INT3412E 20 - Computer Vision: Báo cáo giữa kì

Nhóm 13

1 Đánh giá về bài báo đã chọn

Aggregating Local Descriptors Into a Compact Image Representation

Link Google Colab: **Thị giác máy (INT3412E 20) - Nhóm 13)**

1.1. Tóm tắt về bài báo:
Bài báo này trình bày một phương pháp tổng hợp các mô tả hình ảnh cục bộ thành một biểu diễn nhỏ gọn, đạt được độ chính xác tìm kiếm cao, hiệu suất và sử dụng bộ nhớ tốt cho việc tìm kiếm hình ảnh quy mô lớn.

1.2. Bản phác thảo sâu hơn, rộng hơn về các điểm chính của bài viết, bao gồm các giả định được đưa ra, lập luận được trình bày, dữ liệu được phân tích và kết luận được rút ra:

(a) Các giả định được đưa ra:

- Bài báo xác định 3 điều kiện để hoàn thiện phương pháp tìm kiếm hình ảnh:
  - Độ chính xác
  - Độ hiệu quả
  - Dung lượng sử dụng của biểu diễn ảnh
- Bài báo đã đưa ra những ưu điểm của các biện pháp có sẵn trong việc xử lý 3 yếu tố nói trên và mặt hạn chế của nó:
  - BOF
    - Phương pháp này có nhiều điểm mạnh và được sử dụng bởi những thuật toán phân loại mạnh mẽ.
    - Tuy nhiên trong việc tìm kiếm quy mô lớn thì độ hiệu quả quả và dung lượng tiêu tốn sẽ gặp vấn đề khi phải đối mặt với hơn 10 triệu ảnh.
  - Packing BOF
    - Phương pháp này giúp cải thiện độ hiệu quả cũng như dung lượng tiêu tốn tuy nhiên bị giới hạn bởi việc kích thước từ điển nhỏ dẫn tới độ chính xác thấp
  - Min-Hash approach và phương pháp của Torressani và cộng sự
    - Những phương pháp trên đã sử lý được độ hiệu quả tuy nhiên những kỹ thuật này vẫn đòi hỏi một lượng lớn dung lượng đối với mỗi ảnh.
  - Một vài người dùng GIST descriptors và chuyển đổi chúng sang vector nhị thể tuy nhiên cũng đều không thể thỏa mãn được ba điều kiện được nói trên.
    $\Longrightarrow$ Từ những dẫn chứng về những phương pháp trên tác giả đã đưa ra biện pháp đối với 3 điều kiện được đặt ra bằng cách tối ưu hóa:
    - Biểu diễn ảnh
    - Giảm không gian của những vector này
    - Các thuật toán chỉ mục
- Phương pháp BOF (Bag of features)
  - Phương pháp này trích xuất các đặc trưng cục bộ từ các điểm đặc trung trên hình ảnh.
  - Các đặc trưng cục bộ này sẽ được phân cụm rồi được biểu diễn thành các vector.

  - Từ đó các vector này sẽ được chuẩn hóa để loại bỏ các ảnh hưởng của kích thước ảnh hoặc số lượng.
  - Có rất nhiều phương pháp để chuẩn hóa BOF vector:
    * Manhattan
    * Euclidean
  - Trong bài báo này, sử dụng L2 để chuẩn hóa và sử dụng idf để tính toán.
  - Ưu điểm:
    * Dễ hiểu và triển khai.
    * Thích hợp cho các tác vụ đơn giản như phân loại hình ảnh.
  - Nhược điểm:
    * Mất thông tin về cấu trúc không gian của các đặc trưng trong hình ảnh.
    * Không xử lý sự biến đổi không gian và quy mô của đối tượng.

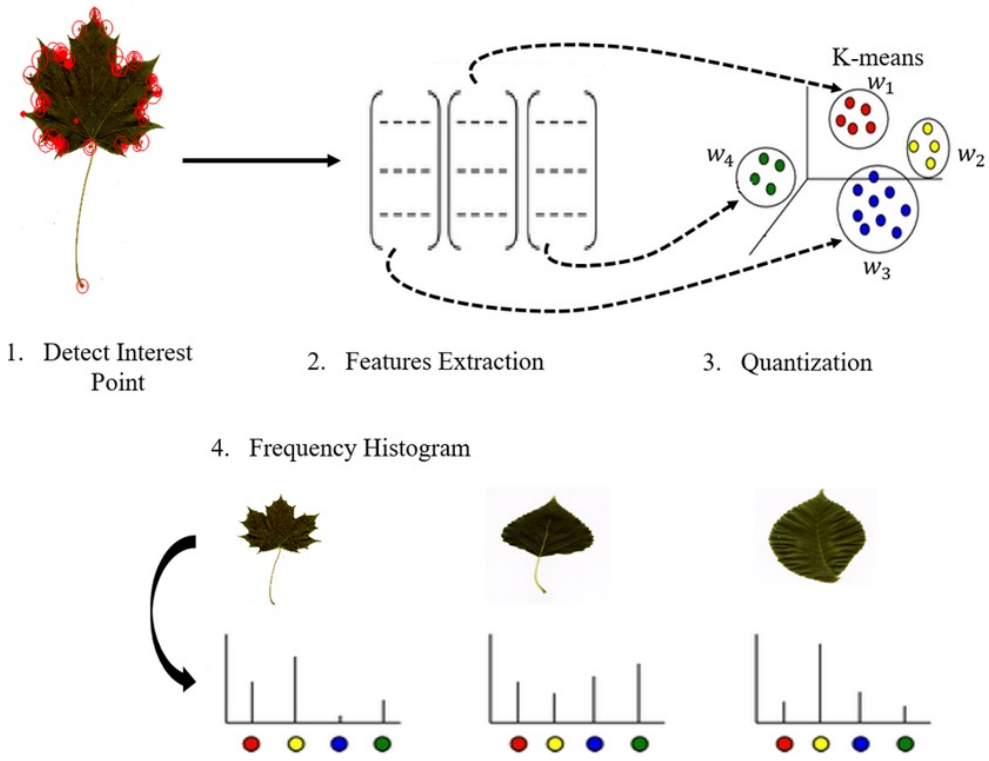


Figure 1: BOF

- Phương pháp Fisher kernel
  - Phương pháp Fisher kernel là một công cụ mãnh mẽ giúp chuyển đổi các tập hợp có kích thước với giá trị thay đổi thành các biểu diễn vector có kích thước cố định với điều kiện tuan theo một mô hình sinh tham số được ước lượng trên tập dữ liệu huấn luyện.
  - Vector biểu diễn này chỉ ra hướng trong không gian tham số mà phân phối đã học được để điều chỉnh khớp với dữ liệu.
  - Ưu điểm:
    * Tận dụng thông tin thống kê về sự phân phối của đặc trưng.
    * Cung cấp một biểu diễn mạnh mẽ hơn so với BoF.
  - Nhược điểm:

* Đòi hỏi một số kỹ thuật và tính toán phức tạp hơn so với BoF.
* Yêu cầu mô hình học máy cụ thể để tích hợp Fisher Kernel.

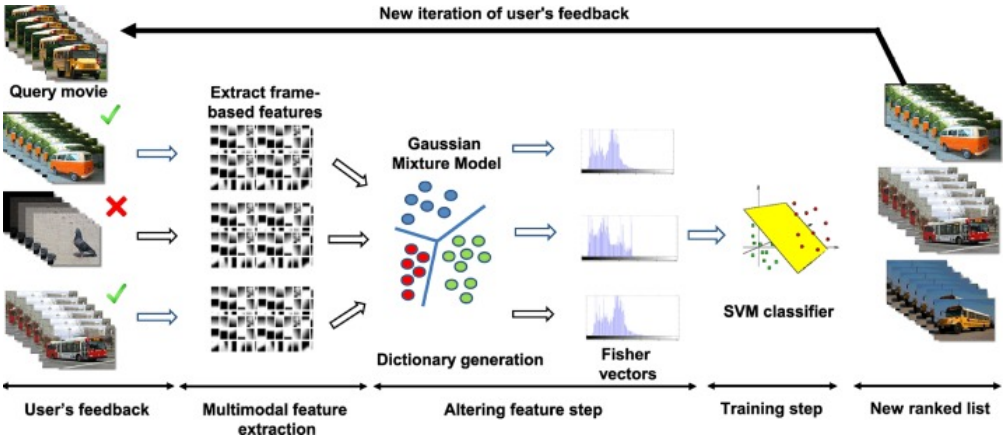


Figure 2: Fisher kernel

- Phương pháp VLAD (Vector of locally aggregated descriptors)
  - Được xây dựng theo quy tắc được thiết lập để đánh giá sự gần gũi của các điểm dữ liệu trong không gian đặc trưng. Nó có thể được hiểu là phương pháp đơn giản của Fisher kernel.
  - Giống như BOF, phương pháp VLAD cũng nhận tập hợp các từ từ hình ảnh. Những đặc trưng hình ảnh này liên kết với những từ hình ảnh gần nhất. Mỗi đặc trưng trong cụm sẽ được sử dụng để tính toán độ chênh lệch với trung bình của các đặc trưng đó. Độ chênh lệch này sau đó được ghép thành một vector VLAD.
  - Mỗi chiều của VLAD đại diện cho một "visual word".
  - Ý tưởng của mô tả VLAD là tổng hợp khoảng cách từ vector x tới ci rồi gán cho ci.
  - Giả định rằng mô tả cục bộ có d chiều thì biểu diễn của chúng ta sẽ có D = k x d chiều. Biểu diễn mô tả này bởi vector vi,j với i =1,...k và j=1...d (i là chỉ số của visual word và j là chỉ số của các thành phần của mô tả cục bộ.Từ đây, thành phần của vector v sẽ được cấu thành tự tổng của các mô tả hình ảnh.

$$v_{i,j} = \sum_{x \text{ such that } \mathrm{NN}(x)=c_i} x_j - c_{i,j}$$

Figure 3: Vector v

  - Vector VLAD sau đó được chuẩn hóa bởi phương pháp L2.
  - Ưu điểm:
    * Giữ lại thông tin về cấu trúc không gian của các đặc trưng.
    * Tích hợp thông tin về sự chênh lệch giữa các đặc trưng và các điểm trung tâm.
    * Hiệu suất tốt trong các nhiệm vụ gán nhãn và phân loại hình ảnh.
  - Nhược điểm:
    * Có thể đòi hỏi bộ dữ liệu lớn hơn để đào tạo mô hình hiệu quả.

∗ Tính toán phức tạp hơn so với BoF.

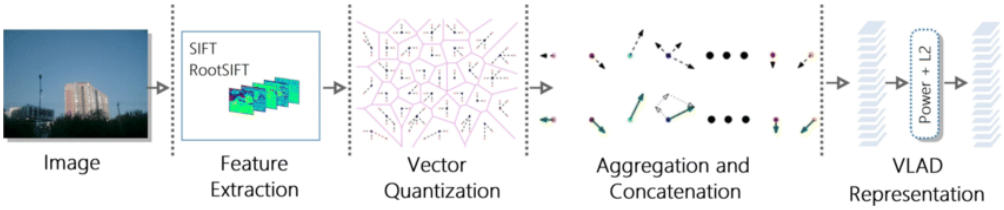


Figure 4: VLAD

- Approximate nearest neighbor:
  Phương pháp này mang lại độ chính xác tốt và thuật toán tìm kiếm cung cấp một sự xấp xỉ rõ ràng của các vector được đánh chỉ mục. Điều này cho phép so sánh sự xấp xỉ vector được giới thiệu bởi việc giảm chiều và quantization. Các tác giả sử dụng biến thể của phương pháp này là "asymmetric distance computation" (ADC), chỉ mã hóa các vector của cơ sở dữ liệu mà không mã hóa vector truy vấn.
  Cho $x \in \mathbb{R}^D$ là một vector truy vấn và $Y = y_1, \ldots, y_n$ là một tập hợp các vector mà chúng ta muốn tìm nearest neighbor $\mathrm{NN}(x)$ của x. Phương pháp ADC bao gồm việc mã hóa mỗi vector y_i thành phiên bản đã được lượng tử hóa $c_i = q(y_i) \in \mathbb{R}^D$. Với một bộ lượng tử $q(\cdot)$ có k trọng tâm, vector được mã hóa bởi $\log_2(k)$ bit, trong đó k là một lũy thừa của 2. Việc tìm nearest neighbor $\mathrm{NNa}(x)$ của x đơn giản chỉ bao gồm việc tính toán phiên bản đã được lượng tử hóa $q_x = q(x)$ của vector truy vấn x và sau đó so sánh nó với các phiên bản đã được lượng tử hóa của tất cả các vector trong Y để tìm nearest neighbor.

$$\mathbf{NN}_a(x) = a\text{-}\arg\min_i ||x - q(y_i)||^2.$$

Figure 5

Trong đó:

$$||x - q(y_i)||^2 = \sum_{j=1,\ldots,m} ||x^j - q_j(y_i^j)||^2,$$

Figure 6

trong đó $\mathbf{y}_i^j$ là subvector thứ j của $\mathbf{y}_i$.
Các khoảng cách bình phương trong tổng này được đọc từ các bảng tra cứu được tính toán, trước khi tìm kiếm, giữa mỗi subvector $\mathbf{x}_j$ và k_s trọng tâm liên quan đến bộ lượng tử tương ứng q_j. Việc tạo ra các bảng này có độ phức tạp là $O(Dk_s)$.
Để có được một xấp xỉ vector tốt, k nên lớn ($k = 2^{64}$ cho một mã 64 bit). Với những giá trị k lớn như vậy, việc học một bộ mã hóa k-means cùng với việc gán cho các trọng tâm trở nên không khả thi. Giải pháp của các tác giả là sử dụng phương pháp lượng tử hóa sản phẩm (product quantization), trong đó định nghĩa bộ lượng tử mà không cần liệt kê rõ các trọng tâm. Một vector $\mathbf{x}$ được chia thành m subvector $\mathbf{x}^1, \ldots, \mathbf{x}^m$ có cùng độ dài D/m. Sau đó, một bộ lượng tử hóa sản phẩm được định nghĩa dưới dạng hàm:

$$q(x) = \big(q_1(x^1), ..., q_m(x^m)\big)$$

Figure 7

mà ánh xạ vector đầu vào **x** thành một bộ các chỉ số bằng cách lượng tử hóa riêng các subvector. Mỗi bộ lượng tử riêng biệt $q_j(\cdot)$ có k_s giá trị tái tạo được học thông qua k-means. Để giới hạn độ phức tạp của việc gán (assignment) là $O(mk_s)$, k_s có giá trị nhỏ (ví dụ: $k_s = 256$). Tuy nhiên, tập hợp các trọng tâm k được tạo ra bởi bộ lượng tử hóa sản phẩm $q(\cdot)$ là lớn: $k = (k_s)^m$.
Phương pháp lượng tử hóa này được chọn vì hiệu suất tuyệt vời của nó, nhưng cũng vì nó biểu diễn chỉ mục dưới dạng xấp xỉ vector: một vector cơ sở dữ liệu y_i có thể được phân rã thành:

$$y_i = q(y_i) + \varepsilon_{\mathbf{q}}(y_i),$$

Figure 8

trong đó $q(y_i)$ là trọng tâm được liên kết với y_i và $\epsilon q(y_i)$ là vector lỗi được tạo ra bởi bộ lượng tử.

- Indexation-aware dimensionality reduction:
  Giảm chiều là một bước quan trọng trong tìm kiếm nearest neighbor gần xấp xỉ, vì nó ảnh hưởng đến phương pháp chỉ mục sau đó. Trong phần này, đối với phương pháp ADC, các tác giả biểu thị sự cân nhắc giữa hoạt động này và phương pháp chỉ mục bằng một độ đo chất lượng duy nhất: sai số xấp xỉ. Vì mục đích trình bày, các tác giả giả định rằng trung bình của các vector là vector không. Điều này gần như đúng đối với các vector VLAD.
  Phân tích thành phần chính (PCA) là một công cụ tiêu chuẩn để giảm chiều: các vector riêng liên quan đến D' giá trị riêng năng lượng cao nhất của ma trận hiệp phương sai vector thực nghiệm được sử dụng để định nghĩa ma trận M ánh xạ một vector $x \in \mathbb{R}^D$ thành một vector $x' = Mx \in \mathbb{R}^{D'}$. Ma trận M là phần trên cùng có kích thước D' x D của một ma trận trực giao.
  Phép giảm chiều này cũng có thể được hiểu trong không gian ban đầu như một phép chiếu. Trong trường hợp đó, x được xấp xỉ bởi:

$$x_{\mathbf{p}} = x - \varepsilon_{\mathbf{p}}(x)$$

Figure 9

trong đó vector lỗi $\epsilon_p(x)$ nằm trong không gian trống của M. Vector x_p liên quan đến x' thông qua giả nghịch đảo của M, tức là chuyển vị của M trong trường hợp này. Do đó, phép chiếu là $x_p = M^\top Mx$. Với mục đích chỉ mục, vector x' sau đó được mã hóa thành $q(x')$ sử dụng phương pháp ADC, cũng có thể được hiểu trong không gian ban đầu có số chiều D như xấp xỉ:

$$q(x_{\mathrm{p}}) = x - \varepsilon_{\mathrm{p}}(x) - \varepsilon_{\mathrm{q}}(x_{\mathrm{p}})$$

Figure 10

trong đó $\epsilon_p(x) \in \text{Null}(M)$ và $\epsilon_q(x_p) \in \text{Null}(M)^{\perp}$ (bởi vì bộ lượng tử ADC được học trong không gian chính) là 2 vector đối giao.

- Balancing the components' variance (Cân bằng phương sai của các thành phần)
  Các tác giả giải quyết vấn đề bằng cách thực hiện một phép biến đổi trực giao sau PCA. Nói cách khác, cho chiều D' và một vector X cần chỉ mục, các tác giả muốn tìm một ma trận trực giao Q sao cho các thành phần của vector đã biến đổi $X'' = QX' = QMX$ có phương sai bằng nhau. Việc tìm ma trận $\tilde{Q}$ nhằm giảm thiểu sai số lượng tử hóa kỳ vọng được đưa vào bởi phương pháp ADC:

$$\tilde{Q} = \arg\min_{Q} \mathbb{E}_X \left[||\varepsilon_{\mathrm{q}}(QMX)||^2 \right].$$

Figure 11

Tuy nhiên, bài toán tối ưu hóa này không thể giải quyết được, vì hàm mục tiêu yêu cầu học bộ lượng tử tích q(.) của cấu trúc ADC tại mỗi vòng lặp. Việc tìm một ma trận Q thỏa mãn mục tiêu cân bằng đơn giản được thực hiện bằng cách chọn nó dưới dạng một ma trận Householder:

$$Q = I - 2vv^{\top},$$

Figure 12

với việc tối ưu hóa được thực hiện trên các thành phần D' của v. Một lựa chọn đơn giản khác là tránh tối ưu hóa này bằng cách sử dụng một ma trận trực giao ngẫu nhiên kích thước D' × D'. Đối với các vector có số chiều cao, lựa chọn này là một giải pháp chấp nhận được đối với tiêu chí cân bằng của các tác giả.

- Joint optimization of reduction/indexing (Tối ưu hóa chung việc giảm/lập chỉ mục)
  Bây giờ chúng ta xem xét vấn đề thứ hai, tức là tối ưu hóa chiều D', đã đặt ràng buộc về số bit B được sử dụng để biểu diễn vector VLAD D chiều x, ví dụ $B = 128$ (16 byte). Khoảng cách Euclidean bình phương giữa giá trị tái tạo và x là tổng của hai lỗi $||\epsilon_p(x)||^2$ và $||\epsilon_q(x_p)||^2$, cả hai đều phụ thuộc vào D' đã chọn. Sai số bình phương trung bình $e(D')$ được đo lường thực nghiệm trên một tập hợp vector học L như sau:

$$\begin{aligned} e(D') &= e_p(D') + e_q(D') \\ &= \frac{1}{\text{card}(\mathcal{L})} \sum_{x \in \mathcal{L}} ||\varepsilon_{\mathrm{p}}(x)||^2 + ||\varepsilon_{\mathrm{q}}(x_{\mathrm{p}})||^2. \end{aligned}$$

Figure 13

Điều này cung cấp cho chúng ta một tiêu chí mục tiêu để tối ưu trực tiếp chiều số, được đạt được bằng cách tìm trên tập học giá trị D' nhỏ nhất làm giảm tiêu chí này. Đối với VLAD, vectors (k=16, D=2048) được tạo ra từ các mô tả SIFT, chúng ta thu được các

sai số trung bình sau cho phép chiếu, lượng tử hóa và tổng sai số bình phương trung bình:

D'	$e_p(D')$	$e_q(D')$	$e(D')$
32	0.0632	0.0164	0.0796
48	0.0508	0.0248	0.0757
64	0.0434	0.0321	**0.0755**
80	0.0386	0.0458	0.0844

Figure 14

- The impact of dimensionality reduction and indexation (Tác động của việc giảm kích thước và lập chỉ mục):
  Chúng ta có thể trình bày VLAD đã được chiếu và lượng tử hóa dưới dạng này, vì cả phép chiếu PCA và ADC đều cung cấp một cách để tái tạo lại vector đã chiếu/lượng tử hóa. Hình sau minh họa cách mà mỗi phép toán này ảnh hưởng đến biểu diễn. Người ta có thể thấy rằng vector chỉ bị thay đổi một cách nhỏ, ngay cả cho một biểu diễn nhỏ gọn với B = 16 byte.

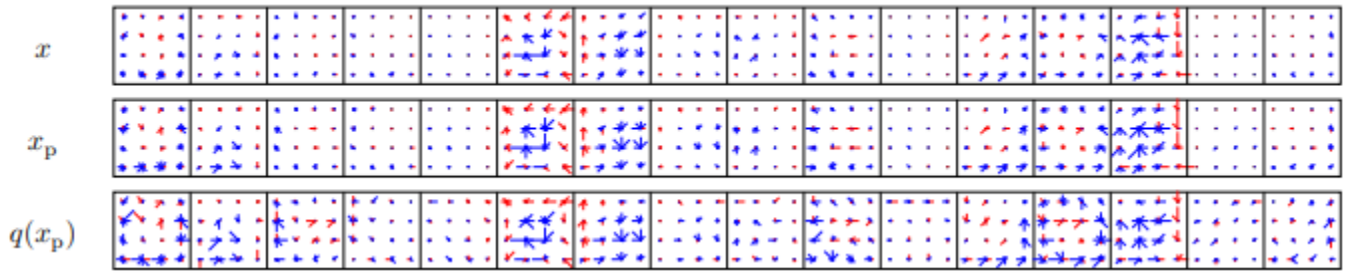


Figure 15: Tác động của các bước mã hóa lên mô tả. Trên cùng: vector VLAD x cho $k = 16$ ($D = 2048$). Giữa: vector x_p bị thay đổi sau phép chiếu vào không gian con PCA (D' = 128). Dưới cùng: vector $q(x_p)$ sau khi được chỉ mục bởi ADC 16×8 (mã 16 byte).

(b) Lập luận được trình bày:

- Bài báo so sánh 3 phương pháp BOF, Fisher kernel và VLAD. Đưa ra quy trình riêng lẻ của chúng, từ đó so sánh.
- Giới thiệu về VLAD đồng thời đưa ra bằng chứng bằng thực nghiệm chứng mứng độ hiệu quả của VLAD ngay cả đối với lượng từ ngữ hình ảnh nhỏ.
- Bằng việc xem xét phương pháp đúng nhất hiện tại "nearest neighbor search method of Product quantization", bài báo đưa ra tầm quan trọng của việc tối ưu hóa thuật toán và chiều qua việc đo lường lỗi của phương pháp Euclidean.

(c) Phân tích dữ liệu và kết luận:

- Bài báo lấy dữ liệu từ 3 bộ dữ liệu:
  - The INRIA Holidays dataset: gồm 1491 hình ảnh về kì nghỉ, 500 trong số đó được sử dụng làm truy vấn. Độ chính xác được đo bằng mean Average Precision (mAP).
  - The University of Kentucky Benchmark (UKB): gồm hình ảnh của 2550 đối tượng, mỗi đối tượng đc thể hiện bằng 4 hình ảnh.

  - Ngoài ra, để đánh giá hoạt động của phương pháp trên quy mô lớn, các tác giả đã tải xuống 10 triệu hình ảnh từ Flickr. Tập dữ liệu INRIA Holidays được kết hợp với tập hợp này để cung cấp ground truth matches.
- Vector biểu diễn hình ảnh:
  - Bài báo so sánh 3 phương pháp chính được nêu ở trên. BOF và VLAD được tham số hóa qua tham số k và Fisher kernel tham số bằng trung bình của vector(u).
  - Khi tối ưu hóa chiều của vector ta thấy VLAD hiệu quả tương đường so với Fisher kernel trong bộ dữ liệu của cả Holiday lẫn UKB. Trong khi đó VLAD cũng hoạt động hiệu quả hơn hẳn BOF.
- So sánh với "the state of art":
  - Từ những dữ liệu được cho về dung lượng tiêu tốn và độ chính xác khi sử dụng bộ dữ liệu Holiday và UKB. Bài báo cho thấy phương pháp của họ đưa ra đạt được chất lượng tương đương với các phương pháp Packing bag-of- features với mức độ lớn về bộ nhớ ít hơn. Tương tự, với việc sử dụng cùng một lượng bộ nhớ, phương pháp này cũng chính xác hơn đáng kể.
- Thí nghiệm trên quy mô lớn:
  - Từ dữ liệu có thể thấy phương pháp bài báo đưa ra đưa ra kết quả tốt hơn rất nhiều so với Packing bag-of-features.

1.3. Hạn chế hoặc mở rộng đối với các ý tưởng trong bài viết:

- Khả năng chống nhiễu: Bài báo tập trung vào việc tổng hợp các mô tả cục bộ nhưng chưa đề cập đến khả năng chống nhiễu đối với các biến thể trong hình ảnh, chẳng hạn như thay đổi góc nhìn, điều kiện ánh sáng, hoặc che khuất. Nghiên cứu tiếp theo có thể khám phá các kỹ thuật để cải thiện khả năng chống nhiễu của phương pháp được đề xuất.
- Tính tổng quát đối với các lĩnh vực khác: Bài báo chủ yếu tập trung vào tìm kiếm hình ảnh. Thú vị nếu khám phá tính áp dụng và hiệu quả của phương pháp đề xuất trong các lĩnh vực khác như tìm kiếm video hoặc truy xuất đối tượng 3D.
- Hiệu suất sử dụng bộ nhớ: Mặc dù bài báo đã giải quyết ràng buộc sử dụng bộ nhớ, có thể tiếp tục nghiên cứu các tối ưu hóa khác để giảm bộ nhớ cần thiết cho việc lập chỉ mục và tìm kiếm cơ sở dữ liệu hình ảnh quy mô lớn.

1.4. Ý kiến về bài báo:
Bài báo này đề xuất một phương pháp tổng hợp đặc trưng hình ảnh cục bộ hiệu quả và tối ưu hóa đồng thời việc giảm số chiều và thuật toán lập chỉ mục. Các ý tưởng trong bài viết được trình bày một cách rõ ràng và logic, và kết quả đánh giá cung cấp bằng chứng về hiệu quả của phương pháp đề xuất:

- Ý tưởng tổng hợp các mô tả địa phương (local descriptors) vào một biểu diễn hình ảnh gọn gàng là một đề xuất đáng chú ý. Phương pháp VLAD (vector of locally aggregated descriptors) đề xuất trong bài báo cung cấp một cách tiếp cận hiệu quả để biểu diễn hình ảnh và giảm kích thước của vector đại diện.
- Tác động tiềm tàng của bài báo này là cải thiện hiệu suất tìm kiếm hình ảnh trên quy mô lớn. Bằng cách kết hợp việc giảm chiều dữ liệu và thuật toán lập chỉ mục, phương pháp đề xuất đã đạt được độ chính xác tìm kiếm tương đương với phương pháp bag-of-features (BOF) với một biểu diễn hình ảnh chỉ chiếm 20 byte. Điều này giúp giảm bớt thời gian tìm kiếm và tăng hiệu suất của hệ thống.
- Ý tưởng kết hợp các phương pháp như BOF, Fisher kernel và phân cụm k-means để tạo ra biểu diễn hình ảnh mới cũng là một đóng góp quan trọng. Điều này cho phép tận dụng các đặc trưng cục bộ mạnh mẽ và tăng cường khả năng phân loại của hệ thống.

2 Chi tiết về phương hướng, mục tiêu mong muốn và kế hoạch thực hiện các mục tiêu mong muốn

- Phương hướng: Ứng dụng phương pháp VLAD (vector of locally aggregated descriptors) để tạo ra một biểu diễn vector cho hình ảnh qua đó có thể tìm kiếm hình ảnh trên quy mô lớn. Bằng cách tạo ra một biểu diễn vector rút gọn cho mỗi hình ảnh, ta có thể tìm kiếm các hình ảnh tương tự trong một cơ sở dữ liệu hình ảnh rất lớn một cách hiệu quả và tiết kiệm bộ nhớ.

- Mục tiêu mong muốn:
  Cân nhắc đồng thời ba ràng buộc: độ chính xác của tìm kiếm, hiệu suất của quá trình tìm kiếm và sử dụng bộ nhớ để lưu trữ biểu diễn hình ảnh

  – Độ chính xác của tìm kiếm: Cung cấp độ chính xác tìm kiếm tốt hơn so với biểu diễn BOF truyền thống

  – Tính hiệu quả: Bằng cách tổng hợp các bộ mô tả cục bộ thành một biểu diễn vectơ nhỏ gọn để có thể tối ưu hóa hiệu quả của quá trình tìm kiếm. Tìm kiếm tập dữ liệu gồm 10 triệu hình ảnh mất khoảng 50 ms.

  – Sử dụng bộ nhớ: Giảm mức sử dụng bộ nhớ trong khi vẫn duy trì độ chính xác của tìm kiếm. Cách biểu diễn này có thể chứa khoảng 20 byte cho mỗi hình ảnh, giúp cải tiến đáng kể so với các phương pháp hiện có.

- Kế hoạch thực hiện:

  B1: Trích xuất đặc trưng cục bộ SIFT từ hình ảnh.

```python
# Hàm để trích xuất đặc trưng cục bộ SIFT từ hình ảnh
def extract_sift_features(image_path):
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    sift = cv2.SIFT_create()
    key_points, descriptors = sift.detectAndCompute(image, None)
    return key_points, descriptors
```

```python
# Trích xuất đặc trưng cục bộ từ tất cả các hình ảnh
all_features = []
for path in image_paths:
    key_points, descriptors = extract_sift_features(path)
    if descriptors is not None:
        all_features.extend(descriptors)

# Chuyển danh sách đặc trưng thành một mảng NumPy
all_features = np.array(all_features)
```

Figure 16: Bước 1

  B2: Tiến hành xây dựng một bộ từ điển hình ảnh bằng phương pháp "K-means Clustering". Bộ từ điển này sẽ gồm k "visual words" đại diện cho các đặc trưng cục bộ trong hình ảnh.

```
# Sử dụng K-Means để tạo codebook
kmeans = KMeans(n_clusters=k, random_state=0)
kmeans.fit(all_features)

# Bộ từ điển hình ảnh (codebook) là các điểm trung tâm của các cụm (clusters)
codebook = kmeans.cluster_centers_
```

Figure 17: Bước 2

B3: Áp dụng thuật toán VLAD để tạo ra các vector mô tả hình ảnh. Đối với mỗi hình ảnh, các đặc trưng cục bộ sẽ được gán cho visual word gần nhất và sau đó, sự khác biệt giữa các đặc trưng và visual word tương ứng sẽ được tính toán và tích lũy.

B4: Tiến hành chuẩn hóa vector VLAD bằng cách tính L2-norm (Euclidean norm) để đảm bảo tính nhất quán giữa các vector mô tả hình ảnh.

```
# Hàm để tính VLAD vector cho một hình ảnh
def compute_vlad_vector(image_path, codebook):
    key_points, descriptors = extract_sift_features(image_path)
    vlad_vector = np.zeros((k, descriptors.shape[1]), dtype=np.float32)

    # Gán các đặc trưng cục bộ cho visual words gần nhất
    labels = kmeans.predict(descriptors)

    for i in range(len(key_points)):
        vlad_vector[labels[i]] += descriptors[i] - codebook[labels[i]]

    # Chuẩn hóa VLAD vector bằng cách tính L2-norm (Euclidean norm)
    vlad_vector = vlad_vector.flatten()
    vlad_vector /= np.linalg.norm(vlad_vector)

    return vlad_vector

# Tính VLAD vectors cho tất cả hình ảnh trong danh sách
vlad_vectors = []
for path in image_paths:
    vlad_vector = compute_vlad_vector(path, codebook)
    vlad_vectors.append(vlad_vector)

# Chuyển danh sách VLAD vectors thành mảng NumPy
vlad_vectors = np.array(vlad_vectors)
```

Figure 18: Bước 3 và 4

B5: Áp dụng quá trình giảm chiều dữ liệu (dimensionality reduction) bằng phương pháp Principal Component Analysis (PCA) để giảm số chiều của vector VLAD. Quá trình này giúp giảm bớt không gian lưu trữ cần thiết và cải thiện tốc độ tìm kiếm.

```
# Áp dụng PCA để giảm chiều dữ liệu
pca = PCA(n_components=n_components)
vlad_vectors_reduced = pca.fit_transform(vlad_vectors)
```

Figure 19: Bước 5

B6: Xây dựng thuật toán chỉ mục (indexing algorithm) để lưu trữ các vector mô tả hình ảnh. Cách tiếp cận này nhằm tối ưu hóa việc so sánh và truy vấn các vector trong quá trình tìm kiếm hình ảnh.

```
# Hàm để tìm và hiển thị hình ảnh tương tự
def search_similar_images(query_vector, kdtree, vlad_vectors_reduced,
                          num_results=1):
    _, indices = kdtree.query(query_vector, num_results)

    if num_results == 1:
        idx = int(indices[0])  # Truy cập chỉ mục đầu tiên
        image_path = image_paths[idx]
        display_image_with_vector(image_path,
                        vlad_vectors_reduced[idx], "Similar Image")
    else:
        for i in range(num_results):
            idx = int(indices[i][0])
            image_path = image_paths[idx]
            display_image_with_vector(image_path,
                vlad_vectors_reduced[idx],f"Similar Image {i + 1}")

# Tạo vector mô tả cho hình ảnh đã có
query_image_path = "/content/eiffel2.jpg"
query_vlad_vector = compute_vlad_vector(query_image_path, codebook)
query_vlad_vector = query_vlad_vector.reshape(1, -1)
query_vlad_vector_reduced = pca.transform(query_vlad_vector)

# Hiển thị vector mô tả của hình ảnh đã có
display_image_with_vector(query_image_path, query_vlad_vector_reduced,
                          "Query Image")

# Tìm và hiển thị hình ảnh tương tự
search_similar_images(query_vlad_vector_reduced, kdtree, vlad_vectors_reduced,
                      num_results=1)
```

Figure 20: Bước 6

B7: Tiến hành đánh giá và so sánh hiệu suất của hệ thống với các phương pháp khác. Đánh giá được thực hiện bằng cách so sánh độ chính xác tìm kiếm, hiệu suất và sử dụng bộ nhớ của hệ thống VLAD với các phương pháp truyền thống khác như Bag-of-Features (BOF).

```python
def compute_accuracy(kdtree, vlad_vectors_reduced, test_paths, codebook, pca):
    correct_matches = 0
    total_queries = len(test_paths)

    for query_image_path in test_paths:
        query_vlad_vector = compute_vlad_vector(query_image_path, codebook)
        query_vlad_vector = query_vlad_vector.reshape(1, -1)
        query_vlad_vector_reduced = pca.transform(query_vlad_vector)

        target = search_similar_images_path(query_vlad_vector_reduced, train_kdtree, train_vlad_vectors_reduced, num_results=1)

        if os.path.dirname(query_image_path) == os.path.dirname(target):
                correct_matches += 1

    accuracy = correct_matches / total_queries

    return accuracy

accuracy = compute_accuracy(train_kdtree, test_vlad_vectors_reduced, test_paths, codebook, pca)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

Figure 21

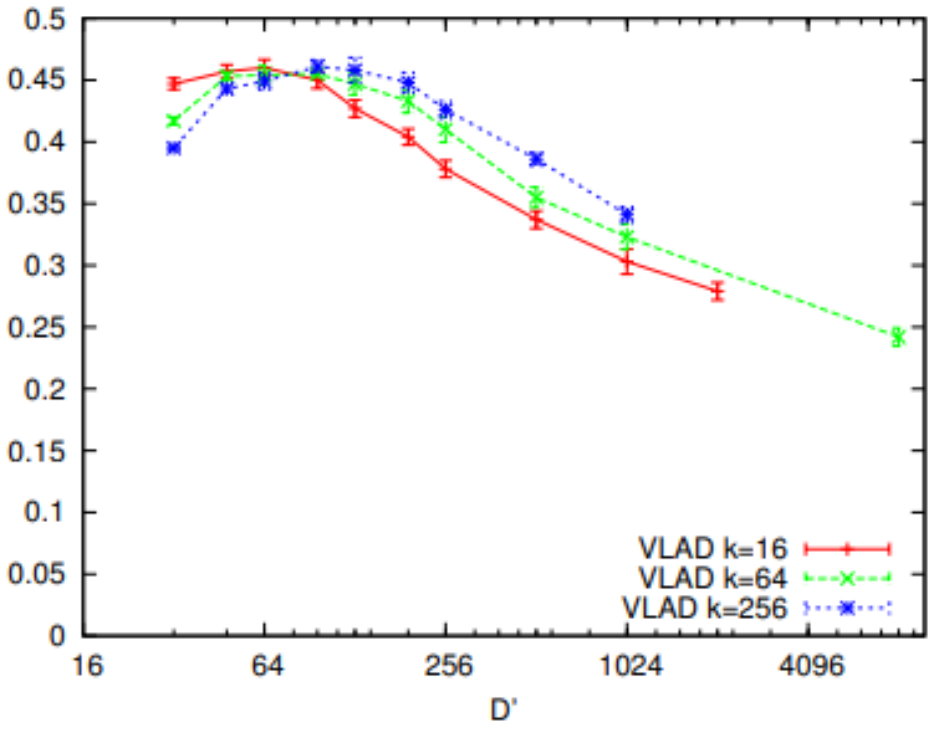


Figure 22: Độ chính xác tìm kiếm trên tập dữ liệu Holidays liên quan đến việc giảm chiều D' khác nhau với ADC 16×8

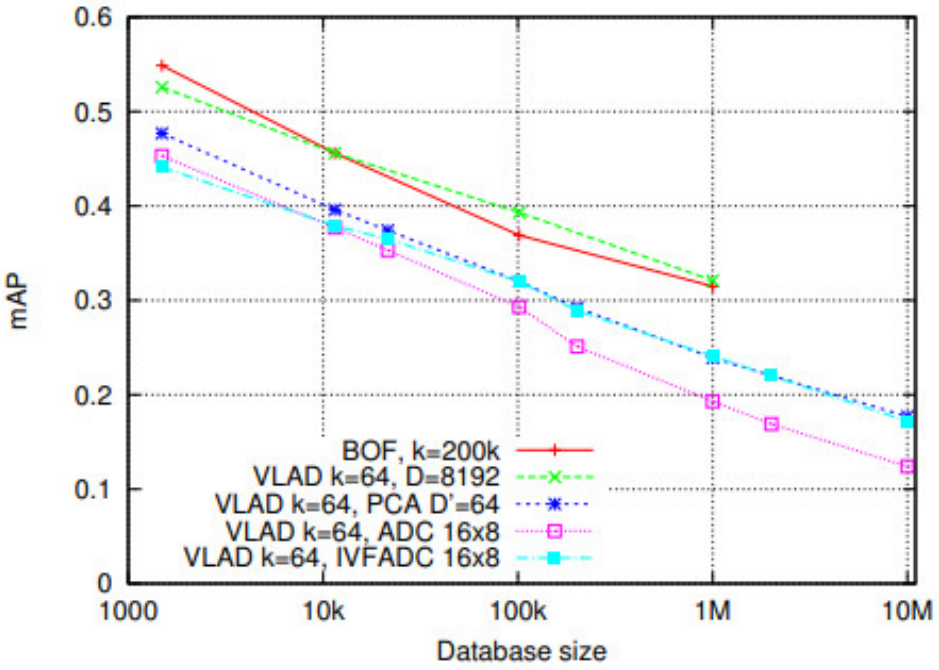


Figure 23: Độ chính xác tìm kiếm trên quy mô lớn

- Kết quả thực hiện:
  - Dataset: The INRIA Holidays dataset (gồm 1491 hình ảnh về kì nghỉ, 500 trong số đó được sử dụng làm truy vấn).
  - Một số kết quả thu được:

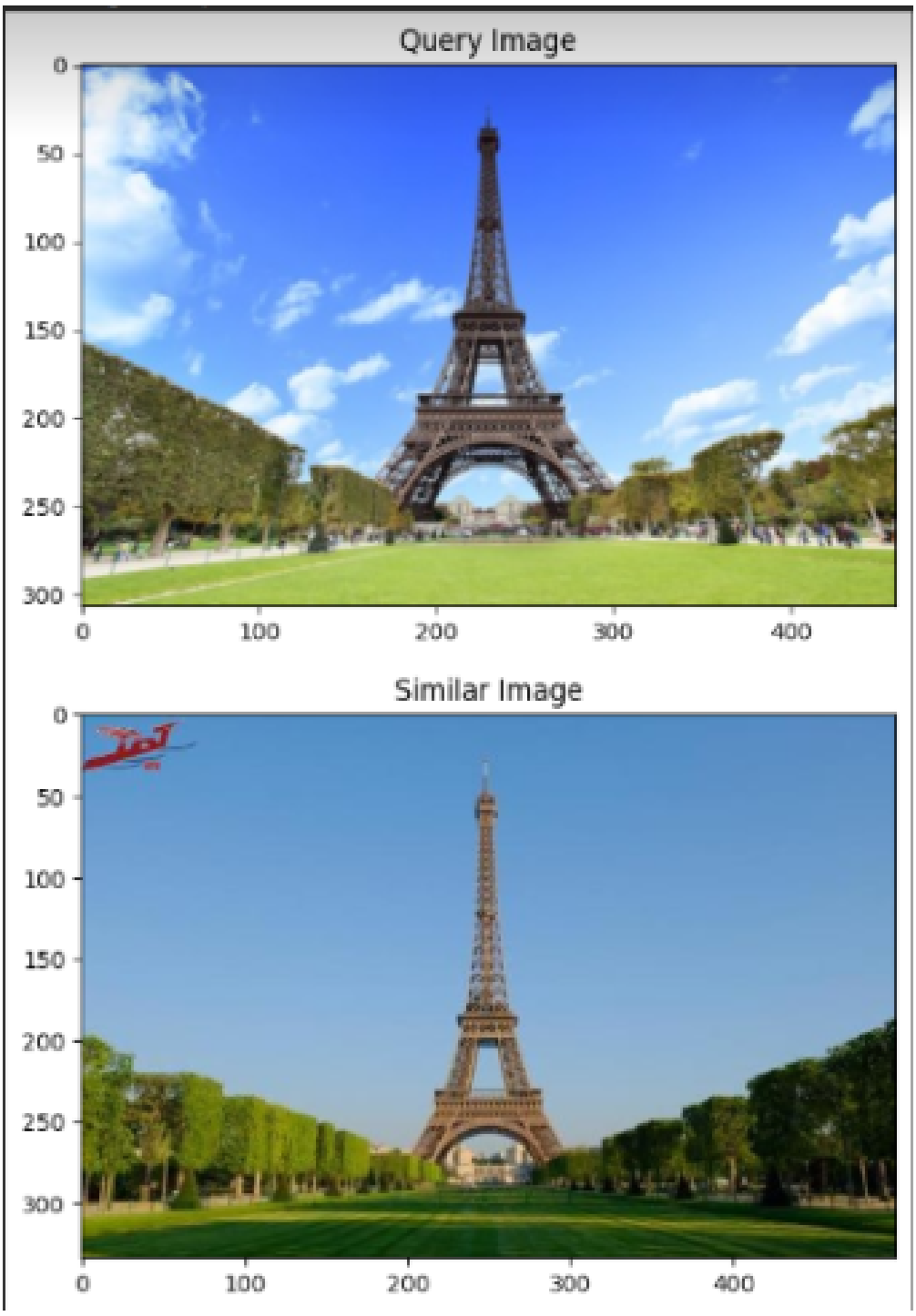

Query Image
0
50
100
150
200
250
300
0
100
200
300
400
Similar Image
0
50
100
150
200
250
300
0
100
200
300
400

Figure 24

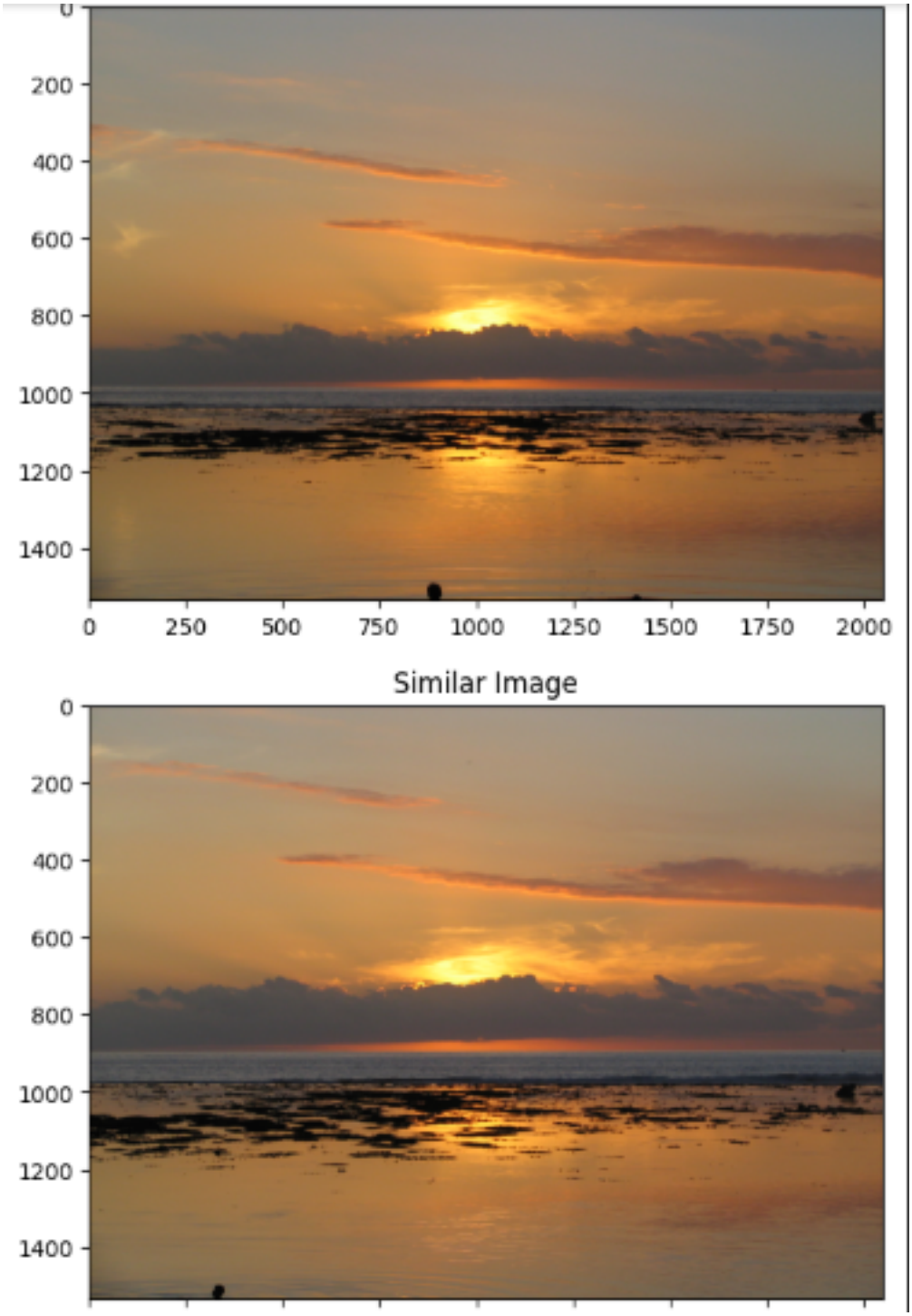

0
200
400
600
800
1000
1200
1400
0
250
500
750
1000
1250
1500
1750
2000
Similar Image
0
200
400
600
800
1000
1200
1400

Figure 25

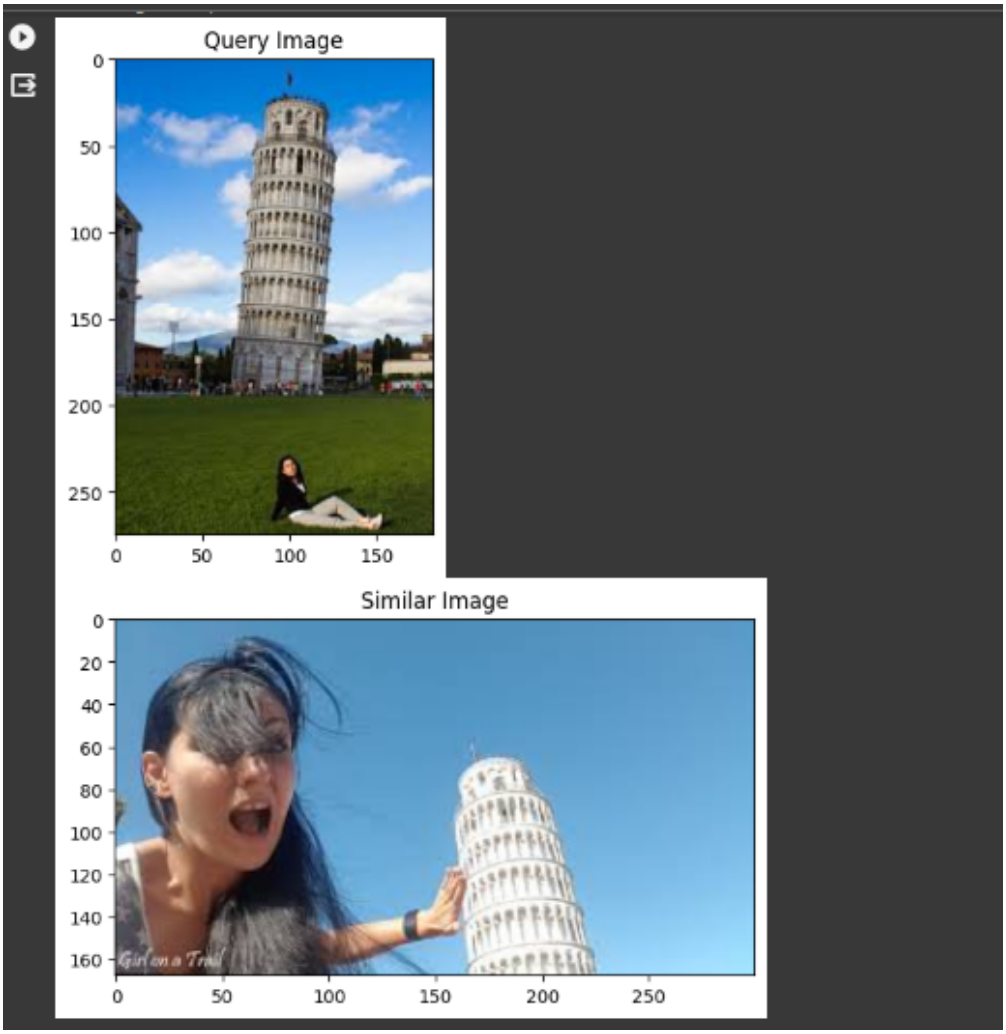


Figure 26

3 Thông tin về từng thành viên trong nhóm và nhiệm vụ được phân công

Người thực hiện	Mã sinh viên	Nhiệm vụ được phân công
Nguyễn Văn An	21021452	Nội dung 1.4, 2 và cài đặt thuật toán
Đỗ Minh Cường	21021459	Nội dung 1.1, 1.3 và thuyết trình
Hà Mạnh Dũng	21021465	Nội dung 1.2, làm slide và so sánh thuật toán

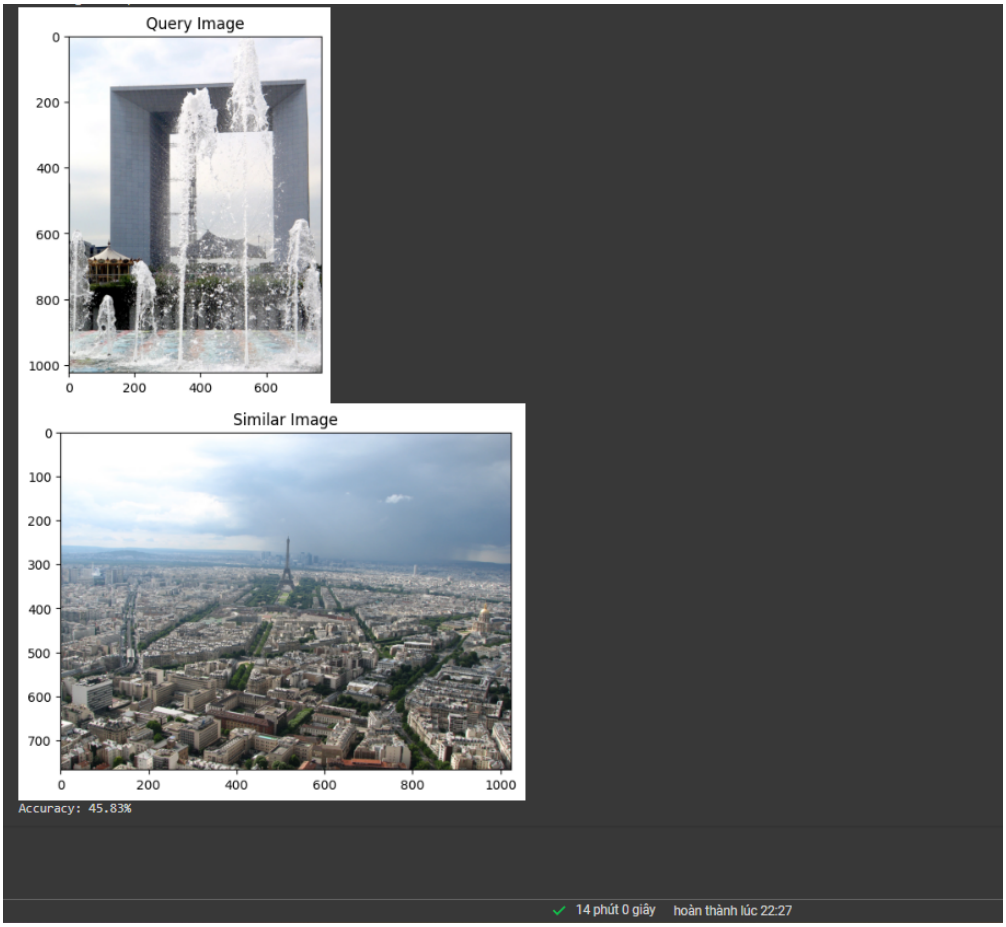

Query Image
Similar Image
Accuracy: 45.83%
14 phút 0 giây
hoàn thành lúc 22:27

Figure 27

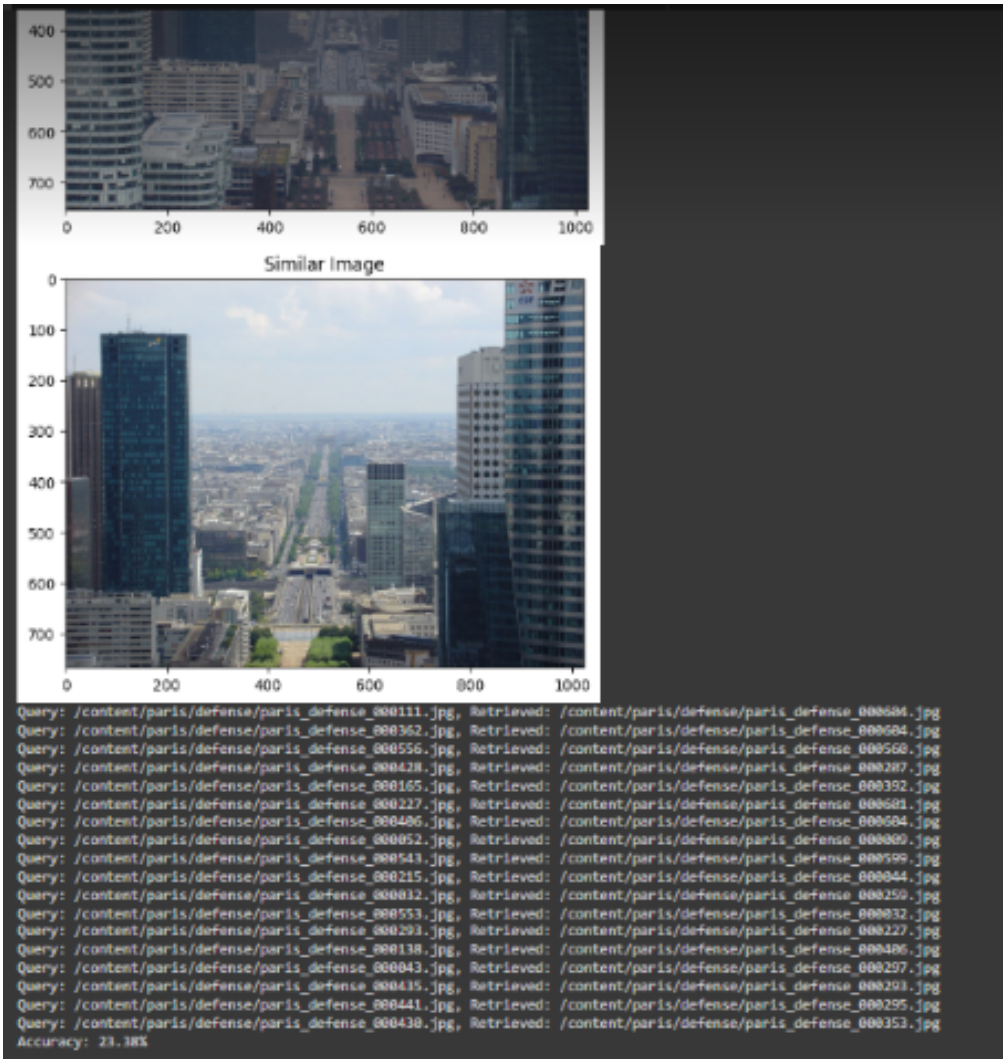
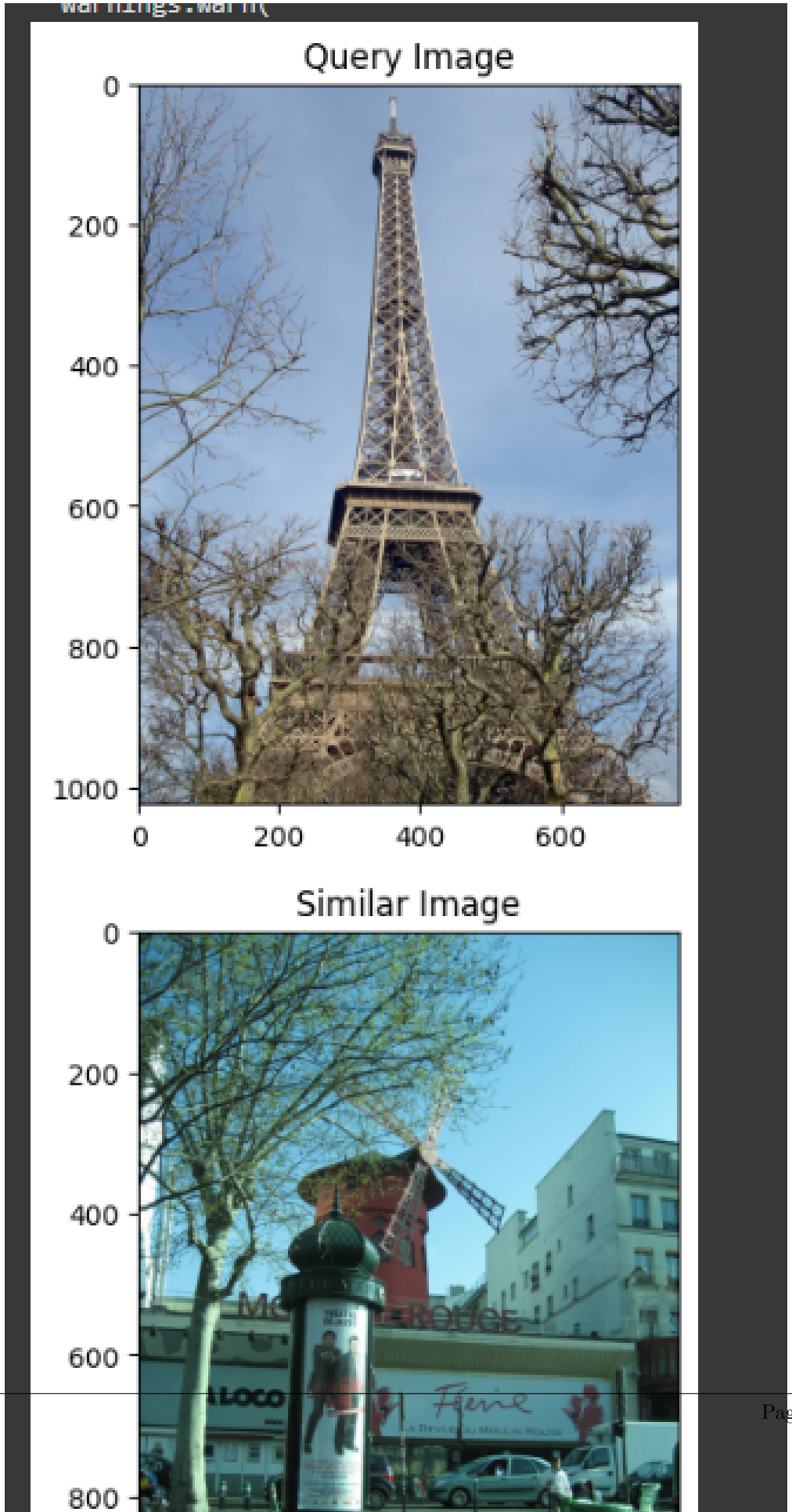

Similar Image
Query: /content/paris/defense/paris_defense_000111.jpg, Retrieved: /content/paris/defense/paris_defense_000604.jpg
Query: /content/paris/defense/paris_defense_000362.jpg, Retrieved: /content/paris/defense/paris_defense_000604.jpg
Query: /content/paris/defense/paris_defense_000556.jpg, Retrieved: /content/paris/defense/paris_defense_000560.jpg
Query: /content/paris/defense/paris_defense_000428.jpg, Retrieved: /content/paris/defense/paris_defense_000207.jpg
Query: /content/paris/defense/paris_defense_000165.jpg, Retrieved: /content/paris/defense/paris_defense_000392.jpg
Query: /content/paris/defense/paris_defense_000227.jpg, Retrieved: /content/paris/defense/paris_defense_000601.jpg
Query: /content/paris/defense/paris_defense_000406.jpg, Retrieved: /content/paris/defense/paris_defense_000604.jpg
Query: /content/paris/defense/paris_defense_000052.jpg, Retrieved: /content/paris/defense/paris_defense_000009.jpg
Query: /content/paris/defense/paris_defense_000543.jpg, Retrieved: /content/paris/defense/paris_defense_000599.jpg
Query: /content/paris/defense/paris_defense_000215.jpg, Retrieved: /content/paris/defense/paris_defense_000044.jpg
Query: /content/paris/defense/paris_defense_000032.jpg, Retrieved: /content/paris/defense/paris_defense_000250.jpg
Query: /content/paris/defense/paris_defense_000553.jpg, Retrieved: /content/paris/defense/paris_defense_000032.jpg
Query: /content/paris/defense/paris_defense_000293.jpg, Retrieved: /content/paris/defense/paris_defense_000227.jpg
Query: /content/paris/defense/paris_defense_000138.jpg, Retrieved: /content/paris/defense/paris_defense_000406.jpg
Query: /content/paris/defense/paris_defense_000043.jpg, Retrieved: /content/paris/defense/paris_defense_000297.jpg
Query: /content/paris/defense/paris_defense_000435.jpg, Retrieved: /content/paris/defense/paris_defense_000293.jpg
Query: /content/paris/defense/paris_defense_000441.jpg, Retrieved: /content/paris/defense/paris_defense_000295.jpg
Query: /content/paris/defense/paris_defense_000430.jpg, Retrieved: /content/paris/defense/paris_defense_000353.jpg
Accuracy: 23.38%

Figure 28


Query Image
0
200
400
600
800
1000
0
200
400
600
Similar Image
0
200
400
600
800