

Trường Đại học Bách Khoa Hà Nội



Bài tập lớn môn học
Đề tài 1: Đếm đối tượng
Đề tài 2: Nhận dạng đối tượng sử dụng Bag of Words (BoW)

Học phần: Thị giác máy tính - IT5409

Giảng viên hướng dẫn: TS. Nguyễn Thị Oanh

Nhóm 14:

Nguyễn Văn Lương	20173249
Bùi Phó Bền	20172968
Nguyễn Hữu Anh Việt	20173465
Vũ Nam Sơn	20173348
Phùng Ngọc Minh	20173257

Hà Nội, tháng 6 năm 2021

Mở đầu	4
Đề tài 1. Đếm đối tượng	5
1. Bài toán	5
2. Dữ liệu	5
3. Phương pháp	5
3.1 Trường hợp đối với ảnh đồ dùng	5
3.1.1 Khử nhiễu muối tiêu	5
3.1.2 Tìm cạnh	5
3.1.3 Lắp đầy các pixel trong cạnh về màu trắng	6
3.1.4 Dùng phép Open Morphological để loại nhiễu	7
3.1.5 Tìm contours	7
3.2 Trường hợp đối với ảnh gạo	8
3.2.1 Phương pháp chung	8
3.2.2 Phương pháp riêng đối với ảnh không thiếu cân bằng sáng	9
3.2.3 Phương pháp riêng đối với ảnh thiếu cân bằng sáng	10
4. Đánh giá và kết luận	10
4.1 Phương pháp đánh giá	10
4.2 Trường hợp đối với ảnh đồ dùng	10
4.3 Trường hợp đối với ảnh gạo	11
Đề tài 2. Nhận dạng đối tượng sử dụng Bag of Words (BoW)	15
1. Bài toán	15
1.1 Phát hiện các keypoints and descriptors	15
1.2 Phân cụm các descriptors	15
1.3 Xây dựng tập histogram	16
1.4 Phân loại ảnh	16
2. Dữ liệu	16
2.1 COIL100	16
2.2 CIFAR10	17
2.3 Một vài bộ dữ liệu thu thập từ Internet khác	18
3. Phương pháp	18
3.1 SIFT(Scale-Invariant Feature Transform)	19
3.2 BRISK	19
3.3 ORB	19
3.4 SVM	19
4. Kết quả thực nghiệm	19
4.1 Bộ COIL	19
4.1.1 ORB	19
4.1.2 BRISK	20
4.1.3 SIFT	20
4.2 Bộ 7 Classes	20
4.2.1 ORB	20
4.2.2 BRISK	21
4.2.3 SIFT	21

4.3 Bộ CIFAR	21
5. Đánh giá và kết luận	21
Tài liệu tham khảo	23

Mở đầu

Hiện nay, với sự phát triển nhanh chóng của cuộc Cách mạng công nghiệp 4.0, Trí tuệ nhân tạo đang được áp dụng vào mọi lĩnh vực trong cuộc sống. Trong đó Xử lý ảnh là một trong những lĩnh vực được dành nhiều sự quan tâm và có những ứng dụng thực tiễn hiệu quả nhất. Bên cạnh sự phát triển của Machine Learning, giúp giải quyết rất hiệu quả những vấn đề của Xử lý ảnh, tuy nhiên đòi hỏi chi phí tính toán khá lớn. Các phương pháp truyền thống tuy độ chính xác không cao bằng nhưng lại có thể hoạt động trên các thiết bị yếu nhất, và trong một số bài toán cụ thể, độ chính xác cũng đạt mức yêu cầu. Tìm hiểu về lĩnh vực này, nhóm em đã lựa chọn đề tài “Đếm đối tượng” và “Nhận dạng đối tượng sử dụng Bag of Words” để áp dụng những lý thuyết được học vào bài toán thực tế.

Đề tài 1. Đếm đối tượng

1. Bài toán

Đếm số lượng đối tượng có sẵn trong ảnh.

2. Dữ liệu

Để thử nghiệm và đánh giá giải pháp của mình, nhóm em sử dụng các ảnh mà cô đã đưa ra và một số ảnh tìm kiếm trên mạng Internet.

3. Phương pháp

Sau khi thử nghiệm các phương pháp khác nhau, nhóm em quyết định chia bài toán thành 2 nhóm riêng cho đồ dùng trên nền tương đối đồng nhất và nhóm hạt gạo.

Ngôn ngữ lập trình nhóm em sử dụng là python, với thư viện numpy hỗ trợ thao tác với mảng và thư viện OpenCV hỗ trợ các hàm xử lý ảnh.

3.1 Trường hợp đối với ảnh đồ dùng

3.1.1 Khử nhiễu muối tiêu

Với ảnh đầu vào, nhóm em thực hiện khử nhiễu muối tiêu do phép khử này không gây ảnh hưởng đến ảnh trong trường hợp không có nhiễu.

Sử dụng hàm **medianBlur()** của OpenCV với kích thước filter là 3x3.

3.1.2 Tìm cạnh

Nhóm em sử dụng thuật toán **Canny()** của thư viện OpenCV để tìm cạnh của đối tượng.

Thuật toán này sử dụng kernel Sobel theo cả 2 chiều ngang và dọc để tính đạo hàm G và hướng tại từng pixel của ảnh. Sau đó kiểm tra tại từng pixel, giá trị đạo hàm đó có phải là cực đại cục bộ theo hướng đạo hàm đó hay không, nếu không thì sẽ loại bỏ. Cuối cùng sẽ sử dụng 2 tham số $minVal$ và $maxVal$ để lọc ra các pixel được coi là cạnh.

- Nếu $G < minVal$, pixel đó chắc chắn không phải cạnh
- Nếu $G > maxVal$, pixel đó chắc chắn là cạnh
- Nếu $minVal \leq G \leq maxVal$, xét xem pixel đó có là hàng xóm của pixel chắc chắn là cạnh không, nếu có thì được coi là cạnh.

Sau khi thử nghiệm trên các ảnh mẫu, nhóm em sử dụng giá trị $minVal = 20$ và $maxVal = 80$.

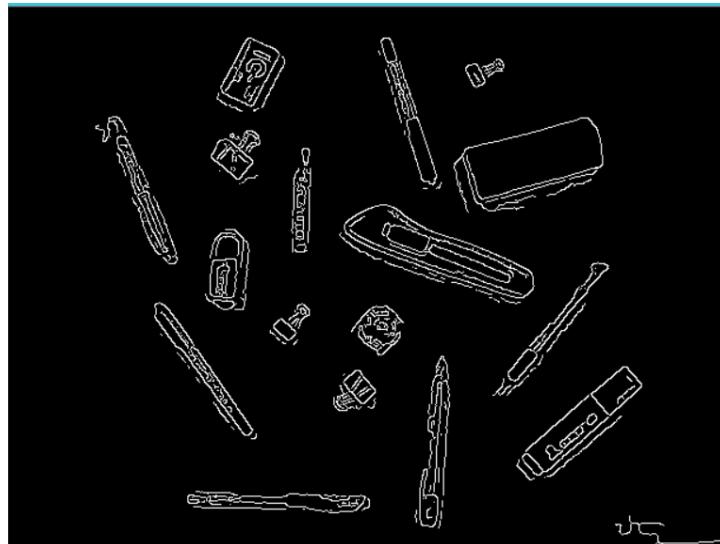
Tiếp theo nhóm em sử dụng phép Close Morphological để kết nối các cạnh bị đứt rời với nhau, bước này giúp liên kết các thành phần của đối tượng nếu bước tìm cạnh không tìm đủ các cạnh của đối tượng.

Kernel sử dụng là hình Ellipse 5x5:

[0 0 1 0 0]
[1 1 1 1 1]
[1 1 1 1 1]

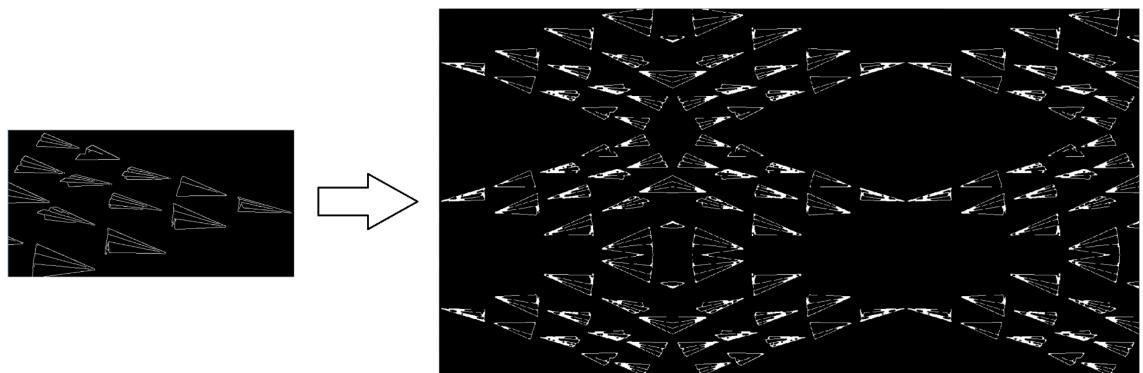
[1 1 1 1 1]
[0 0 1 0 0]

Ví dụ:



3.1.3 Lấp đầy các pixel trong cạnh về màu trắng

Để lấp đầy các pixel trong cạnh, trước tiên cần tìm các vùng cạnh liên thông bằng cách tìm contours trên mặt nạ cạnh. Tuy nhiên đối với các đối tượng ở rìa ảnh, cạnh có thể không bao hết đối tượng. Giải quyết vấn đề này, nhóm em đã thực hiện padding thêm vào ảnh với phương thức: lật ảnh theo chiều ngang, theo chiều dọc, theo cả 2 chiều và ghép lại với nhau thành 1 ảnh với kích thước 3×3 lần ảnh ban đầu.



Sau đó dùng thuật toán **findContours()** của OpenCV để tìm contours và sử dụng hàm **drawContours()** để tô tất cả các pixel trong vùng contours thành màu trắng.

Cắt vùng ảnh tương ứng ảnh ban đầu, thu được mặt nạ nhị phân với các vùng trắng là các vùng đại diện cho 1 đối tượng.

Ví dụ:



3.1.4 Dùng phép Open Morphological để loại nhiễu

Sử dụng phép Open Morphological với mặt nạ nhị phân đối tượng để loại bỏ nhiễu. Bước này việc lựa chọn kernel rất quan trọng vì nó có thể làm tách rời các phần của đối tượng nếu kernel quá lớn.

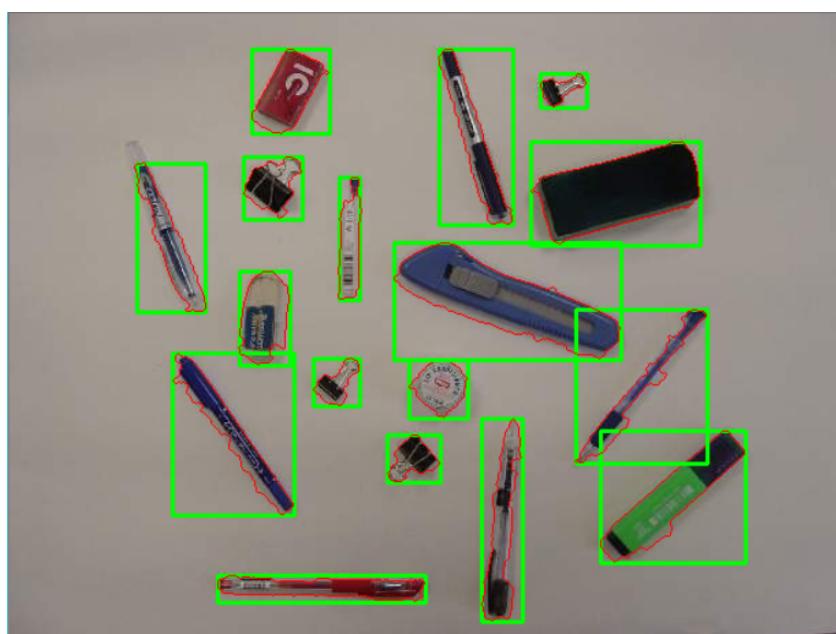
Kernel sử dụng là hình Ellipse 3x3:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

3.1.5 Tìm contours

Thực hiện tìm contours trên mặt nạ và đếm số lượng contours, đồng thời loại bỏ những contours có diện tích bao trong nhỏ hơn ngưỡng cho trước để không coi đó là đối tượng. Nhóm em lựa chọn ngưỡng này là **S/900** với S là diện tích của ảnh.

Ví dụ:



3.2 Trường hợp đối với ảnh gạo

Với bộ ảnh cô đưa ra, các ảnh hạt gạo bao gồm nhiều loại nhiễu: nhiễu muối tiêu, nhiễu cân bằng về độ sáng từng vùng, nhiều dạng tàn số. Vì vậy nhóm em thiết kế giải pháp chung và riêng cho từng loại để so sánh và đánh giá:

3.2.1 Phương pháp chung

- B1: Chuyển ảnh về thang màu xám
- B2: Khử nhiễu muối tiêu với bộ lọc trung vị filter kích thước 5×5
- B3: Tự động khử nhiễu dạng tàn số:
 - Chuyển ảnh từ miền không gian sang miền tàn số với phép biến đổi Fourier, sử dụng thư viện numpy.
 - Chuẩn hóa về miền giá trị nhỏ để dễ dàng thao tác bằng phép lũy log của ma trận miền tàn số.
 - Sử dụng một filter kích thước $k \times k$ tính hiệu của điểm đó với giá trị trung bình của các điểm xung quanh để lọc ra các ứng cử viên cho giá trị cần lọc trên miền tàn số, ví dụ $k = 3$

[0.125 0.125 0.125]

[0.125 -1 0.125]

[0.125 0.125 0.125]

$k = 2 * \text{size_filter} + 1$ là tham số truyền vào, mặc định = 2.

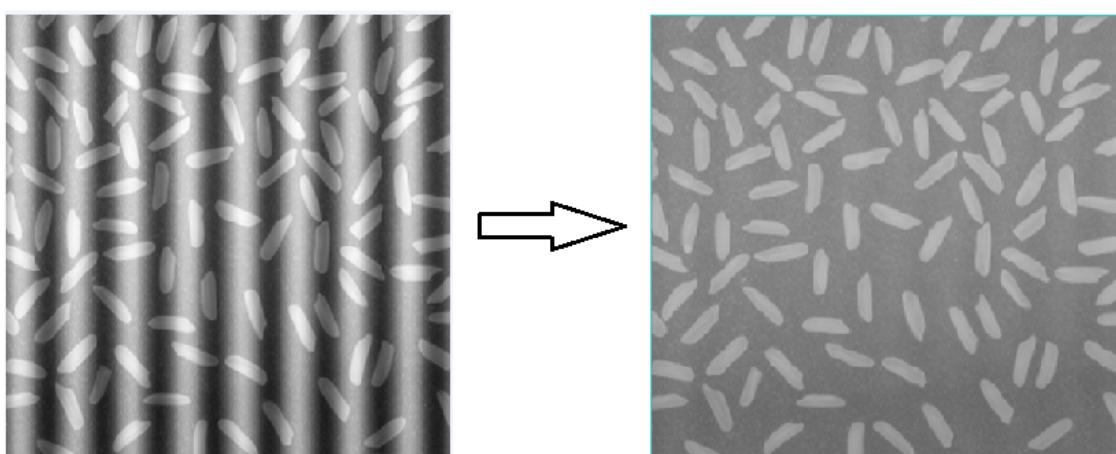
Riêng điểm tâm của ma trận, luôn là cực đại nên gán = 0 vì đây không phải nhiễu.,,

Kết quả thu được một mặt nạ.

Lọc các giá trị $> \text{diff_from_center_point}$ trên mặt nạ này, mặc định $\text{diff_from_center_point} = 50$, thu được một danh sách các điểm là ứng cử viên cho điểm gây nhiễu.

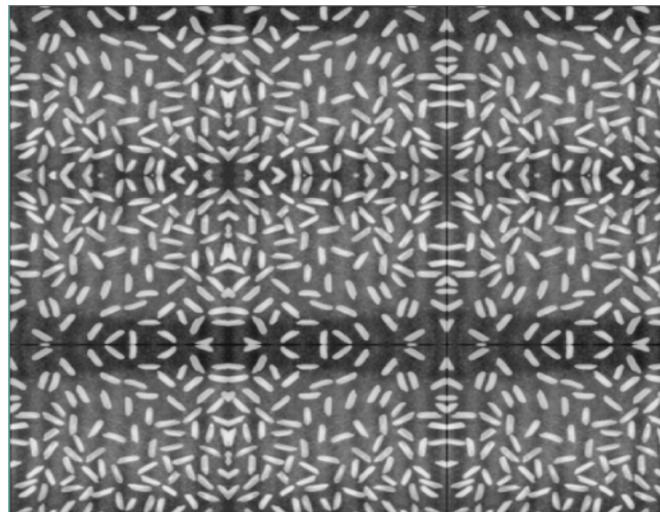
- Duyệt từng ứng cử viên, so sánh giá trị điểm đó với giá trị lớn nhất còn lại trong ma trận $k \times k$ (với $k = 2 * \text{size_filter} + 1$) các điểm xung quanh điểm đó. Nếu < 20 thì loại khỏi danh sách. Danh sách thu được là danh sách các điểm gây nhiễu cần loại bỏ.
- Gán các giá trị trên miền tàn số của các điểm gây nhiễu về 1.
- Đảo ngược từ miền tàn số về miền không gian, thu được ảnh sau khi lọc.

Ví dụ:



- B4: Làm mờ ảnh để giảm bớt nhiễu, sử dụng filter Gaussian kích thước 5x5.
- B5: Padding tương tự trường hợp ảnh đồ dùng để tránh lỗi khi xử lý các đối tượng ở rìa ảnh.
- B6: Cân bằng histogram cục bộ, chia ảnh thành các ô nhỏ và cân bằng histogram trên các ô đó. Tuy nhiên trong trường hợp ảnh có nhiều nhiễu thì cân bằng này sẽ tăng độ nhiễu lên rất lớn, thư viện OpenCV hỗ trợ giải quyết vấn đề này. Sử dụng lớp CLAHE của OpenCV với tham số $clipLimit = 5$, $tileGridSize = (w/50, h/50)$ với w, h là kích thước chiều ngang và dọc của ảnh. Tham số $clipLimit$ càng nhỏ (>0) thì mức độ nhiễu càng ít tuy nhiên hiệu quả cân bằng sáng lại càng thấp.

Ví dụ:



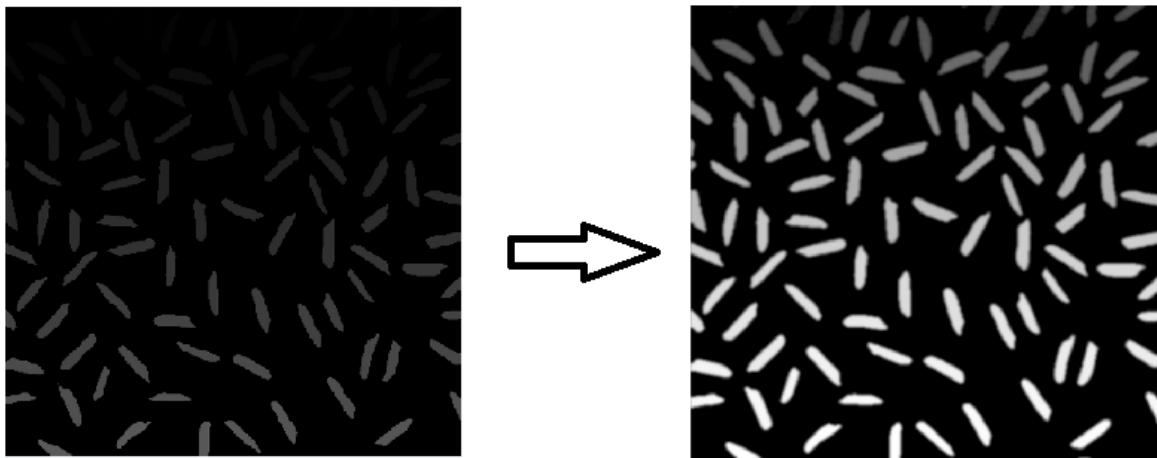
- B7: Nhị phân ảnh cục bộ:
- Sử dụng hàm `adaptiveThreshold()` của OpenCV để nhị phân ảnh. kernel sử dụng là Gaussian với kích thước 55x55, tham số C=-12 là giá trị để trừ đi sau khi tính ngưỡng từ kernel Gaussian để làm ngưỡng xét cuối cùng cho pixel đó. việc lấy giá trị -12 này giúp loại bỏ bớt các vùng có giá trị đều nhau. Việc này không làm mất đối tượng do kích thước kernel là 55x55 luôn lớn hơn khá nhiều kích thước của 1 hạt gạo trong ảnh. Cắt ảnh tương ứng vùng ảnh ban đầu để lấy mặt nạ tương ứng của ảnh ban đầu.
- B8: Thực hiện phép **Close Morphological** để làm đầy các đối tượng
- B9: Thực hiện phép **Erosion** để tách rời các đối tượng dính vào nhau trên mặt nạ nhị phân.
- B10: Tìm contours, loại những contours có diện tích vùng bao trong <0.08 lần kích thước contours lớn nhất (hạt gạo lớn nhất), đếm số contours chính là số lượng hạt gạo cần đếm.

3.2.2 Phương pháp riêng đối với ảnh không thiếu cân bằng sáng

- Thực hiện tương tự các bước từ B1-B5 như phương pháp chung.
- B6: Tương tự B6 ở phương pháp chung, thay C = -5 vì độ nhiễu không bị tăng do không sử dụng bước cân bằng histogram cục bộ nên không cần ngưỡng này quá nhỏ.
- Tiếp theo thực hiện tương tự các bước B8-B10 ở phương pháp chung, thay ngưỡng diện tích vùng bao = 5 vì không còn nhiều lớn không phải hạt gạo gây ra do cân bằng histogram cục bộ.

3.2.3 Phương pháp riêng đối với ảnh thiếu cân bằng sáng

- Thực hiện tương tự các bước B1-B4 của phương pháp chung
- B5: Cân bằng histogram trên toàn bộ ảnh thay vì cục bộ do ảnh rất ít nhiễu.



- B6: Nhị phân ảnh cục bộ:
- Tương tự B7 ở phương pháp chung, sử dụng kernel MEAN với kích thước 21x21 (không cần quá lớn do kích thước giao nhở hơn và nền rất ít nhiễu), C = -2.
- B7: Tương tự B10 ở phương pháp chung

4. Đánh giá và kết luận

4.1 Phương pháp đánh giá

Tiêu chí đo độ chính xác:

Acc: Phần trăm đối tượng phát hiện chính xác trên ảnh so với số đối tượng dự đoán đưa ra

$$Acc = \frac{\text{Số dự đoán chính xác}}{\text{Số dự đoán đưa ra}}$$

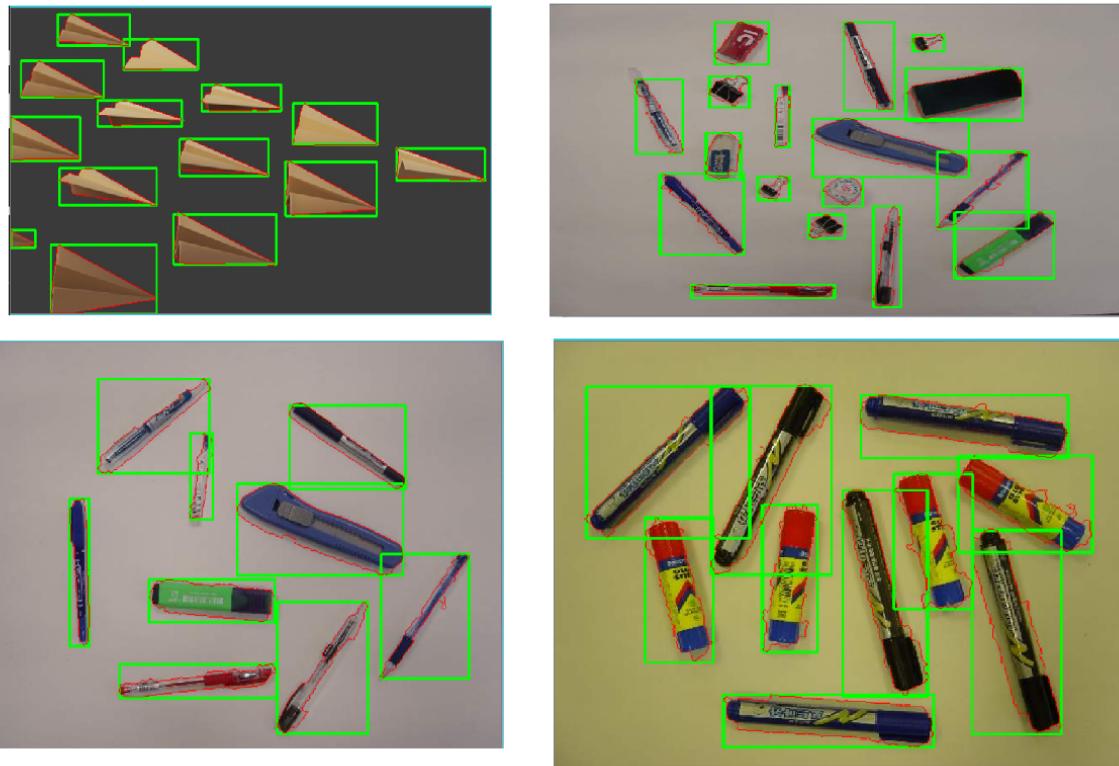
Pre: Phần trăm đối tượng đã phát hiện chính xác trên ảnh so với số đối tượng thực tế

$$Pre = \frac{\text{Số dự đoán chính xác}}{\text{Số đối tượng thực tế}}$$

4.2 Trường hợp đối với ảnh đồ dùng

4.2.1 Kết quả

Một số ví dụ:



Kết quả trên tập ảnh cô đưa ra:

Acc = 100%

Pre = 100%

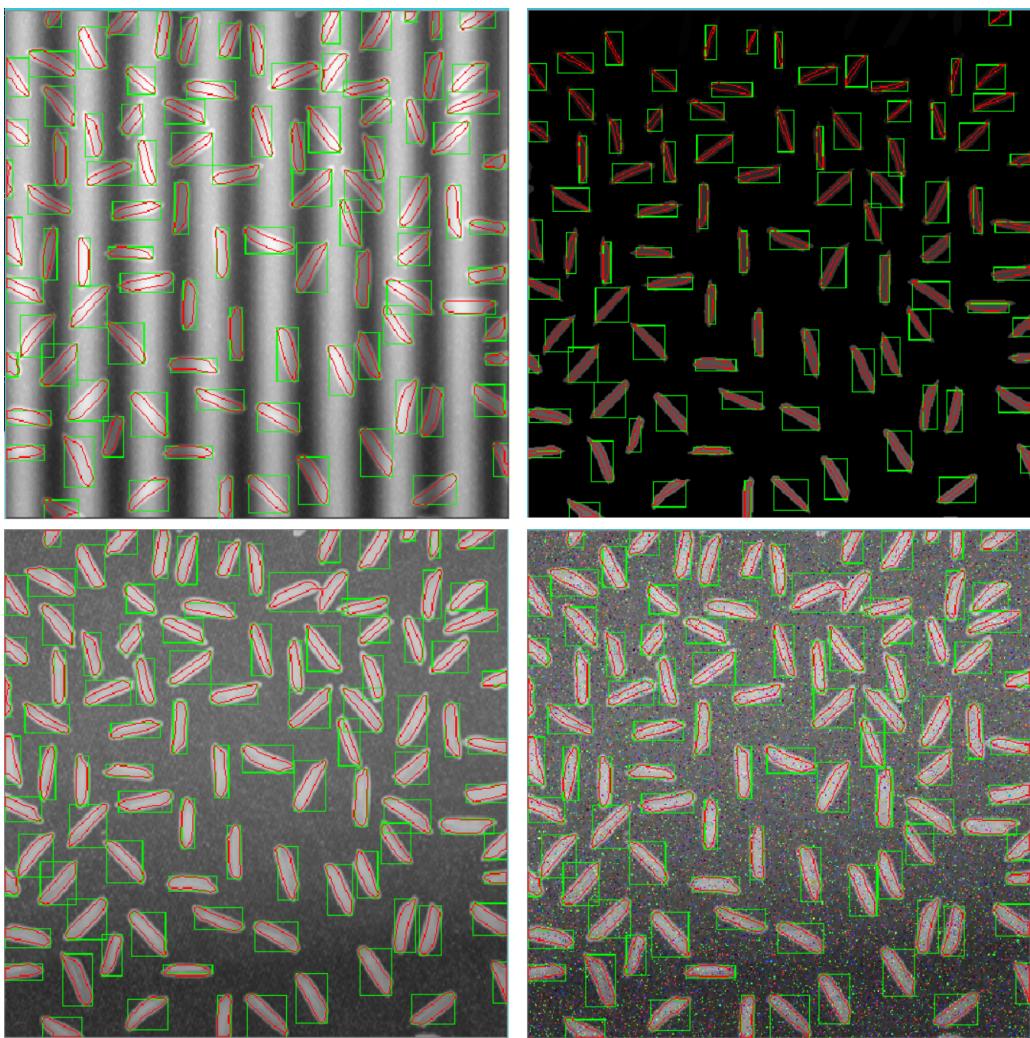
4.2.2 Đánh giá

Giải pháp hoạt động tốt với hình ảnh đã đưa ra, tuy nhiên có hạn chế và ưu điểm sau:

- **Ưu điểm:**
 - Hoạt động hoàn toàn tự động, người dùng không cần chọn các tham số.
 - Độ chính xác cao với các ảnh có nền tương đối đồng nhất, các đối tượng nằm tách rời nhau và màu sắc không quá giống với nền.
 - Thời gian xử lý nhanh ~ 0.02s
- **Hạn chế:**
 - Chưa giải quyết được vấn đề nền không đồng nhất. Điều này do thuật toán tìm cạnh không loại bỏ được các vùng không đồng nhất trên nền và sẽ coi đó là cạnh từ đó sẽ gây lỗi cho các bước xử lý sau.
 - Chưa xử lý được các trường hợp đối tượng dính hoặc nằm trên nhau. Điều này do khi tìm contours và lắp đầy trên mặt nạ cạnh sẽ làm mất đi những đối tượng nằm trên nhau hoặc dính vào nhau sẽ bị tính là 1.

4.3 Trường hợp đối với ảnh gạo

4.3.1 Phương pháp chung

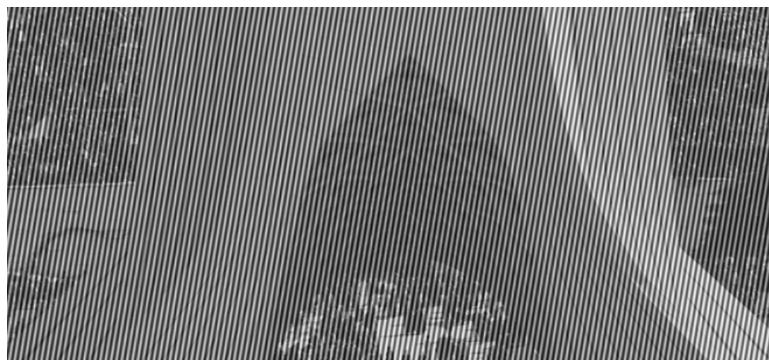


Kết quả trên tập ảnh cô đưa ra :

Acc = 99.88%

Pre = 94.55%, do trong quá trình xử lý và lọc nhiễu, các hạt gạo quá nhỏ sẽ bị loại bỏ vì bị coi là nhiễu.

- Ưu điểm:
 - Hoạt động tự động, người dùng không cần chọn các tham số nhưng có thể tùy chọn các tham số chính xác hơn trong trường hợp cụ thể
 - Bộ lọc nhiễu tần số hoạt động tự động và rất tốt, thử nghiệm trên nhiều ảnh tương tự đều loại bỏ tốt tuy nhiên chưa loại được các dạng quá phức tạp vd:



- Độ chính xác **Acc** cao, không bị nhận nhầm đối tượng.
- Hạn chế:

- Độ chính xác thấp với các ảnh thiếu cân bằng sáng ($Pre = 90\%$). Điều này do hạn chế của hàm cân bằng histogram cục bộ.
- Chưa xử lý được các trường hợp đối tượng dính hoặc nằm trên nhau. Điều này do các bước làm mờ, khử median, và nhị phân ảnh chưa tách rời được các đối tượng.
- Chưa đếm được các hạt gạo quá nhỏ ~8% kích thước hạt gạo lớn nhất. Điều này do quá trình lọc nhiễu, có thể thay đổi ngưỡng tuy nhiên sẽ gây ra nhiễu nhiều hơn.
- Thời gian hơi chậm ~ 0.13s, điều này do hàm lọc nhiễu dạng tần số hơi phức tạp.

4.3.2 Phương pháp với ảnh không thiếu cân bằng sáng

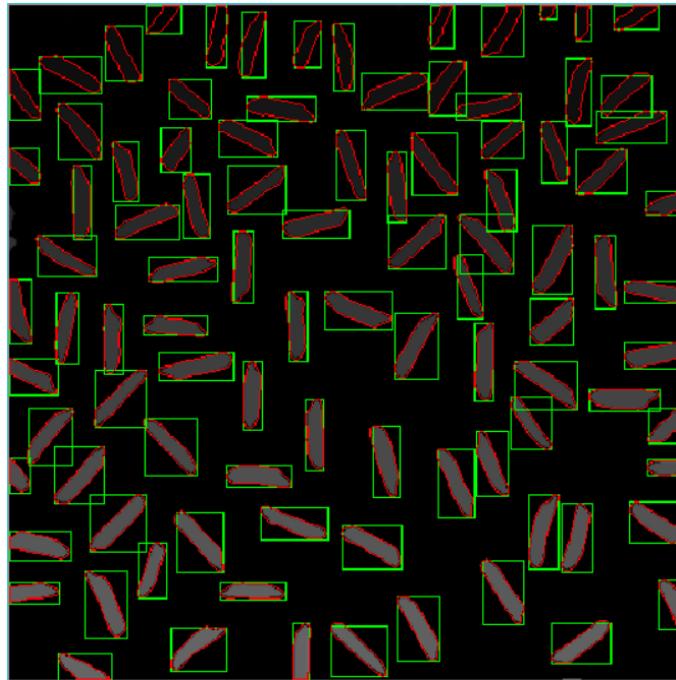


Acc = 99,67%

Pre = 97,69%

- Ưu điểm:
 - Tốt hơn phương pháp chung
 - Hoạt động tự động, người dùng không cần chọn các tham số nhưng có thể tùy chọn các tham số chính xác hơn trong trường hợp cụ thể
 - Bộ lọc nhiễu tần số hoạt động tự động và rất tốt, thử nghiệm trên nhiều ảnh tương tự đều loại bỏ tốt tuy nhiên chưa loại được các dạng quá phức tạp vd:
 - Độ chính xác **Acc** cao, không bị nhận nhầm đối tượng.
- Hạn chế:
 - Chưa xử lý được các trường hợp đối tượng dính hoặc nằm trên nhau. Điều này do các bước làm mờ, khử median, và nhị phân ảnh chưa tách rời được các đối tượng.
 - Chưa đếm được các hạt gạo quá nhỏ ~8% kích thước hạt gạo lớn nhất. Điều này do quá trình lọc nhiễu, có thể thay đổi ngưỡng tuy nhiên sẽ gây ra nhiễu nhiều hơn.
 - Thời gian hơi chậm ~ 0.13s, điều này do hàm lọc nhiễu dạng tần số hơi phức tạp

4.3.3 Phương pháp với ảnh thiếu cân bằng sáng



Kết quả với ảnh thiếu cân bằng sáng:

Acc = 100%

Pre = 97%

- **Ưu điểm:**
 - Tốt hơn phương pháp chung
 - Hoạt động tự động, người dùng không cần chọn các tham số nhưng có thể tùy chọn các tham số chính xác hơn trong trường hợp cụ thể
 - Bộ lọc nhiễu tần số hoạt động tự động và rất tốt, thử nghiệm trên nhiều ảnh tương tự đều loại bỏ tốt tuy nhiên chưa loại được các dạng quá phức tạp vd:
 - Độ chính xác **Acc** cao, không bị nhận nhầm đối tượng, **Pre** tương đối cao.
- **Hạn chế:**
 - Chưa xử lý được các trường hợp đối tượng dính hoặc nằm trên nhau. Điều này do các bước làm mờ, khử median, và nhị phân ảnh chưa tách rời được các đối tượng.
 - Chưa đếm được các hạt gạo quá nhỏ ~8% kích thước hạt gạo lớn nhất. Điều này do quá trình lọc nhiễu, có thể thay đổi ngưỡng tuy nhiên sẽ gây ra nhiều nhiễu hơn.
 - Thời gian hơi chậm ~ 0.13s, điều này do hàm lọc nhiễu dạng tần số hơi phức tạp
 - Đối với một số trường hợp ảnh đặc biệt, hàm cân bằng histogram có thể làm mờ ảnh hơn và độ chính xác giảm xuống.

Đề tài 2. Nhận dạng đối tượng sử dụng Bag of Words (BoW)

1. Bài toán

Bài toán nhận diện đối tượng sử dụng BOW lấy ý tưởng từ BOW trong lĩnh vực xử lý ngôn ngữ tự nhiên. Trong xử lý ngôn ngữ tự nhiên, BOW là một từ điển chứa các từ xuất hiện trong văn bản và tần suất xuất hiện của các từ đó trong văn bản. Tần suất này sau đó được dùng làm đặc trưng cho văn bản để thực hiện các nhiệm vụ như phân loại văn bản. Trong thị giác máy tính, BOW là tập các điểm keypoints trong ảnh, các điểm nổi bật trong ảnh giúp phân biệt các ảnh với nhau. Nhiệm vụ chính của bài toán là đi xây dựng tập BOW này và sau đó sử dụng nó để phân loại hình ảnh.

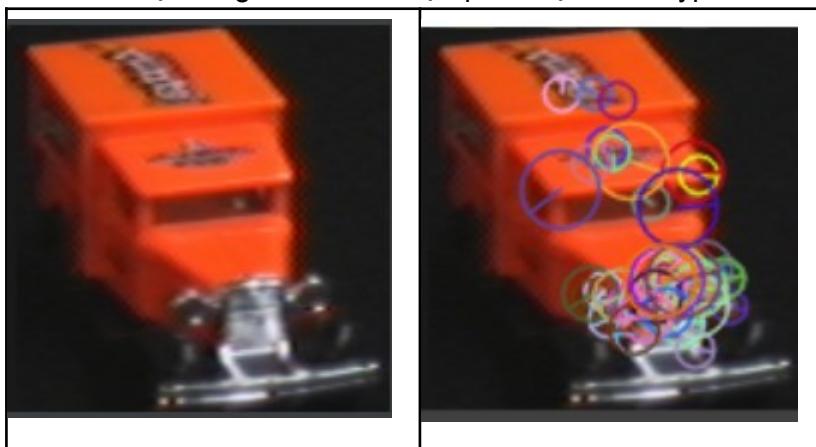
Các bước chính để thực hiện bao gồm:

1. Phát hiện các keypoints and descriptors
2. Phân cụm các descriptors
3. Xây dựng tập histogram
4. Phân loại ảnh

1.1 Phát hiện các keypoints and descriptors

Các keypoints hay các đặc trưng cục bộ(local features) là các điểm hoặc vùng nổi bật trong ảnh còn descriptors là một vector biểu diễn cho điểm(vùng) đó. Có nhiều phương pháp để phát hiện các keypoints và xây dựng descriptors, mỗi phương pháp sẽ phát hiện được một số lượng keypoints khác nhau ở từng ảnh và độ dài vector biểu diễn keypoints cũng khác nhau. Trong đề tài này, nhóm em sử dụng 3 phương pháp là SIFT, BRISK và ORB.

Ví dụ ảnh gốc và ảnh được phát hiện các keypoints:



1.2 Phân cụm các descriptors

Sau khi lấy được tất cả các descriptors của tập dữ liệu huấn luyện(train), tiến hành phân cụm các descriptors này. Các descriptors miêu tả những keypoints giống nhau sẽ có khoảng cách giữa các vector descriptor biểu diễn chúng gần nhau. Dựa vào tính chất này, sau khi chạy một thuật toán phân cụm, trong trường hợp này là KMeans, thì các descriptor

biểu diễn các điểm giống hoặc tương tự nhau sẽ được phân vào cùng một cụm. Ở mỗi cụm ta có một centroid(tâm) và vector này chính là các Words trong Bag of Words.

1.3 Xây dựng tập histogram

Các đặc trưng của ảnh giờ đây sẽ là tần suất xuất hiện của các Words trong ảnh đó. Từ tập các descriptors đã trích được ở từng ảnh, gán từng descriptor cho centroid(Word) gần nó nhất. Tần suất xuất hiện của Word trong ảnh chính là tần suất của các descriptor trong ảnh được gán vào cụm của centroid ứng với Word đó.

1.4 Phân loại ảnh

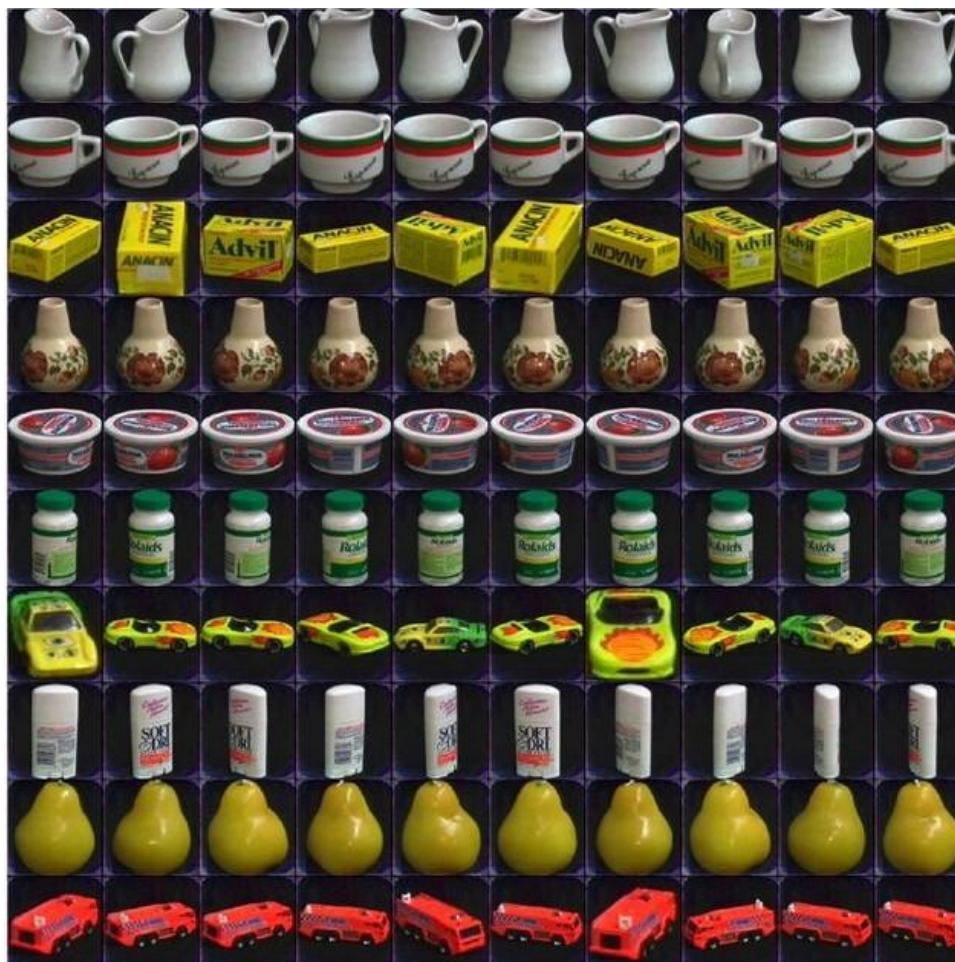
Sử dụng tập Histogram đã xây dựng để phân loại ảnh. Phương pháp phân loại nhóm sử dụng là SVM .

2. Dữ liệu

2.1 COIL100

Nguồn: <https://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php>

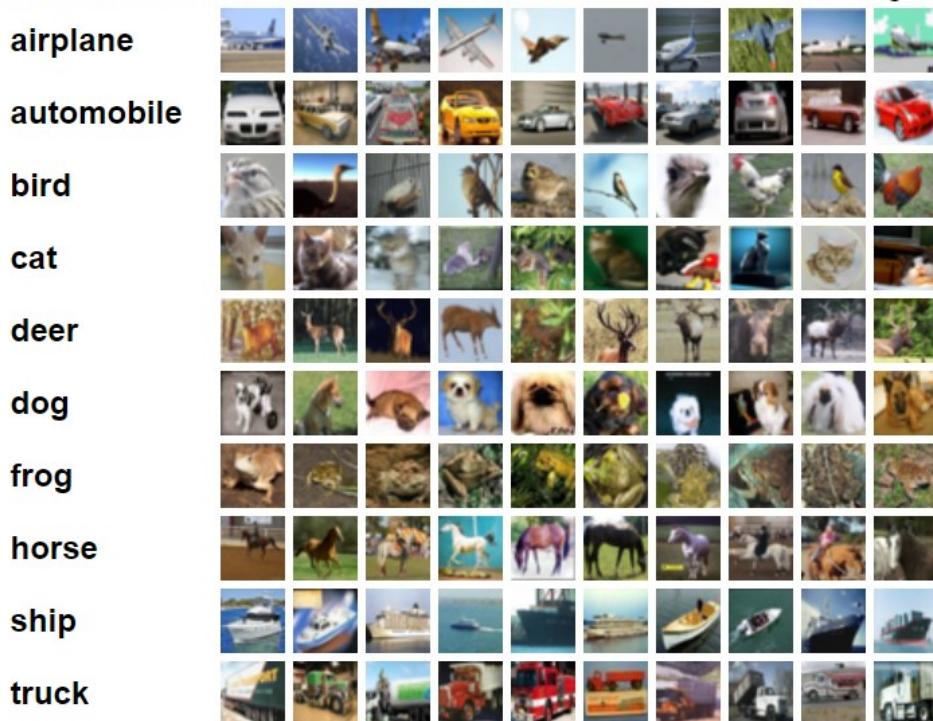
COIL100 là bộ dữ liệu gồm 7200 bức ảnh kích thước 128x128 pixels chụp cận của 100 đồ vật, mỗi đồ vật có 72 bức ảnh. 72 ảnh ứng với mỗi đồ vật được chụp ở các góc sai khác nhau 5 độ.



2.2 CIFAR10

Nguồn: <https://www.cs.toronto.edu/~kriz/cifar.html>

Bộ dữ liệu đầy đủ gồm 60000 ảnh kích thước 28x28 pixel thuộc 10 lớp, mỗi lớp có 6000 ảnh.



2.3 Một vài bộ dữ liệu thu thập từ Internet khác

Bộ dữ liệu 7 classes: **city**, **face**, **green**, **house_building**, **house_indoor**, **office**, **sea**. Kích thước các ảnh khác nhau, số lượng dữ liệu mỗi lớp không đều nhau. Có khoảng tất cả 1000 ảnh.

Nguồn: <https://github.com/gurkandemir/Bag-of-Visual-Words/tree/master/dataset>

city	face	green	house_building	house_indoor	office	sea

3. Phương pháp

Nhằm mục đích làm quen với phương pháp BOW và đánh giá các yếu tố ảnh hưởng đến kết quả, nhóm em thực nghiệm trên các bộ dữ liệu khác nhau và sử dụng các phương pháp lựa chọn đặc trưng khác nhau, các tham số khác nhau rồi so sánh kết quả. Ở mỗi phương pháp, thử nghiệm với số cụm khác nhau, tìm ra bộ tham số SVM tốt nhất tương ứng với số cụm đó từ tập train và đánh giá kết quả trên tập test. Kết quả trên tập test dùng để so sánh các phương pháp với nhau.

Sau đây là các phương pháp trích đặc trưng cục bộ và phương pháp phân loại sử dụng:

3.1 SIFT(Scale-Invariant Feature Transform)

Lợi ích chính của việc sử dụng đặc trưng SIFT là chúng không bị ảnh hưởng bởi kích thước và hướng của ảnh

Các bước trong quá trình SIFT:

- **Constructing a Scale Space:** xây dựng scale space, đảm bảo các feature không phụ thuộc vào kích thước của ảnh
- **Keypoint Localisation:** phát hiện các keypoints của ảnh
- **Orientation Assignment:** đảm bảo các keypoints không bị ảnh hưởng bởi hướng của ảnh
- **Keypoint Descriptor:** Gán cho mỗi keypoint một descriptor biểu diễn nó.

Sau các bước này, mỗi keypoint trong ảnh được biểu diễn bởi một vector 128 chiều.

3.2 BRISK

Các bước thực hiện giống với SIFT, nhưng mỗi keypoint được biểu diễn bởi vector 64 chiều.

3.3 ORB

Các bước thực hiện giống với SIFT, nhưng mỗi keypoint được biểu diễn bởi vector 32 chiều.

3.4 SVM

Là phương pháp phân loại có giám sát với ý tưởng chính là tìm các siêu phẳng phân chia các điểm dữ liệu trong không gian sao cho khoảng cách giữa các siêu phẳng là lớn nhất. Có nhiều tham số cần tối ưu cho một mô hình SVM. Trong project này, nhóm em thử nghiệm tối ưu 3 tham số với các giá trị thử nghiệm là:

```
{'C': [0.1, 1, 10, 100, 1000],  
 'gamma': [1, 0.1, 0.01, 0.001, 0.0001],  
 'kernel': ['rbf', 'linear']}
```

Ngoài ra, đối với một số bộ dữ liệu không cân bằng giữa các lớp, nhóm em sử dụng thêm trọng số để việc phân loại chính xác hơn.

4. Kết quả thực nghiệm

4.1 Bộ COIL

4.1.1 ORB

k	Best SVM parameters	Test accuracy
100	{'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}	0.480
200	{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}	0.491
500	{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}	0.544
1000	{'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}	0.54

4.1.2 BRISK

k	Best SVM parameters	Test accuracy
100	{'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}	0.504
200	{'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}	0.534
500	{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}	0.578
1000	{'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}	0.559

4.1.3 SIFT

k	Best SVM parameters	Test accuracy
100	{'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}	0.744
200	{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}	0.787
500	{'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}	0.834
1000	{'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}	0.83

4.2 Bộ 7 Classes

4.2.1 ORB

k	best SVM parameters	Test accuracy
100	{'C': 1, 'gamma': 0.01, 'kernel': 'rbf'}	0.480
200	{'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}	0.476
500	{'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}	0.529

4.2.2 BRISK

k	best SVM parameters	Test accuracy
100	{'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}	0.548
200	{'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}	0.490
500	{'C': 500, 'gamma': 0.001, 'kernel': 'rbf'}	0.519

4.2.3 SIFT

k	best SVM parameters	Test accuracy
100	{'C': 1, 'gamma': 0.01, 'kernel': 'rbf'}	0.610
200	{'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}	0.533
500	{'C': 1, 'gamma': 0.01, 'kernel': 'rbf'}	0.619

4.3 Bộ CIFAR

Các phương pháp trên không phát hiện ra được các keypoints và descriptor đối với bộ CIFAR.

5. Đánh giá và kết luận

Ở các thử nghiệm, SIFT là phương pháp cho kết quả tốt nhất. Điều đó cho thấy khi tăng số chiều biểu diễn vector thì độ chính xác cũng sẽ tăng.

Kết quả của phương pháp SIFT cao vượt trội ở bộ dữ liệu COIL100 khi độ chính xác trên tập test đạt trên 83%. Điều này có thể giải thích bởi phương pháp SIFT phù hợp với bộ dữ liệu COIL100. Vì bộ dữ liệu này ở các lớp là cùng một vật thể ở các hướng khác nhau, mà SIFT giúp nhận diện các đối tượng mà không quan tâm đến hướng của chúng.

- Các phương pháp BRISK và ORB cho kết quả độ chính xác khoảng 50% ở cả 2 bộ dữ liệu, BRISK có độ chính xác cao hơn.
- Ở bộ dữ liệu 7 lớp, kết quả SIFT cao hơn so với 2 phương pháp còn lại nhưng không quá chênh lệch. Vì bộ dữ liệu này các ảnh tuy thuộc cùng chủ đề nhưng các thực thể khác nhau nên lợi thế của SIFT là không bị ảnh hưởng bởi hướng và kích thước ảnh không phát huy được với bộ dữ liệu này.
- Mỗi bộ dữ liệu sẽ có một kích thước BOW phù hợp(tức tham số k). Với ảnh có càng nhiều chi tiết thì k càng cao. Qua thực nghiệm có thể thấy khi tăng dần k từ 100 thì độ chính xác thường tăng đến giá trị phù hợp nhất cho bộ dữ liệu đó rồi sẽ giảm. Trong các thực nghiệm trên, giá trị k = 500 thường cho kết quả tốt nhất.
- Phương pháp BOW sẽ phù hợp với một số bộ dữ liệu nhất định. Nếu có dữ liệu tốt và phương pháp lựa chọn đặc trưng và phân loại phù hợp thì sẽ có kết quả cao. Đây là một phương pháp phân loại truyền thống nên kết quả không thể so sánh với Deep Learning.
- Đề xuất cải thiện và giải pháp: Trong quá trình trích đặc trưng cục bộ, có nhiều ảnh không phát hiện được đặc trưng nào. Ở bộ COIL100(ảnh 128x128), tỉ lệ không phát hiện được đặc trưng là khoảng 5%(khoảng gần 300 ảnh). Ở bộ CIFAR10(ảnh 28x28), tỉ lệ này là 97% với BRISK và 100% với ORB. Do vậy, có thể kết hợp BOW với các phương pháp tăng cường độ phân giải ảnh.

Tài liệu tham khảo

1. Numpy. <https://numpy.org/>
2. OpenCV. <https://opencv.org/>
3. ContourFeatures. OpenCV.
https://docs.opencv.org/master/dd/d49/tutorial_py_contour_features.html
4. Coil100. <https://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php>
5. Cifar10. <https://www.cs.toronto.edu/~kriz/cifar.html>
6. BoVW. <https://github.com/gurkandemir/Bag-of-Visual-Words/tree/master/dataset>
7. Introduction to SIFT (Scale-Invariant Feature Transform). OpenCV.
https://docs.opencv.org/master/da/df5/tutorial_py_sift_intro.html