Bài tập thực hành
CLUSTERING

GVHD: TS. Lê Đình Duy, ThS. Mai Tiến Dũng
SVTH: Trương Vĩ Thiên - MSSV: 14520874 - KHTN2014
Khoa: Khoa học Máy tính - Môn: Máy học trong Thị giác Máy tính
Trường ĐH Công nghệ Thông tin, Tp. HCM, Việt Nam

Sơ lược

Ở bài tập thực hành này, chúng ta tìm hiểu chính về 4 phương pháp phân cụm dữ liệu (clustering) là K-Means, DBSCAN, Spectral Clustering và Agglomerative Clustering; bên cạnh đó còn có 2 phương pháp rút trích đặc trưng (feature extracting) là Local Binary Pattern (LBP) và Histogram of Gradient (HOG). Các tập dữ liệu (datasets) được sử dụng trong bài tập gồm Hand-written digits và Labeled Human Faces có sẵn từ thư viện SciLearn.

Mục lục

Sơ lược **1**

Mục lục **1**

Giới thiệu **2**

Phương pháp Feature Extracting 2

Local Binary Pattern 2

Histogram of Oriented Gradient 2

Phương pháp Clustering 2

K-Means 2

DBSCAN 2

Spectral Clustering 2

Agglomerative Clustering 3

Thực nghiệm **3**

Sinh dữ liệu từ Gaussians và phân cụm bằng thuật toán K-Means 3

Phân cụm dữ liệu Hand-written digits bằng 4 thuật toán phân cụm 3

Phân cụm dữ liệu Human Faces trích xuất theo phương pháp LBP bằng 4 thuật toán phân cụm 5

Phân cụm dữ liệu Human Faces trích xuất theo phương pháp HOG bằng 4 thuật toán phân cụm 7

Tài liệu tham khảo 8

1. Giới thiệu

Bài toán Clustering là bài toán tìm cách nhóm các đối tượng đã cho vào các cụm (gọi là clusters) sao cho các đối tượng cùng một cụm thì có giá trị tương tự nhau và các đối tượng khác cụm thì không tương tự nhau.

1.1. Phương pháp Feature Extracting

Chèn thư viện:

```
from skimage.feature   import local_binary_pattern, hog
def load_dataset():
     faces = fetch_lfw_people(min_faces_per_person=70,
resize=0.4)
     data_lbp = []
     data_hog = []
     target = faces.target
     for image in faces.images:
     feature_lbp = local_binary_pattern(image, P=24, R=4)
     _, feature_hog = hog(image, orientations=8,
pixels_per_cell=(16,16), cells_per_block=(1, 1),
visualise=True)
     data_lbp.append(feature_lbp.flatten())
     data_hog.append(feature_hog.flatten())
     np.save(file='data_target.npy', arr=target)
     np.save(file='data_lbp_feature.npy', arr=data_lbp)
     np.save(file='data_hog_feature.npy', arr=data_hog)
```

1.1.1. Local Binary Pattern

LBP là một phương pháp rút trích đặc trưng dựa trên sự chênh lệch độ sáng của các điểm lân cận được phát triển bởi Ojala và các đồng nghiệp. Tại một vị trí pixel (Xo,Yo), từng giá trị độ sáng xung quanh pixel sẽ được so sánh với độ sáng pixel đang xét, nếu lớn hơn sẽ thay thành giá trị 1, ngược lại thay thành giá trị 0. Từ đó ra được một dãy nhị phân biểu diễn cho pixel đang xét.

Từ ý nghĩ trên, tác giả và đồng nghiệp mở rộng toán tử LBP đến lân cận tròn với các bán kính và số điểm lân cận khác nhau.

1.1.2. Histogram of Oriented Gradient

HOG là một phương pháp rút trích đặc trưng dựa trên sự phân bố về cường độ và hướng của các cạnh (edge) của ảnh được phát triển bởi Robert K. McConnel.

1.2. Phương pháp Clustering

Chèn thư viện:

```
from sklearn.cluster
import KMeans, DBSCAN, SpectralClustering,
AgglomerativeClustering
```

1.2.1. K-Means

K-Means là thuật toán phân cụm đã biết trước số lượng cụm (n_clusters > 0), sao khoảng cách Euclidean của các điểm thuộc cụm đến tâm cụm (centroids). Sử dụng:

```
def tech_kmeans(data, n_clusters):
    return KMeans(n_clusters=n_clusters).fit(data)
```

1.2.2. DBSCAN

DBSCAN là thuật toán phân cụm các điểm cùng một phạm vi, các điểm không thuộc cụm và có sự phân bố rời rạc sẽ được coi là một noise. Sử dụng:

```
def tech_dbscan(data,eps,min_samples):
    return
DBSCAN(eps=eps,min_samples=min_samples,algorithm='kd_tree
').fit(StandardScaler().fit_transform(data))
```

1.2.3. Spectral Clustering

Spectral Clustering là thuật toán đưa các điểm tương đồng về một hướng (trong đồ thị có n hướng tương đương n cụm). Sử dụng:

```
def tech_spectral(data,n_clusters):
    return
SpectralClustering(n_clusters=n_clusters).fit(pairwise.co
sine_similarity(data))
```

1.2.4. Agglomerative Clustering

Agglomerative Clustering là thuật toán tìm các điểm gần nhau nhất để kết hợp lại thành một clusters và tiếp tục cho đến khi chỉ còn một cluster. Sử dụng:

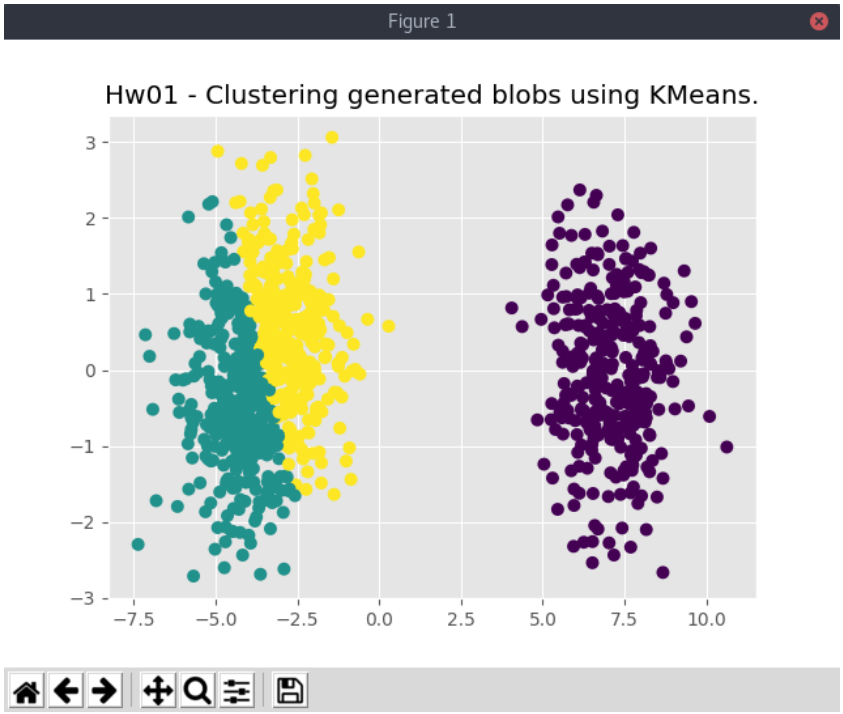
```
def tech_agglo(data,n_clusters):
    return
AgglomerativeClustering(n_clusters=n_clusters).fit(data)
```

2. Thực nghiệm

Github: https://github.com/truongvithien/Clustering

2.1. Sinh dữ liệu từ Gaussians và phân cụm bằng thuật toán K-Means

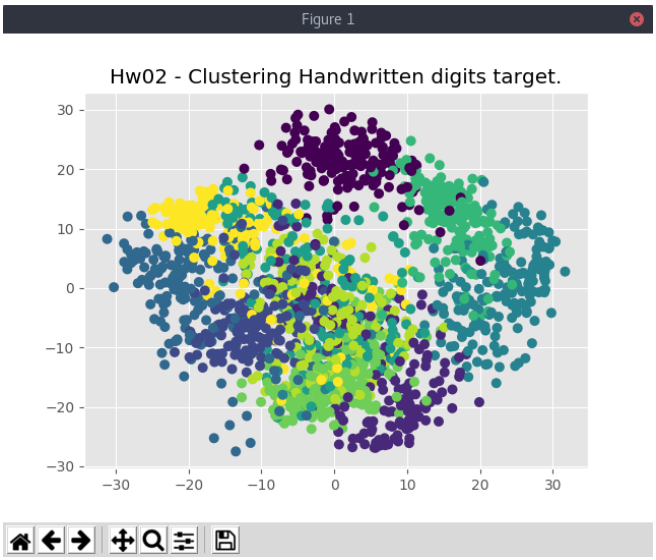
Sinh 1000 điểm ngẫu nhiên với mỗi điểm gồm 2 đặc trưng, sau đó dùng phương pháp K-Means để phân cụm:



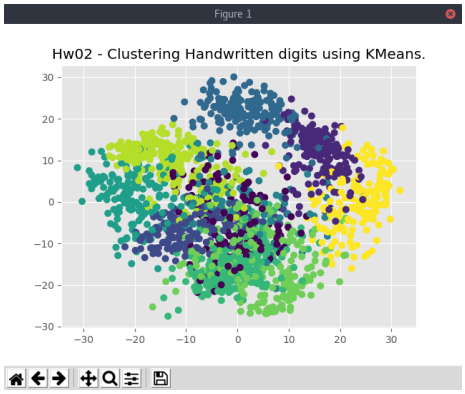
Hình 1. Kết quả clustering dữ liệu sinh ngẫu nhiên bằng phương pháp K-Means

2.2. Phân cụm dữ liệu Hand-written digits bằng 4 thuật toán phân cụm

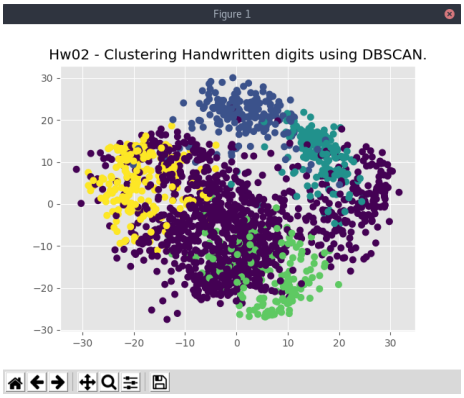
Dùng bộ dữ liệu Hand-written digits gồm 10 chữ số, sau đó dùng 4 phương pháp phân cụm khác nhau:



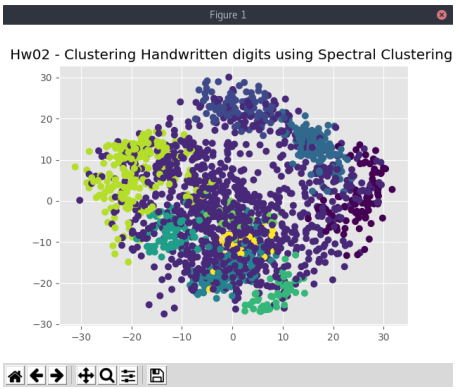
Hình 2.1. : Kết quả đã được dán nhãn để so khớp.



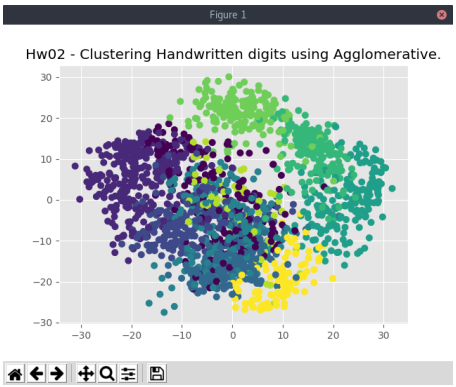
Hình 2.2. K-Means



Hình 2.3. DBSCAN



Hình 2.4. Spectral Clustering



Hình 2.5. Agglomerative

Bảng đánh giá thời gian chạy và độ chính xác:

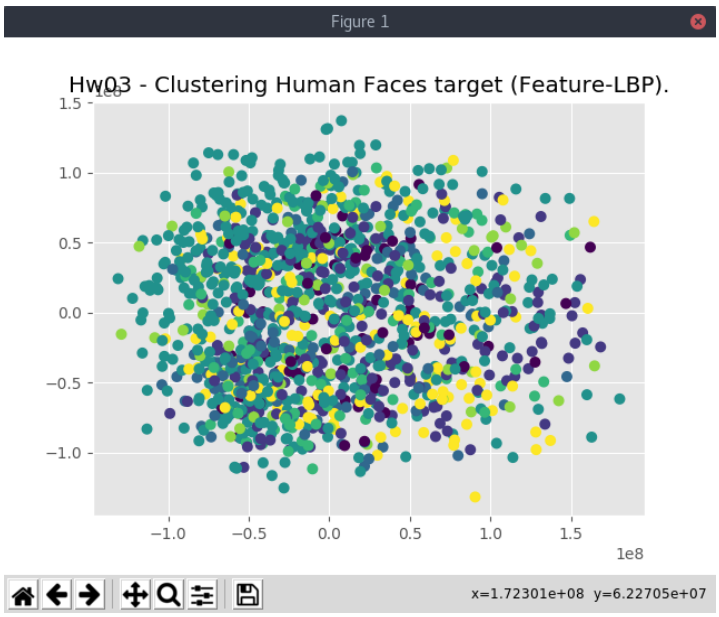
	KMeans	DBSCAN	Spectral	Agglomerative
Thời gian	0.36	0.47	1.56	0.20
Độ tương đồng	73.86%	34.86%	39.51%	85.61%

Nhận xét:

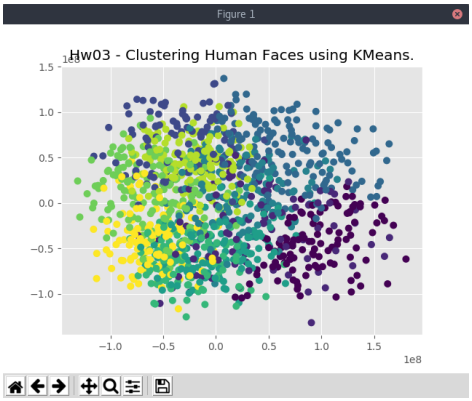
- Độ tương đồng: Target > Agglomerative > KMeans >> Spectral > DBSCAN
- Thời gian chạy: Agglomerative < KMeans < DBSCAN << Spectral
- Thuật toán Agglomerative được đánh giá cao nhất trong bộ dữ liệu này.

2.3. Phân cụm dữ liệu Human Faces trích xuất theo phương pháp LBP bằng 4 thuật toán phân cụm

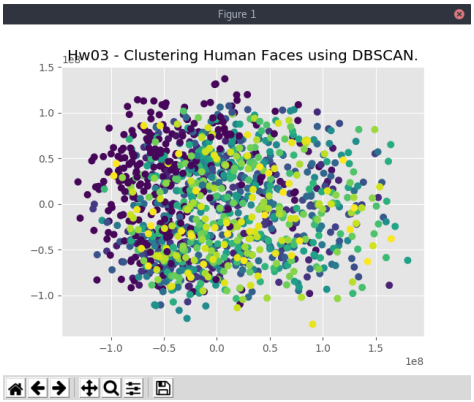
Dùng bộ dữ liệu Human faces được rút trích đặc trưng bằng phương pháp Local Binary Pattern gồm 7 nhóm, sau đó dùng 4 phương pháp phân cụm khác nhau:



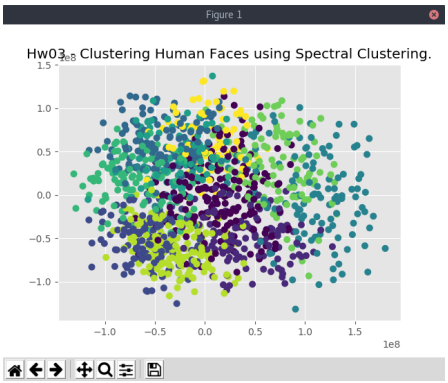
Hình 2.1. : Kết quả đã được dán nhãn để so khớp.

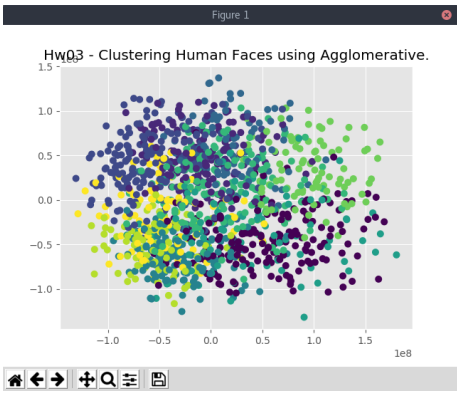


Hình 2.2. K-Means



Hình 2.3. DBSCAN





Hình 2.4. Spectral Clustering Hình 2.5. Agglomerative

Bảng đánh giá thời gian chạy và độ chính xác:

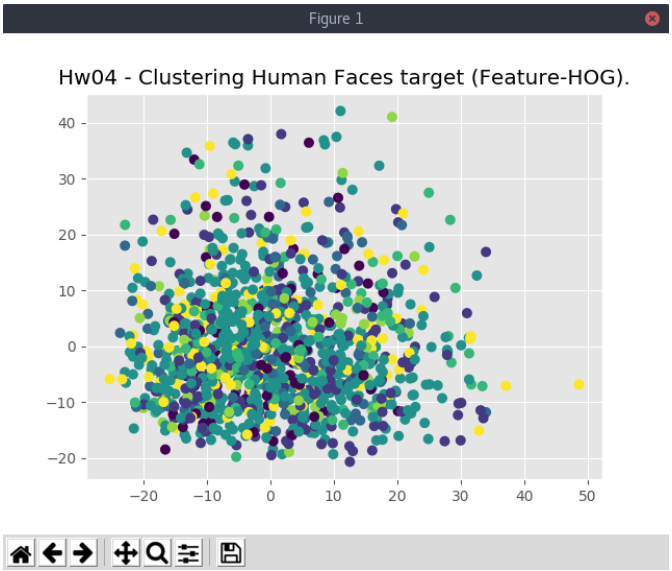
	KMeans	DBSCAN	Spectral	Agglomerative
Thời gian	3.49	7.37	0.85	1.24
Độ tương đồng	04.04%	03.92%	02.34%	06.83%

Nhận xét:

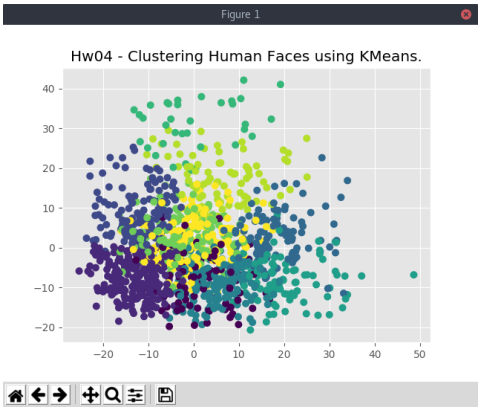
- Độ tương đồng: Target >> Agglomerative > KMeans > DBSCAN > Spectral
- Thời gian chạy: Spectral < Agglomerative << KMeans << DBSCAN
- Thuật toán Agglomerative được đánh giá cao nhất trong bộ dữ liệu này, tuy nhiên độ tương đồng với kết quả so khớp quá thấp.

2.4. Phân cụm dữ liệu Human Faces trích xuất theo phương pháp HOG bằng 4 thuật toán phân cụm

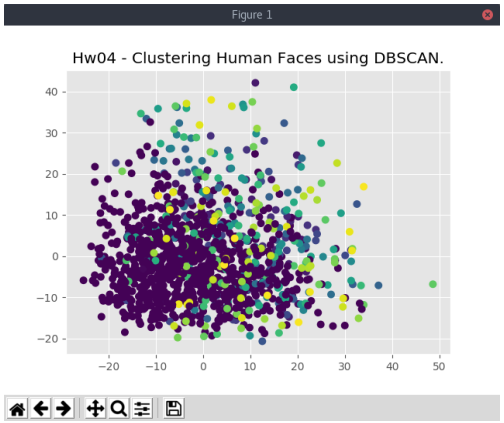
Dùng bộ dữ liệu Human faces được rút trích đặc trưng bằng phương pháp Histogram of Oriented Gradient gồm 7 nhóm, sau đó dùng 4 phương pháp phân cụm khác nhau:



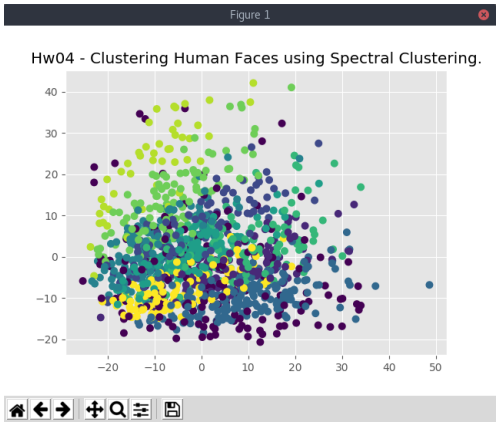
Hình 2.1. : Kết quả đã được dán nhãn để so khớp.



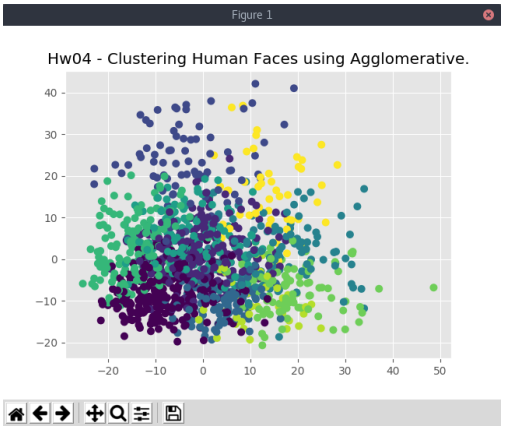
Hình 2.2. K-Means



Hình 2.3. DBSCAN



Hình 2.4. Spectral Clustering



Hình 2.5. Agglomerative

Bảng đánh giá thời gian chạy và độ chính xác:

	KMeans	DBSCAN	Spectral	Agglomerative
Thời gian	3.37	7.81	0.98	1.23
Độ tương đồng	02.03%	01.38%	00.91%	01.92%

Nhận xét:

- Độ tương đồng: Target >> KMeans > Agglomerative > DBSCAN > Spectral
- Thời gian chạy: Spectral < Agglomerative << KMeans << DBSCAN
- Thuật toán Agglomerative được đánh giá cao nhất trong bộ dữ liệu này, tuy nhiên độ tương đồng với kết quả so khớp quá thấp.

Tài liệu tham khảo

[1] Scikit-Learn, The Digit Dataset. <http://scikit-learn.org/stable/auto_examples/datasets/plot_digits_last_image.html>.

[2] Scikit-Learn, The Labeled Faces in the Wild face recognition dataset. <http://scikit-learn.org/stable/datasets/labeled_faces.html>

[3] Scikit-Learn, KMeans. <http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

[4] Scikit-Learn, DBSCAN. <http://scikit-learn.org/stable/modules/generated/sklearn.cluster.dbscan.html>

[5] Scikit-Learn, Spectral Clustering. <http://scikit-learn.org/stable/modules/generated/sklearn.cluster.spectral_clustering.html>

[6] Scikit-Learn, Agglomerative Clustering. <http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>

[7] Scikit-Image, Local Binary Pattern. <http://scikit-image.org/docs/dev/auto_examples/features_detection/plot_local_binary_pattern.html>

[8] Scikit-Image, Histogram of Oriented Gradient. <http://scikit-image.org/docs/dev/auto_examples/features_detection/plot_hog.html>