**4.2 Hiện thực các module**

**4.2.1 Module tiền xử lý dữ liệu**

Để đảm bảo độ chính xác, công việc tiền xử lý dữ liệu cần phải được thực hiện đồng bộ cho cả quá trình huấn luyện, đánh giá và sử dụng của hệ thống trích xuất hình ảnh. Ở cả ba quá trình trên, hình ảnh khi đầu và sẽ được điều chỉnh kích thước về kích thước chuẩn của hệ thống là 224 x 224 x 3 pixel. Đồng thời, giá trị đầu vào của từng pixel sẽ được chuẩn hóa bằng cách chia cho 255 và trừ cho mean của toàn bộ pixel của hình đó, nghĩa là, mean của từng hình sẽ bằng 0. Điều này giúp giảm khoảng giá trị của các pixel về từ [0, 255] về [-0.5, 0.5], tạo thuận lợi lớn cho quá trình huấn luyện mô hình.

**4.2.2 Module trích xuất đặc trưng hình ảnh**

Trong phương pháp mà nhóm hiện thực, module trích xuất đăc trưng hình ảnh chính là một CNN. Để có thể tạo ra một CNN trích xuất đặc trưng hiệu quả, nhóm phải tiến hành quá trình huấn luyện và đánh giá một cách khách quan. Quá trình này gồm ba bước chính là: thu thập dữ liệu, huấn luyện CNN cho bài toán phân loại ảnh, và cuối cùng là tinh chỉnh CNN với hàm triplet loss nhắm giúp mạng đạt được hiệu quả cao hơn trong nhiệm vụ trích xuất đặc trưng ảnh.

***a. Thu thập dữ liệu***

Sau quá trình tìm kiếm tập dữ liệu phù hợp với đề tài, nhóm đã quyết định chọn tập hình ảnh Stanford Online Products để sử dụng cho quá trình huấn luyện và đánh giá kết quả. Đây là bộ giữ liệu được các tác giả ở đại học Stanford lấy về từ eBay.com, bao gồm 120.053 bức ảnh từ 22.634 loại sản phẩm khác nhau và được chia thành 12 nhóm sản phẩm chính. Đây là một bộ dữ liệu với số lượng hình ảnh rất lớn, tuy nhiên, phạm vi của luận văn cùng với những hạn chế về mặt tài nguyên tính toán, nhóm đã quyết định chỉ sử dụng 2000 bức ảnh thuộc về hai class là quạt (fan) và ấm nước (kettle) để huấn luyện cho mô hình của nhóm. Dưới đây là một vài hình ảnh của 2 class nhóm lựa chọn:

{Hình ảnh quạt} {Hình ảnh ấm nước}

Sau khi đã chọn được dữ liệu, nhóm tiến hành phân loại dữ liệu thành tập train và test dựa vào thông tin được cung cấp sẵn trong bộ Stanford Online Products. Trong đó, tập train được sử dụng để giúp mô hình học được khả năng trích xuất vector đặc trưng của hình ảnh. Tập test được dùng để đo lường độ hiệu quả sau cùng của mô hình, chính vì vậy, tập dữ liệu này được tách riêng hoàn toàn khỏi quá trình huấn luyện nhằm đảm bảo độ chính xác cho công việc đánh giá kết quả. Từ tập train, nhóm tiếp tục tách ra một tập nhỏ khác, gọi là tập validation. Tập validation này giúp nhóm đánh giá được khả năng học của mô hình ngay tại thời điểm huấn luyện, từ đó đưa ra các điều chỉnh cần thiết để cải thiện hiệu quả của mô hình. Cuối cùng, từ tập dữ liệu ban đầu, nhóm thu được ba tập dữ liệu nhỏ hơn như sau:

|  |  |  |
| --- | --- | --- |
| **Tập dữ liệu** | **Quạt** | **Ấm nước** |
| Train | 600 hình | 600 hình |
| Validation | 200 hình | 200 hình |
| Test | 200 hình | 200 hình |

Như vậy, nhóm đã hoàn tất quá trình thu thập dữ liệu, bước tiếp là tiến hành xây dựng và huấn luyện mô hình CNN cho bài toán phân loại hình ảnh.

***b. Huấn luyện CNN cho bài toán phân loại ảnh***

Ở bước này, nhóm bắt đầu với việc sử dụng kiến trúc mạng VGG16 để huấn luyện. Tuy nhiên với kiến trúc này, việc huấn luyện diễn ra khá lâu, nếu kết quả không tố thì cần phải tốn nhiều thời gian để huấn luyện lại. Do đó, nhóm quyết định đổi sang sử dụng kiến trúc VGG chỉ gồm 11 layer, đây cũng là kiến trúc được team VGG sử dụng để khởi tạo trọng số cho các mạng VGG lớn hơn như VGG13 hay VGG16. Thay đổi này giúp cho việc huấn luyện diễn ra nhanh hơn, qua đó, nhóm có thể quan sát được kết quả sớm và đưa ra các điều chỉnh phù hợp. Bảng dưới đây thể hiện kiến trúc CNN được nhóm sử dụng:

Layer (type) Output Shape Param #

=================================================================

block1\_conv1 (Conv2D) (None, 224, 224, 64) 1792

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_1 (Activation) (None, 224, 224, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_1 (MaxPooling2 (None, 112, 112, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block2\_conv1 (Conv2D) (None, 112, 112, 128) 73856

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_2 (Activation) (None, 112, 112, 128) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_2 (MaxPooling2 (None, 56, 56, 128) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block3\_conv1 (Conv2D) (None, 56, 56, 256) 295168

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_3 (Activation) (None, 56, 56, 256) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block3\_conv2 (Conv2D) (None, 56, 56, 256) 590080

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_4 (Activation) (None, 56, 56, 256) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_3 (MaxPooling2 (None, 28, 28, 256) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block4\_conv1 (Conv2D) (None, 28, 28, 512) 1180160

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_5 (Activation) (None, 28, 28, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block4\_conv2 (Conv2D) (None, 28, 28, 512) 2359808

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_6 (Activation) (None, 28, 28, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_4 (MaxPooling2 (None, 14, 14, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block5\_conv1 (Conv2D) (None, 14, 14, 512) 2359808

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_7 (Activation) (None, 14, 14, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

block5\_conv2 (Conv2D) (None, 14, 14, 512) 2359808

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

activation\_8 (Activation) (None, 14, 14, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_5 (MaxPooling2 (None, 7, 7, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout\_1 (Dropout) (None, 7, 7, 512) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten\_1 (Flatten) (None, 25088) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

fc1 (Dense) (None, 4096) 102764544

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout\_2 (Dropout) (None, 4096) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

fc2 (Dense) (None, 4096) 16781312

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

predictions (Dense) (None, 2) 8194

=================================================================

Total params: 128,774,530

Trainable params: 128,774,530

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ngoài kiến trúc của mạng, thì các siêu tham số được sử dụng cũng đóng vai trò quan trọng vào kết quả của việc huấn luyện CNN. Do đó, nhóm đã phải tiến hành việc huấn luyện rất nhiều lần để tìm ra bộ siêu tham số mang lại kết quả tốt, đạt được mục tiêu đặt ra cho việc huấn luyện mạng là độ chính xác 90%, mAP từ 75%. Cuối cùng, nhóm đã sử dụng bộ siêu tham số như sau cho việc huấn luyện mạng:

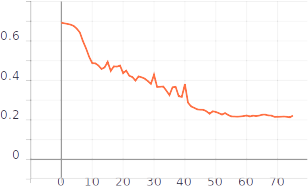
batch size: 32

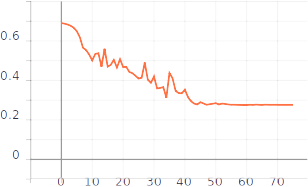
learning rate: 0.001

momentum: 0.9

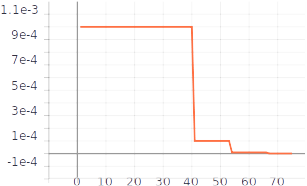
Ngoài ra, nhóm còn sử dụng các kỹ thuật early stopping và learning rate schedule giảm tình trạng overfiting và tăng hiệu quả học của mạng. Cụ thể, nhóm cài đặt keras để theo dõi loss của tập validation, nếu sau 5 epochs mà validation loss không giảm thì mô hình sẽ tự động giảm learning rate xuống 10 lần (lr = lr \* 0.1) và giảm tối thiểu tới 1x10-6. Việc này giúp quá trình học chậm lại, tăng khả năng tiếp cận điểm cực tiểu của hàm số loss. Ngoài ra, nếu sau 15 epochs, mô hình vẫn không thể cải thiện validation loss, thì keras sẽ tự động dừng việc huấn luyện lại và trả các trọng số lại trạng thái ở epoch mà mô hình đạt validation loss thấp nhất. Đây là kỹ thuật early stopping được dùng phổ biến trong huấn luyện mô hình để tránh tình trạng overfiting.

Sau khi đã cài đặt đầy đủ các thông số, nhóm bắt đầu huấn luyện mô hình với 200 epochs. Ngoài ra, để có thể theo dõi các thông số loss và validation của quá trình huấn luyện sau từng epoch, nhóm đã thêm cài đặt tensorboard vào mô hình, và sau đây là đồ thì training loss cùng validation loss:

  
Illustration 1: Training loss

  
Illustration 2: Validation loss

Qua đồ thị trên, có thể quan sát thấy là mô hình học tương đối tốt trong 10 epochs đầu tiên, khi mà cả train loss và validation loss đều giảm đều đặn. Từ epoch thứ 10 đến 35, đồ thị loss bắt đầu có những đoạn lên xuống bất thường, không hoàn toàn đi xuống như những epochs đầu. Tuy nhiên, nhìn chung thì loss vẫn đang có xu hướng giảm, bằng chứng rõ nhất là learning rate vẫn giữ giá trị 0.001 như lúc bắt đầu huấn luyện, tức là sau ít nhất 5 epochs, validation loss vẫn đạt được một giá trị nhỏ nhất mới. Ta có thể quan sát đồ thị sự thay đổi của learning rate qua hình dưới đây:

  
Illustration 3: Learning rate

Như vậy, theo đồ thị, learning rate bắt đầu giảm từ epoch 40, điều này phù hợp với quan sát từ đồ thị validation loss phía trên. Việc giảm learning rate này đã tỏ ra có hiệu quả khi loss tiếp tục giảm thêm khoảng 15 epochs nữa, trước khi có xu hướng đi ngang. Trong khoảng epoch 55 tới 75, validation loss không còn giảm đáng kể, dù learning rate tiếp tục giảm thêm 2 lần nữa, cuối cùng quá trình huấn luyện được tự động kết thúc sau 76 epochs. Nhờ nhóm đã cài đặt từ trước, keras tự động trả về mô hình tại epoch 61, epoch mà mô hình đat được validation loss thấp nhất là: 0.27537, đồng thời tại epoch này, độ chính xác của tập validation là 89.75%. Các thông số khác cụ thể như sau:

|  |  |  |
| --- | --- | --- |
|  | **Train** | **Validation** |
| **Loss** | 0.2156 | 0.27537 |
| **Accuracy** | 0.9161 | 89.75% |

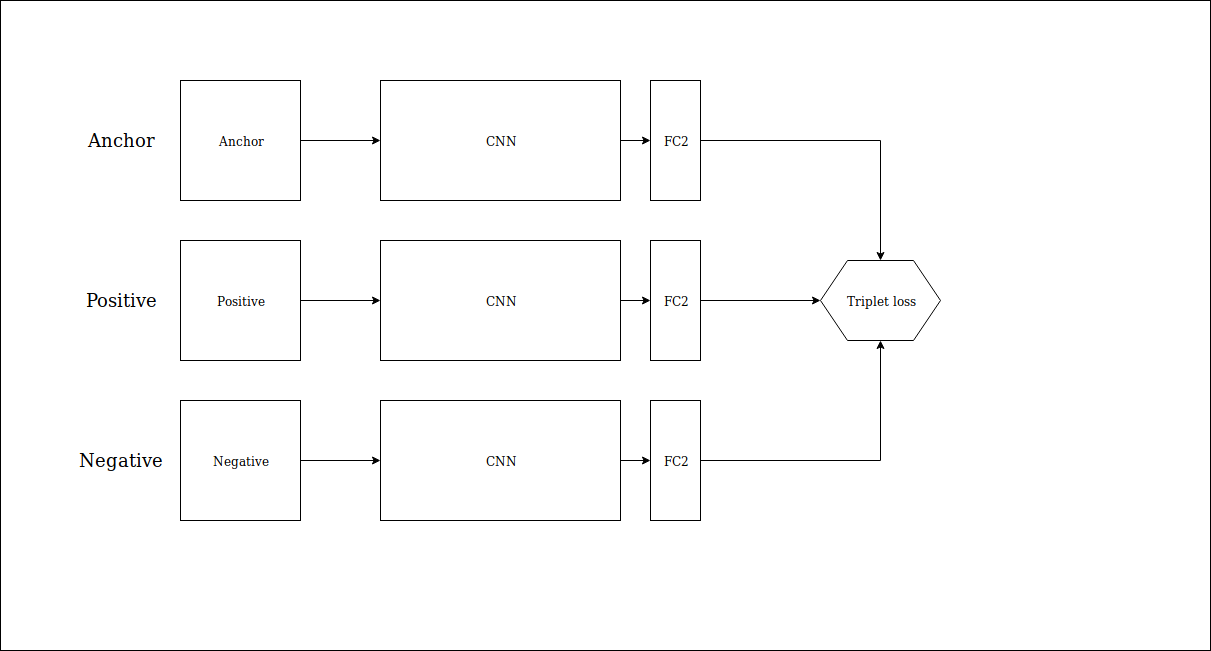
Ngoài việc sử dụng thông số về độ chính xác để đo lường hiệu quả của CNN ở bài toán phân loại ảnh, thì nhóm còn sử dụng phương pháp đo lường khác là mean average precision (mAP) và precision at K để đo lường khả năng trích xuất các hình ảnh tương tự, kết quả đo khi sử dụng layer FC2 làm vector features như sau:

|  |  |  |
| --- | --- | --- |
|  | **Không dùng Kmeans** | **Dùng Kmeans** |
| mAP | 0.779 | 0.779 |
| P@10 | 0.865 |  |
| p@30 | 0.844 |  |
| P@50 | 0.831 |  |

Nhận thấy kết quả huấn luyện CNN đã đạt được mục tiêu ban đầu đề ra, nhóm quyết định lưu mô hình lại để tiếp tục tinh chỉnh với triplet loss, chi tiết sẽ được đề cập ngay sau đây.

***c. Tinh chỉnh với triplet loss***

Sau khi thu được một mô hình CNN với kết quả tương đối tốt, nhóm thực hiện tinh chỉnh mô hình với triplet loss. Nhóm sẽ sử dụng một mô hình gồm 3 mạng CNN được huấn luyện từ bước trước để tạo đầu vào cho bước tinh chỉnh này, mô hình cụ thể như sau:

Để huấn luyện mô hình triplet loss, đòi hỏi nhóm phải cung cấp các triplet gồm 3 hình ảnh: 1 hình chuẩn (anchor), 1 hình cùng class (positive) và một hình khác class (negative). Các triplet này có thể được tạo ra từ tập train bằng cách kết hợp tất cả các trường hợp có thể lại. Tuy nhiên, tổ hợp các triplet là mộ con số rất lớn, khi mà với 2 class, mỗi class gồm 600 hình, số triplet có thể tạo ra sẽ là 2 \* 600 \* 599 \* 600 = 431.280.000, hơn 430 triệu triplet!!!

Để giảm số triplet cần thiết nhưng vẫn giữ được độ hiệu quả, nhóm chỉ tạo ra 1 triplet cho mỗi hình, đồng thời số triplet này được tạo ra dựa theo từng batch hình được đưa vào, cụ thể như sau: với mỗi batch gồm 16 hình “quạt” và 16 hình “ấm nước”, nhóm sẽ tạo ra 2\*16 = 32 triplet sao cho với mỗi hình anchor (A), nhóm sẽ tìm một positive (P) để d(A, P) là lớn nhất trong số tất cả các cặp A, P tìm được trong batch; tương tự, nhóm cũng tìm một negative N sao cho d(A, N) là nhỏ nhất. Cuối cùng, nhóm sẽ loại bỏ triplet nào không thỏa điều kiện d(A, P) - d(A, N) + margin >= 0. Lý do là vì khoảng cách giữa 3 hình trong những triplet này đã đúng như mong muốn, không cần điều chỉnh nữa.

Như vậy, sau khi đã có được phương pháp để tạo các triplet, nhóm bắt đầu thực tinh chỉnh mô hình. Với việc mô hình CNN đã có những kết quả khá tốt, nhóm quyết định chỉ thực hiện huấn luyện trên 2 layer fully connected cuối cùng của mạng và giữ nguyên các trọng số ở những layer còn lại. Sau 5 epochs thì các kết quả đã có sự cải thiện rõ ràng như bảng dưới đây:

|  |  |  |
| --- | --- | --- |
|  | **Không dùng Kmeans** | **Dùng Kmeans** |
| mAP | 0.827 | 0.840 |
| P@10 | 0.870 |  |
| p@30 | 0.855 |  |
| P@50 | 0.849 |  |