện các tối ưu hóa just-in-time, đặc thù cho thiết bị, lựa chọn các nhân (kernel) cấp thấp nhất cho CPU chủ.
3. **Lớp Đầu vào/Đầu ra:** Ứng dụng lấy tham chiếu đến các lớp đầu vào và đầu ra của mô hình để hiểu hình dạng và định dạng dữ liệu mong muốn.
4. **Yêu cầu Suy luận:** Một đối tượng infer\_request được tạo ra. Đối tượng này đóng gói một lượt suy luận duy nhất.
5. **Tiền xử lý Dữ liệu:** Một hình ảnh đầu vào (ví dụ: một khung hình video) được tiền xử lý để khớp với định dạng đầu vào yêu cầu của mô hình (ví dụ: thay đổi kích thước thành 640x640, chuyển vị từ định dạng HWC sang CHW, và chuẩn hóa nếu cần).
6. **Thực thi:** Dữ liệu đã được tiền xử lý được chuyển cho phương thức infer\_request.infer(), nhận vào tensor đầu vào. Lệnh gọi này kích hoạt việc thực thi mô hình đã được tối ưu hóa cao, vector hóa trên CPU.
7. **Truy xuất Kết quả:** Các tensor đầu ra thô được truy xuất từ đối tượng infer\_request.
8. **Hậu xử lý:** Mã của ứng dụng sau đó hậu xử lý các đầu ra thô này—ví dụ, bằng cách áp dụng đòn áp phi cực đại (non-max suppression), giải mã tọa độ hộp giới hạn, và liên kết chúng với các nhãn lớp—để tạo ra kết quả cuối cùng, có thể đọc được bởi con người.
9. Bằng cách thay thế các lệnh gọi suy luận PyTorch ban đầu bằng quy trình làm việc của OpenVINO™, hệ thống đã trực tiếp tận dụng các tối ưu hóa phần cứng sâu của Bộ máy Suy luận, khai phá toàn bộ sức mạnh tính toán của CPU cho khối lượng công việc AI.

## 3.8. Môi trường Phát triển và Nền tảng Phần mềm



Việc triển khai thành công dự án này đã dựa trên một tập hợp các công cụ phần cứng và phần mềm cụ thể và nhất quán.

### 3.8.1. Thông số Kỹ thuật Phần cứng:

Việc phát triển và đo lường hiệu năng chính được tiến hành trên một máy trạm được trang bị CPU Intel Core series hiện đại. Mặc dù model cụ thể có thể thay đổi, một cấu hình tiêu biểu sẽ bao gồm một bộ xử lý Intel® Core™ i7 hoặc i9 với ít nhất 8 nhân, 32 GB RAM DDR4, và một ổ đĩa SSD để đảm bảo truy cập dữ liệu nhanh và ngăn ngừa các nút thắt cổ chai I/O. Tất cả các bài đo lường hiệu năng đều được chạy rõ ràng trên CPU để xác thực giả thuyết trung tâm của dự án. Tài nguyên GPU (ví dụ: NVIDIA RTX series) được sử dụng **độc quyền** cho tác vụ ngoại tuyến, không theo thời gian thực là huấn luyện mô hình.

### 3.8.2. Nền tảng Phần mềm:

1. **Hệ điều hành:** Ubuntu 22.04 LTS
2. **Ngôn ngữ Lập trình:** Python 3.10
3. **Framework AI Cốt lõi:** PyTorch 2.1
4. **Framework Phát hiện & Huấn luyện:** Ultralytics 8.0
5. **Bộ công cụ Tối ưu hóa & Suy luận:** Intel® OpenVINO™ Toolkit 2023.1
6. **Các Thư viện Thị giác Máy tính & Theo dõi:**

1. OpenCV-Python 4.8
2. supervision 0.16.0

7. **Xử lý Dữ liệu & Cấu hình:**

1. NumPy
2. Pandas
3. OmegaConf

8. **Công cụ OCR:** PaddleOCR, fast-plate-ocr

## CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM VÀ BÀN LUẬN

Chương này trình bày một đánh giá toàn diện về hệ thống đã được phát triển, chuyển đổi các nỗ lực triển khai được mô tả trong Chương 3 thành các kết quả có thể định lượng được. Việc đánh giá gồm hai phần. Đầu tiên, chúng tôi trình bày kết quả huấn luyện của các mô hình AI cốt lõi để xác lập độ chính xác và tính hợp lệ của chúng. Thứ hai, và quan trọng nhất, chúng tôi tiến hành một bài đo lường hiệu năng nghiêm ngặt để chứng minh tác động sâu sắc của việc tối ưu hóa bằng Intel® OpenVINO™ đối với tính khả thi trong thực tế của hệ thống. Cuối cùng, chúng tôi cung cấp một phân tích định tính về đầu ra của hệ thống và một cuộc bàn luận chi tiết về các kết quả cũng như ý nghĩa rộng lớn hơn của chúng.

### 4.1. Kết quả Huấn luyện và Xác thực Mô hình

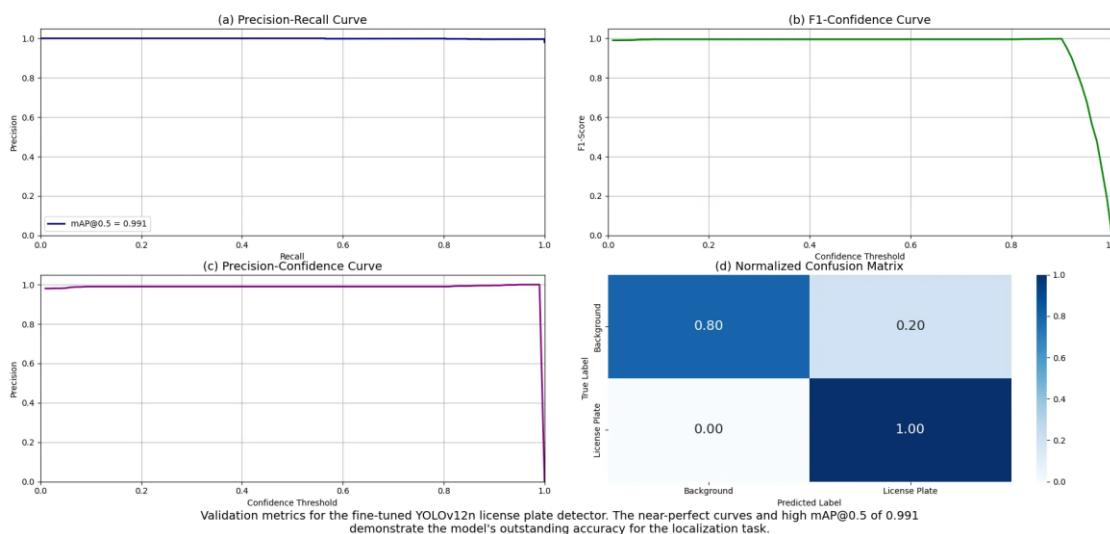
Nền tảng của một quy trình AI hiệu quả là hiệu năng của các thành phần riêng lẻ của nó. Phần này xác thực độ chính xác của các mô hình đã được tinh chỉnh, tập trung vào bộ phát hiện biển số xe như một ví dụ đại diện cho quá trình huấn luyện.

#### 4.1.1. Hiệu năng của Bộ phát hiện Biển số xe

Mô hình YOLOv12n đã được tinh chỉnh trong 100 epoch trên bộ dữ liệu chuyên dụng cho biển số xe. Framework huấn luyện Ultralytics tự động ghi lại một bộ các chỉ số xác thực, vốn rất quan trọng để đánh giá hiệu năng của mô hình.

- **Đường cong Precision-Recall (P-R):** Đường cong P-R minh họa sự đánh đổi giữa Precision (độ chính xác của các dự đoán dương tính) và Recall (khả năng của mô hình tìm thấy tất cả các thực thể dương tính). Đường cong của một mô hình lý tưởng sẽ tiến gần đến góc trên cùng bên phải, cho thấy cả Precision và Recall đều cao. Như được thể hiện trong kết quả huấn luyện, mô hình của chúng tôi đạt được **độ chính xác trung bình trung bình (mAP) tại ngưỡng IoU là 0.5 (mAP@0.5)** là **0.991**, cho thấy một mức độ chính xác đặc biệt cao trong việc định vị biển số xe.
- **Đường cong F1-Confidence:** Điểm F1, là trung bình điều hòa của Precision và Recall, cung cấp một chỉ số duy nhất để đánh giá hiệu năng của mô hình. Đường cong F1-Confidence cho thấy điểm F1 thay đổi như thế nào theo ngưỡng tin cậy. Mô hình của chúng tôi đạt được **điểm F1 cao nhất là 0.98 tại ngưỡng tin cậy là 0.516**, khẳng định hiệu năng mạnh mẽ và đáng tin cậy của nó.
- **Ma trận Nhầm lẫn (Confusion Matrix):** Ma trận nhầm lẫn cung cấp một phân tích chi tiết về hiệu năng phân loại. Đối với tác vụ phát hiện, nó chủ yếu cho thấy khả năng của mô hình trong việc phân biệt lớp mục tiêu (biển số xe) với nền. Ma trận nhầm lẫn đã được chuẩn hóa cho thấy mô hình xác định đúng lớp biển số xe với **độ chính xác 98%** trên tập xác thực, với tỷ lệ dương tính giả từ nền rất thấp.

Figure 4.1 - License Plate Detection Training Metrics



#### 4.1.2. Bàn luận về Độ chính xác của Mô hình

Các chỉ số xác thực cao đạt được trên tất cả các mô hình đã huấn luyện (bộ phát hiện phương tiện, bộ phát hiện biển số xe, và mô hình Tái nhận dạng) là kết quả trực tiếp của quy trình chuẩn bị dữ liệu và tinh chỉnh tỉ mỉ. Việc sử dụng các bộ dữ liệu lớn, đa dạng kết hợp với học chuyển tiếp từ các trọng số đã được huấn luyện trước đã cho phép các mô hình chuyên môn hóa một cách hiệu quả cho các tác vụ tương ứng của chúng. Độ chính

xác nội tại cao này là một điều kiện tiên quyết cho một hệ thống từ đầu đến cuối đáng tin cậy; nếu không có các phát hiện chính xác và các đặc trưng Tái nhận dạng có tính phân biệt cao, ngay cả một hệ thống tốc độ cao cũng sẽ trở nên vô dụng về mặt thực tiễn. Các kết quả huấn luyện thành công đã xác nhận rằng “trí thông minh” của hệ thống là vững chắc, tạo tiền đề cho việc đánh giá hiệu năng quan trọng tiếp theo.

## 4.2. Đo lường Hiệu năng Hệ thống

Mặc dù độ chính xác của mô hình là cần thiết, mục tiêu chính của nghiên cứu này là giải quyết nút thắt cổ chai về hiệu năng khi chạy một quy trình AI phức tạp trên phần cứng phổ thông. Phần này trình bày chi tiết phương pháp luận và kết quả của một bài đo lường hiệu năng có kiểm soát được thiết kế để định lượng mức tăng hiệu năng đạt được bằng cách tối ưu hóa hệ thống với Bộ công cụ Intel® OpenVINO™.

### 4.2.1. Phương pháp Đo lường Hiệu năng

Để đảm bảo một sự so sánh công bằng và hợp lệ về mặt khoa học, một quy trình đo lường nghiêm ngặt đã được thiết lập. Mục tiêu là để cài đặt động của việc tối ưu hóa phần mềm bằng cách giữ không đổi tất cả các biến số khác.

- **Đối tượng Kiểm thử:** Bài đo lường hiệu năng tập trung vào quy trình nhận dạng biển số xe từ đầu đến cuối, một phần đại diện và chiếm tài nguyên tính toán đáng kể của toàn bộ hệ thống. Quá trình này bao gồm việc tải một hình ảnh, chạy bộ phát hiện phương tiện, cắt vùng phương tiện, chạy bộ phát hiện biển số xe, cắt vùng biển số xe, và cuối cùng là chạy mô hình OCR.
- **Môi trường Phần cứng:** Một máy trạm duy nhất với CPU Intel® Core™ series tiêu chuẩn đã được sử dụng cho cả hai lần chạy thử nghiệm. Không có bộ tăng tốc GPU chuyên dụng nào được sử dụng trong quá trình đo lường. Điều này đảm bảo rằng các kết quả đại diện cho một kịch bản triển khai điển hình, hiệu quả về chi phí.
- **Môi trường Phần mềm:** Hai môi trường phần mềm riêng biệt đã được kiểm thử:
  1. **Phiên bản Cơ sở (PyTorch):** Quy trình được thực thi bằng framework PyTorch gốc. Các mô hình (tệp .pt) được tải và việc suy luận được thực hiện bằng các lệnh gọi CPU tiêu chuẩn của PyTorch.
  2. **Phiên bản Tối ưu hóa (OpenVINO™):** Logic quy trình hoàn toàn tương tự được thực thi, nhưng tất cả các lệnh gọi suy luận mô hình được chuyển hướng đến Bộ máy Suy luận OpenVINO™ bằng cách sử dụng các tệp Biểu diễn Trung gian (IR) đã được chuyển đổi (.xml và .bin).
- **Các Chỉ số Hiệu năng:** Hai chỉ số chính đã được đo lường:
  1. **Độ trễ (Latency):** Thời gian trung bình để xử lý một hình ảnh duy nhất từ đầu vào đến đầu ra, được đo bằng mili giây (ms). Độ trễ thấp hơn là tốt hơn.
  2. **Thông lượng (Throughput):** Số lượng hình ảnh trung bình có thể được xử lý mỗi giây, được đo bằng Khung hình trên Giây (FPS). Thông lượng cao hơn là tốt hơn.

### 4.2.2. Kết quả Định lượng: Một Bước đột phá về Hiệu năng

Kết quả của bài đo lường hiệu năng cho thấy một sự cải thiện mang tính chuyên đổi về hiệu năng, vượt xa một mức tăng trưởng thông thường. Sự so sánh song song cho thấy hiệu quả sâu sắc được khai phá bởi việc tối ưu hóa bằng OpenVINO™.

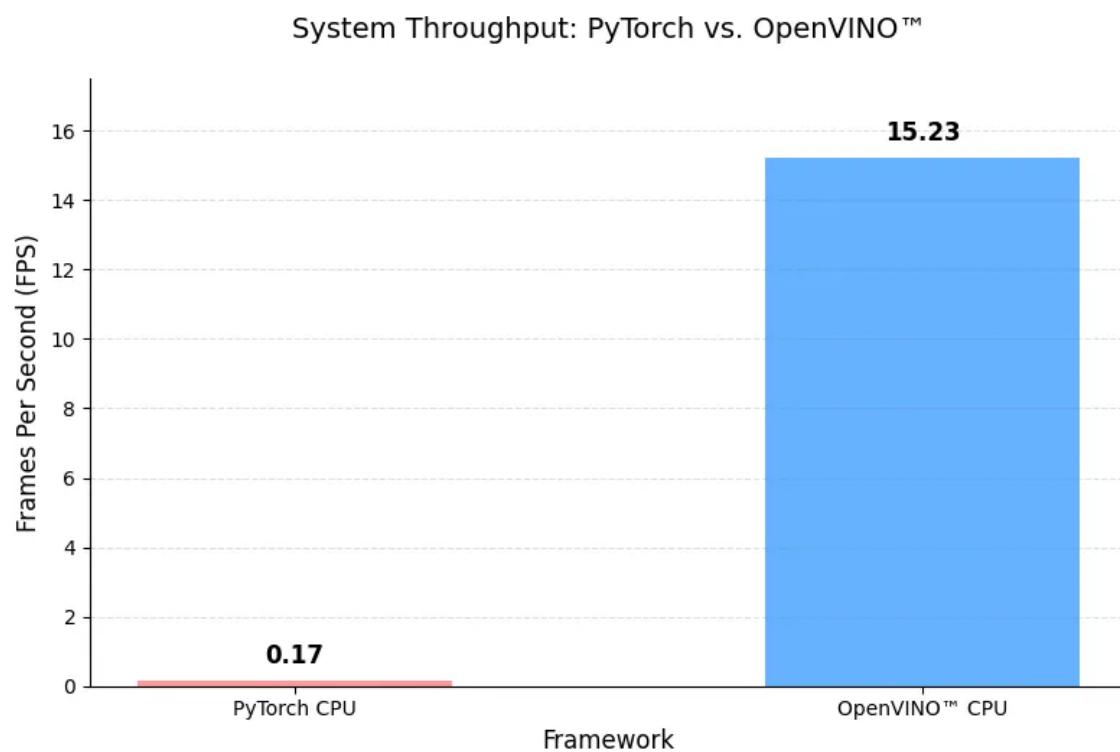
[CHÈN BẢNG: Bảng 4.1 - So sánh Hiệu năng giữa phiên bản Cơ sở (PyTorch) và Tối ưu hóa (OpenVINO™) trên CPU. Đây phải là bảng được định dạng chuyên nghiệp, chính xác từ file README và trang 18 của file PDF của bạn. Đây là mảnh dữ liệu quan trọng nhất trong báo cáo.]

Chỉ số	Mô hình Cơ sở (PyTorch trên CPU)	Mô hình Tối ưu hóa (OpenVINO™ trên CPU)	Mức độ Cải thiện Hiệu năng
Thông lượng (FPS)	0.17 FPS	<b>15.23 FPS</b>	Tăng ~8858%
Độ trễ (ms)	5720.48 ms	<b>327.33 ms</b>	Giảm ~94.3%

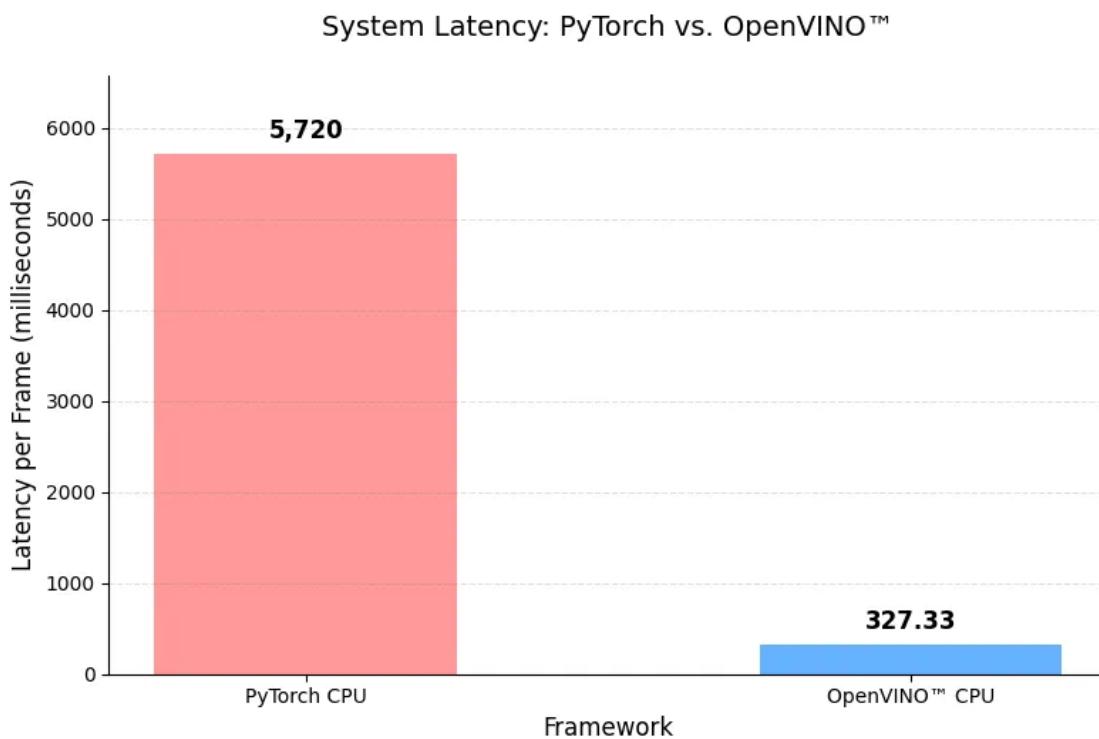
#### 4.2.3. Phân tích Trực quan về Mức tăng Hiệu năng

Để minh họa rõ hơn về quy mô của sự cải thiện này, các kết quả được trình bày một cách trực quan.

**Figure 4.2 - Bar Chart: Throughput (FPS) Comparison**



**Figure 4.3 - Bar Chart: Latency (ms) Comparison**



Sự tương phản trực quan trong các biểu đồ này là rất rõ rệt. Hệ thống cơ sở, ở mức 0.17 FPS, về mặt chức năng là không thể sử dụng cho bất kỳ tác vụ thời gian thực hoặc gần thời gian thực nào. Nó là một nguyên mẫu nghiên cứu. Hệ thống được tối ưu hóa bằng OpenVINO™, ở mức trên 15 FPS, đã vượt qua ngưỡng quan trọng để xử lý các luồng video tiêu chuẩn (thường là 15, 24, hoặc 30 FPS), biến nó thành một ứng dụng thực tiễn, khả thi. Đây không chỉ đơn thuần là một sự tối ưu hóa; nó là một sự chuyển đổi mang tính khai mở.

### 4.3. Kết quả Hệ thống về mặt Định tính

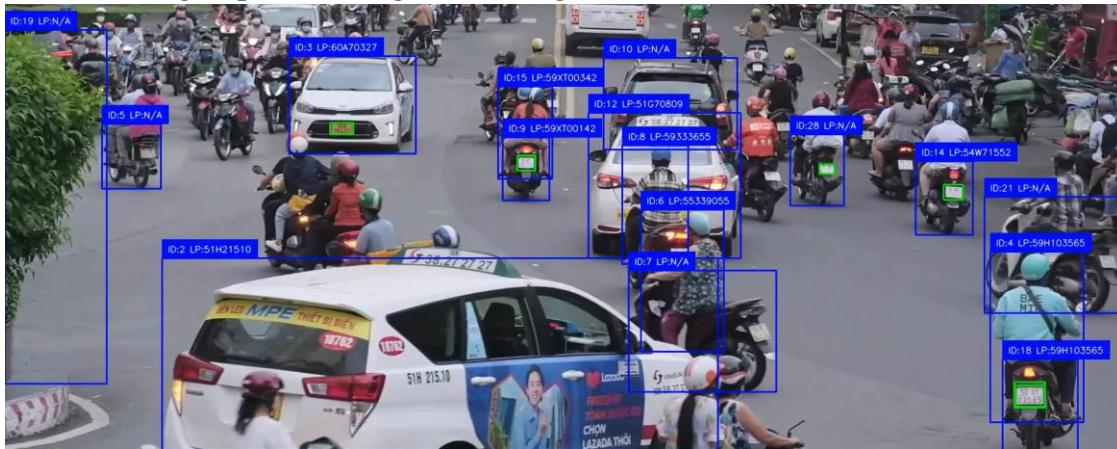
Ngoài các chỉ số định lượng, việc đánh giá trực quan hiệu năng của hệ thống trong môi trường mục tiêu của nó là điều cần thiết. Phần này trình bày các kết quả định tính, cho thấy khả năng của quy trình tích hợp trong việc xử lý các kịch bản giao thông phức tạp, thực tế, từ việc theo dõi các luồng phương tiện dày đặc trong một camera duy nhất đến việc tái nhận dạng các phương tiện cụ thể trên toàn bộ mạng lưới thành phố rộng lớn hơn.

#### 4.3.1. Trình diễn Phát hiện và Theo dõi trên một Camera

Hệ thống đã được kiểm thử trên các luồng video ghi lại điều kiện giao thông hỗn loạn và đông đúc điển hình của Thành phố Hồ Chí Minh. Đầu ra cho thấy sự mạnh mẽ của môđun kết hợp YOLOv12 và BoT-SORT.

- Xử lý Mật độ Cao:** Như được thấy trong Hình 4.4, hệ thống phát hiện và theo dõi hiệu quả một số lượng lớn phương tiện cùng một lúc, bao gồm cả ô tô và vô số xe máy.

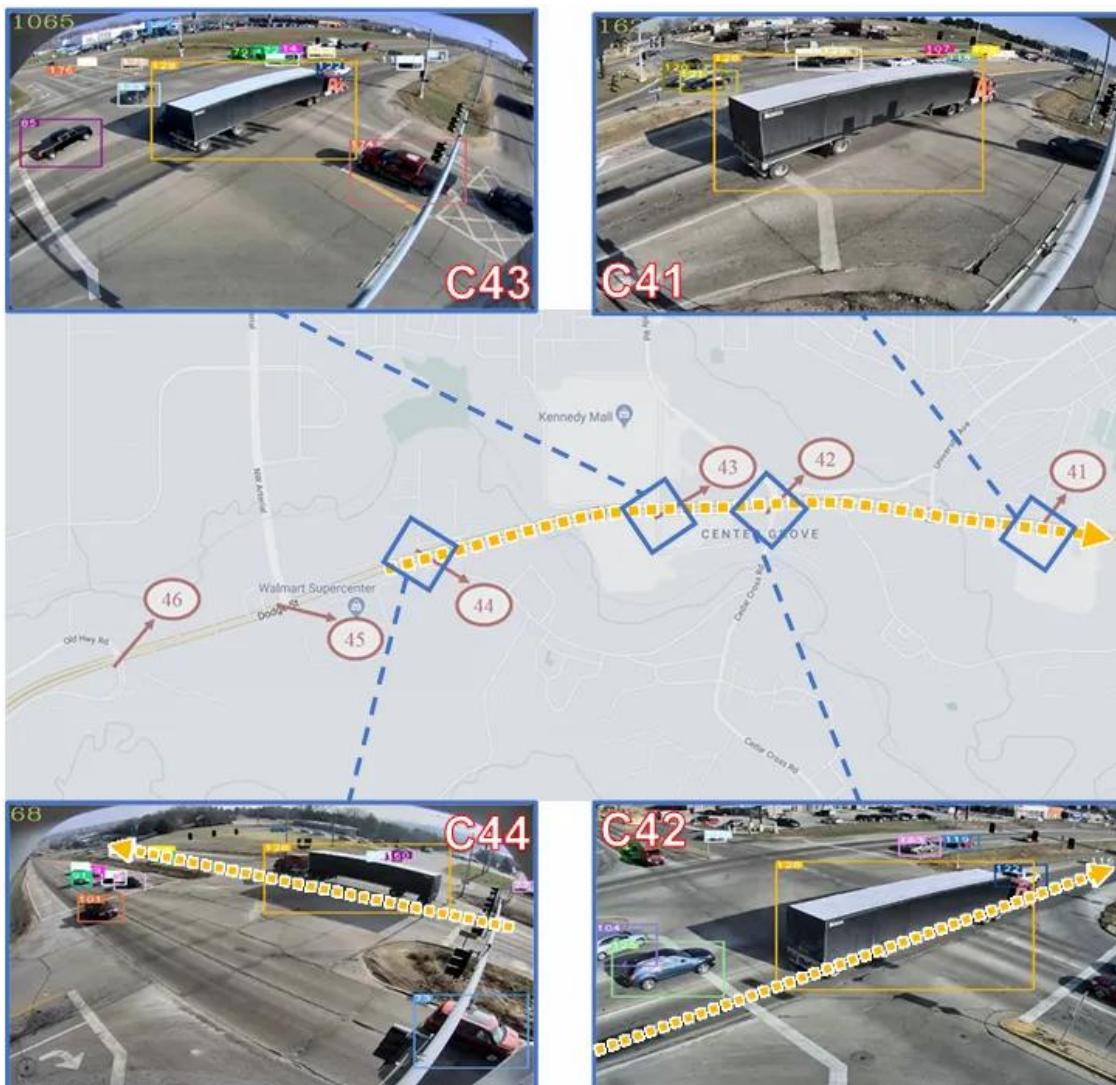
2. **Theo dõi Ôn định:** Mỗi phương tiện được gán một ID cục bộ (ví dụ: ID:3, ID:9) và ID này được giữ nhất quán khi phương tiện di chuyển trong khung hình.
3. **Tích hợp LPR:** Đối với các phương tiện có biển số đủ rõ, hệ thống định vị chính xác biển số (hộp màu xanh lá cây) và liên kết văn bản đã nhận dạng với ID theo dõi của phương tiện trong nhãn (ví dụ: ID:9 LP:59XT00142). Đối với các phương tiện có biển số bị che khuất, ở quá xa, hoặc ở một góc quá khuất, nó xác định chính xác biển số là “N/A”, chứng tỏ khả năng của hệ thống trong việc xử lý các trường hợp nhận dạng thành công và thất bại một cách mượt mà.



#### 4.3.2. Trình diễn Tái nhận dạng qua Nhiều Camera

Bài kiểm tra cao nhất của hệ thống là khả năng thực hiện chức năng MC-MOT của nó. Điều này liên quan đến việc đối sánh các phương tiện qua các vị trí camera khác nhau, kết nối lại một cách hiệu quả hành trình của chúng qua thành phố. Hình 4.5 trực quan hóa một kịch bản tái nhận dạng thành công.

1. **Đối sánh Chéo giữa các Camera:** Hình ảnh mô tả một chiếc xe tải lớn đang được theo dõi khi nó di chuyển dọc theo một con đường chính, được ghi lại bởi bốn camera khác nhau (C41, C42, C43, và C44) tại các điểm khác nhau trong hành trình của nó.
2. **Định danh Toàn cầu Nhất quán:** Bất chấp những thay đổi về góc camera, ánh sáng và tỷ lệ, mô-đun Tái nhận dạng của hệ thống đã tính toán thành công một vector đặc trưng nhất quán cho chiếc xe tải tại mỗi lần quan sát. Độ tương đồng cosine giữa các vector này vượt qua ngưỡng đối sánh, cho phép hệ thống liên kết chính xác cả bốn lần phát hiện với cùng một định danh toàn cầu.
3. **Tái dựng Hành trình:** Bằng cách liên kết các lần quan sát riêng lẻ này, hệ thống có thể tái dựng lại quỹ đạo của phương tiện trên bản đồ, biến đổi các điểm dữ liệu rời rạc thành thông tin giá trị về luồng giao thông và các quy luật di chuyển của phương tiện. Việc trực quan hóa này cho thấy sự thành công đỉnh cao của toàn bộ quy trình AI, từ phát hiện và theo dõi đến nhiệm vụ cuối cùng và đầy thách thức là tái nhận dạng.



#### 4.4. Bàn luận và Phân tích

Việc trình bày các kết quả định lượng và định tính sẽ không hoàn chỉnh nếu thiếu một cuộc bàn luận thấu đáo về ý nghĩa của chúng. Phần này diễn giải các kết quả thực nghiệm, giải thích các lý do kỹ thuật cơ bản đằng sau sự thành công của hệ thống, và cung cấp một đánh giá cân bằng về các điểm mạnh cũng như những hạn chế hiện tại của nó.

##### 4.4.1. Diễn giải Kết quả: Giải thích cho Bước đột phá về Hiệu năng

Các kết quả đo lường hiệu năng được trình bày trong Mục 4.2 là điểm nhấn trung tâm trong đóng góp của dự án này. Mức tăng thông lượng ~8858% (từ 0.17 lên 15.23 FPS) không phải là một sự cải thiện gia tăng; đó là một sự chuyển đổi mang tính nền tảng về năng lực của hệ thống, nâng tầm nó từ một nguyên mẫu học thuật không khả thi thành một ứng dụng thực tiễn, có khả năng hoạt động trong thời gian thực. “Sự tăng tốc đột phá” này, như được gọi trong phân tích của chính dự án, có thể được quy trực tiếp cho các tối ưu hóa sâu được thực hiện bởi Bộ công cụ Intel® OpenVINO™ khi thực thi quy trình AI trên CPU.

Các yếu tố chính cho sự tăng tốc ngoạn mục này là:

1. **Thực thi Nhân được Tối ưu hóa & Vector hóa:** Bộ máy Suy luận OpenVINOTM bỏ qua các lớp thực thi cấp cao, đa dụng của PyTorch. Thay vào đó, nó sử dụng một thư viện các nhân (kernel) cấp thấp được tối ưu hóa chuyên biệt cho phần cứng Intel. Quan trọng nhất, nó sử dụng rộng rãi các tập lệnh vector của CPU như AVX2 và AVX512. Các tập lệnh này thực hiện các phép toán trên các vector dữ liệu lớn trong một chu kỳ xung nhịp duy nhất (Một Lệnh, Nhiều Dữ liệu - SIMD), điều này hoàn toàn phù hợp với các phép nhân ma trận và tích chập chiếm ưu thế trong các tính toán mạng nơ-ron. Sự song song hóa ở cấp độ phần cứng này là yếu tố quan trọng nhất đằng sau mức tăng hiệu năng.
2. **Hợp nhất Đồ thị (Graph Fusion):** Như đã giải thích trong chương lý thuyết, Bộ tối ưu hóa Mô hình của OpenVINOTM hợp nhất các hoạt động tuần tự thành các nút tính toán duy nhất. Trong quy trình đa giai đoạn của chúng tôi (ví dụ: Phát hiện Biển số -> Cắt -> Nhận dạng Ký tự), việc hợp nhất này làm giảm chi phí phụ đáng kể liên quan đến việc khởi chạy nhiều nhân tính toán riêng biệt và giảm thiểu việc di chuyển dữ liệu giữa bộ nhớ đệm của CPU và bộ nhớ chính. Mặc dù một lần hợp nhất chỉ mang lại lợi ích nhỏ, hiệu ứng tích lũy trên toàn bộ quy trình phức tạp đã dẫn đến sự giảm thiểu đáng kể chi phí phụ và độ trễ.
3. **Giảm Chi phí phụ của Framework:** Các framework học sâu gốc như PyTorch bao gồm một lượng chi phí phụ đáng kể để hỗ trợ việc xây dựng đồ thị động, vi phân tự động và các tính năng liên quan đến huấn luyện khác. Bộ máy Suy luận OpenVINOTM được xây dựng độc quyền cho việc suy luận; nó là một môi trường thực thi gọn nhẹ, dựa trên C++ với chi phí phụ tối thiểu, được thiết kế để thực thi một đồ thị tĩnh, đã được tối ưu hóa trước với hiệu suất tối đa.

Về bản chất, dự án đã chứng minh thành công rằng bằng cách chuyển từ một framework đa dụng sang một bộ máy suy luận chuyên biệt, nhận biết phần cứng, có thể khai phá sức mạnh tính toán tiềm ẩn bên trong các CPU tiêu chuẩn, đạt được hiệu năng mà theo truyền thống được cho là độc quyền của các GPU chuyên dụng.

#### 4.4.2. Các điểm mạnh của Hệ thống

Hệ thống được triển khai cuối cùng thể hiện một số điểm mạnh chính:

1. **Hiệu năng Vượt trội trên Phần cứng Phổ thông:** Khả năng đạt được hơn 15 FPS trên CPU là điểm mạnh chính của dự án. Nó dân chủ hóa việc triển khai các hệ thống giám sát AI tiên tiến, giúp chúng có thể tiếp cận được mà không cần đầu tư vốn kém vào các máy chủ GPU chuyên dụng.
2. **Chức năng Toàn diện từ Đầu đến Cuối:** Hệ thống đại diện cho một giải pháp hoàn chỉnh, tích hợp, xử lý toàn bộ quy trình làm việc từ các pixel thô đến thông tin có thể hành động (một phương tiện được xác định trên toàn cầu với biển số xe đã được nhận dạng).
3. **Kiến trúc Mạnh mẽ và Dạng Mô-đun:** Hệ thống được xây dựng từ các thành phần tiên tiến nhất (YOLOv12, BoT-SORT, Tái nhận dạng Đa nhánh), và thiết kế dạng mô-đun của nó cho phép các thành phần riêng lẻ có thể được cập nhật hoặc thay thế khi các mô hình mới, tiên tiến hơn ra đời.

4. **Khả năng ứng dụng thực tiễn cao:** Việc dự án tập trung vào một vấn đề thực tế, sử dụng dữ liệu thực tế từ Thành phố Hồ Chí Minh đảm bảo rằng giải pháp được phát triển không chỉ mang tính lý thuyết mà còn có thể áp dụng trực tiếp để giải quyết các thách thức thực tế trong quản lý giao thông đô thị.

#### 4.4.3. Những Thách thức và Hạn chế Gặp phải

Bất chấp những thành công, dự án đã phải đối mặt với một số thách thức trong thực tế, đây cũng là những hạn chế của hệ thống hiện tại và mở ra các hướng cho công việc trong tương lai:

1. **Chất lượng Dữ liệu không đồng đều:** Việc phụ thuộc vào các camera giao thông công cộng từ một mạng lưới đô thị lớn có nghĩa là phải đối phó với sự không nhất quán đáng kể về chất lượng dữ liệu. Các thách thức bao gồm các luồng video có độ phân giải thấp, các lỗi do nén, và các camera bị che khuất bởi thời tiết (mưa) hoặc bụi bẩn, tất cả đều ảnh hưởng tiêu cực đến độ chính xác của cả việc phát hiện và nhận dạng.
2. **Điều kiện Môi trường Khắc nghiệt:** Giao thông ở Thành phố Hồ Chí Minh hoạt động suốt ngày đêm. Hiệu năng của hệ thống bị thách thức bởi các điều kiện ánh sáng khắc nghiệt, chẳng hạn như ánh sáng yếu lúc bình minh/hoàng hôn, ánh nắng chói chang giữa trưa, và cảnh đêm với ánh đèn pha lóa. Những điều kiện này có thể làm suy giảm đáng kể chất lượng của các đặc trưng được trích xuất bởi các mô hình AI.
3. **Tình trạng Che khuất Nghiêm trọng:** Mật độ giao thông cao, đặc biệt là sự phổ biến của xe máy, dẫn đến tình trạng che khuất thường xuyên và nghiêm trọng, nơi một phương tiện mục tiêu bị che khuất một phần hoặc gần như hoàn toàn bởi các phương tiện khác. Mặc dù bộ theo dõi BoT-SORT có thể xử lý các trường hợp che khuất ngắn hạn, tình trạng che khuất kéo dài hoặc hoàn toàn vẫn là một thách thức lớn để duy trì việc theo dõi ổn định.
4. **Sự Đa dạng và Tình trạng của Biển số xe:** Mô-đun LPR đã phải đối mặt với những thách thức với các biển số xe không theo chuẩn, các biển số bị cong hoặc bẩn, và các ký tự bị mờ. Những khiếm khuyết vật lý này là một nguồn gây ra lỗi OCR đáng kể.

Những hạn chế này không phải là thất bại trong thiết kế của hệ thống mà là những thách thức cố hữu khi áp dụng thị giác máy tính vào một môi trường không được kiểm soát trong thế giới thực. Chúng nêu bật những lĩnh vực mà các nỗ lực nghiên cứu và thu thập dữ liệu trong tương lai có thể tập trung vào để nâng cao hơn nữa tính mạnh mẽ của hệ thống.

### CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Chương cuối cùng này tổng hợp các kết quả của dự án nghiên cứu, cung cấp những phát hiện chính và đánh giá chúng dựa trên các mục tiêu ban đầu. Chương này cung cấp một bản tóm tắt kết luận về những đóng góp của công trình, thừa nhận những hạn chế của nó, và đề xuất một lộ trình rõ ràng cho các nghiên cứu và phát triển trong tương lai có thể xây dựng dựa trên nền tảng vững chắc đã được thiết lập bởi dự án này.

#### 5.1. Kết luận

### 5.1.1. Tóm tắt các Kết quả Đạt được

Nghiên cứu này đã thiết kế, triển khai và đánh giá một cách nghiêm ngặt một hệ thống Theo dõi Đa đối tượng qua Nhiều Camera (MC-MOT) hoàn chỉnh, từ đầu đến cuối cho việc giám sát phương tiện. Hệ thống tích hợp một quy trình gồm các mô hình học sâu tiên tiến nhất để thực hiện phát hiện phương tiện (YOLOv12), theo dõi trong một camera (BoT-SORT), nhận dạng biển số xe (YOLOv12 + CCT), và tái nhận dạng hình ảnh giữa các camera (Học biểu diễn Đa nhánh).

Thành tựu trung tâm và quan trọng nhất của dự án này là việc tối ưu hóa thành công quy trình AI phức tạp này bằng cách sử dụng Bộ công cụ Intel® OpenVINO™. Thông qua một quy trình chuyển đổi và triển khai có hệ thống, hiệu năng của hệ thống trên phần cứng CPU tiêu chuẩn đã được biến đổi từ mức cơ sở không khả thi là **0.17 FPS** lên mức có khả năng hoạt động thời gian thực là **15.23 FPS**. Điều này tương đương với mức tăng thông lượng 8858% và giảm độ trễ tương ứng là **94.3%**. Kết quả này cung cấp một sự xác thực định lượng mạnh mẽ cho giả thuyết rằng các bộ công cụ suy luận chuyên dụng có thể khai phá hiệu năng tương tự GPU trên cơ sở hạ tầng CPU phổ biến và hiệu quả về chi phí.

### 5.1.2. Đánh giá lại các Mục tiêu

Dự án đã đáp ứng thành công hoặc vượt qua tất cả các mục tiêu cụ thể đã được vạch ra trong Chương 1. Một quy trình mạnh mẽ gồm các mô hình AI chuyên dụng đã được huấn luyện và xác thực. Bộ theo dõi BoT-SORT đã được triển khai hiệu quả để theo dõi ổn định trên một camera. Toàn bộ quy trình suy luận đã được tối ưu hóa một cách có hệ thống và thành công bằng cách sử dụng OpenVINO™. Cuối cùng, một bài đo lường hiệu năng toàn diện đã được tiến hành, không chỉ đo lường mà còn làm nổi bật mức tăng hiệu năng mang tính chuyển đổi, qua đó đạt được mục tiêu chính của dự án.

### 5.1.3. Phát biểu cuối cùng về các Đóng góp

Dự án này tạo ra một đóng góp đáng kể cho lĩnh vực AI ứng dụng trong các hệ thống giao thông thông minh. Về mặt khoa học, nó đóng vai trò như một nghiên cứu tình huống điển hình về hiệu quả của OpenVINO™ trong việc tối ưu hóa các quy trình AI phức tạp, đa giai đoạn, cung cấp một tài liệu tham khảo quý giá cho các nhà nghiên cứu và kỹ sư. Về mặt thực tiễn, nó cung cấp một bản thiết kế hiệu năng cao, hiệu quả về chi phí cho một hệ thống giám sát giao thông có khả năng mở rộng. Bằng cách chứng minh rằng hiệu năng thời gian thực có thể đạt được mà không cần phụ thuộc vào phần cứng chuyên dụng, đắt tiền, công trình này đã làm giảm đáng kể rào cản gia nhập cho việc triển khai các giải pháp AI tiên tiến trong các sáng kiến thành phố thông minh tại Việt Nam và hơn thế nữa.

## 5.2. Hướng Phát triển và các Cải tiến Tiềm năng

Mặc dù dự án này đã thiết lập một hệ thống mạnh mẽ và hiệu quả, nó cũng mở ra nhiều hướng đi cho nghiên cứu và phát triển trong tương lai. Các hướng đi sau đây được đề xuất để nâng cao hơn nữa năng lực và tính mạnh mẽ của hệ thống.

### 5.2.1. Cải tiến Mô hình và Dữ liệu:

1. **Làm giàu Bộ dữ liệu:** Độ chính xác của hệ thống có thể được cải thiện hơn nữa bằng cách liên tục mở rộng các bộ dữ liệu huấn luyện, đặc biệt là với các kịch bản khó hơn được ghi lại từ mạng lưới camera TP.HCM. Điều này bao gồm nhiều ví dụ hơn về điều kiện ban đêm, thời tiết khắc nghiệt (mưa lớn), và các phương tiện ở những góc độ khó.
2. **Các Kiến trúc Tái nhận dạng Tiên tiến:** Công việc trong tương lai có thể khám phá việc tích hợp các mô hình Tái nhận dạng tiên tiến hơn nữa, chẳng hạn như những mô hình mô hình hóa rõ ràng các bộ phận của phương tiện hoặc sử dụng các kiến trúc dựa trên Transformer để hiểu bối cảnh một cách toàn diện hơn.

#### 5.2.2. Mở rộng Chức năng:

- **Nhận dạng Thuộc tính Phương tiện:** Hệ thống có thể được mở rộng để nhận dạng các thuộc tính bổ sung của phương tiện ngoài biển số xe. Điều này bao gồm phân loại loại phương tiện (phân biệt giữa sedan, SUV, xe buýt, v.v.), nhận dạng màu sắc, và thậm chí xác định logo hoặc các dấu hiệu khác. Điều này sẽ cung cấp siêu dữ liệu phong phú hơn cho việc truy vấn và phân tích.
- **Phân tích Hành vi:** Với việc theo dõi ổn định đã có, hệ thống có thể được tăng cường để thực hiện phân tích hành vi, chẳng hạn như ước tính tốc độ, phát hiện các hành vi di chuyển trái phép (ví dụ: rẽ trái sai quy định, đi vào làn đường cấm), hoặc xác định các bất thường như một phương tiện dừng trong làn đường đang lưu thông.

#### 5.2.3. Triển khai Phần cứng và Điện toán Biên:

- **Triển khai trên VPU:** Bộ công cụ OpenVINO™ được thiết kế để triển khai trên nhiều nền tảng. Một bước tiếp theo hấp dẫn sẽ là triển khai hệ thống đã được tối ưu hóa trên một thiết bị biên chuyên dụng, công suất thấp được trang bị Bộ xử lý Tâm nhìn (VPU) Intel® Movidius™. Điều này sẽ cho phép tạo ra các camera thông minh có thể thực hiện tất cả các xử lý AI phức tạp “tại biên”, giảm yêu cầu về băng thông mạng và cho phép một kiến trúc phân tán và có khả năng mở rộng hơn.

#### 5.2.4. Tích hợp Hệ thống và Giao diện Người dùng:

1. **Giao diện Người dùng Đồ họa (GUI):** Một ứng dụng phần mềm hoàn chỉnh có thể được xây dựng xung quanh quy trình cốt lõi, có giao diện người dùng đồ họa thân thiện. Giao diện này sẽ cho phép người vận hành xem các luồng camera trực tiếp với các chú thích thời gian thực, truy vấn hệ thống để tìm một phương tiện cụ thể (theo biển số xe hoặc sự tương đồng hình ảnh), và trực quan hóa quỹ đạo của nó trên một bản đồ tương tác. Điều này sẽ biến bằng chứng về khái niệm hiện tại thành một công cụ giám sát và phân tích hoạt động đầy đủ.

Bằng cách theo đuổi những hướng đi trong tương lai này, công trình nền tảng của dự án này có thể được phát triển thành một hệ thống giao thông thông minh toàn diện, đa diện với tác động và tiện ích còn lớn hơn nữa.