ĐẠI HỌC QUỐC GIA TP. HCM

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

──────── \* ───────

BÁO CÁO MÔN HỌC

NHẬN DẠNG THỊ GIÁC VÀ ỨNG DỤNG

**ĐỒ ÁN: PHÂN LOẠI ẢNH**

Giảng viên hướng dẫn: **TS. Lê Đình Duy**

**TS. Nguyễn Tấn Trần Minh Khang**

Học viên thực hiện: **Lê Thanh Phương – MSHV: CH1602008**

**Hồ Chí Minh, Năm 2018**

Địa chỉ liên kết

<https://github.com/phuonglt11/BaocaoVRA>

<https://drive.google.com/drive/u/0/folders/1dEHo6v25-iAqgo1xXJPxfxRyM7Zfz-8N>

Mục lục

[ĐỒ ÁN PHÂN LOẠI ẢNH 3](#_Toc503014896)

[I. GIỚI THIỆU TỔNG QUAN 3](#_Toc503014897)

[**1.** **Các dạng rút trích đặc trưng** 3](#_Toc503014898)

[**2.** **Các loại thuật toán học máy** 3](#_Toc503014899)

[**3.** **Mô hình bài toán VRA** 4](#_Toc503014900)

[**4.** **Bài toán phân lớp** 5](#_Toc503014901)

[**Step 1: Set Up Image Category Sets** 5](#_Toc503014902)

[**Step 2: Create Bag of Features** 6](#_Toc503014903)

[**Step 3: Train an Image Classifier With Bag of Visual Words** 6](#_Toc503014904)

[**Step 4: Classify an Image or Image Set** 7](#_Toc503014905)

[II. THỰC NGHIỆM CÁC LOẠI RÚT TRÍCH TRONG MATLAB 7](#_Toc503014906)

[**1.** **Histogram of oriented gradients (HOG)** 7](#_Toc503014907)

[**2.** **Local binary patterns (LBP)** 9](#_Toc503014908)

[**3.** **Create Bag of Visual Words (BoW)** 10](#_Toc503014909)

[III. MÔ TẢ BỘ DỮ LIỆU VÀ MÔ HÌNH PHÂN LOẠI ẢNH 11](#_Toc503014910)

[**1.** **Bộ dữ liệu thực nghiệm** 11](#_Toc503014911)

[**2.** **Mô hình phân loại ảnh** 11](#_Toc503014912)

[IV. THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ CỦA 3 MÔ HÌNH 12](#_Toc503014913)

[**1.** **HOG Features** 12](#_Toc503014914)

[**2.** **BoW Features** 15](#_Toc503014915)

[**3.** **AlexNet Features** 16](#_Toc503014916)

[**So sánh các features và thuật toán cho mô hình phân loại ảnh** 18](#_Toc503014917)

[V. KẾT LUẬN 19](#_Toc503014918)

[**Tài liệu tham khảo** 19](#_Toc503014919)

### ĐỒ ÁN PHÂN LOẠI ẢNH

I. GIỚI THIỆU TỔNG QUAN

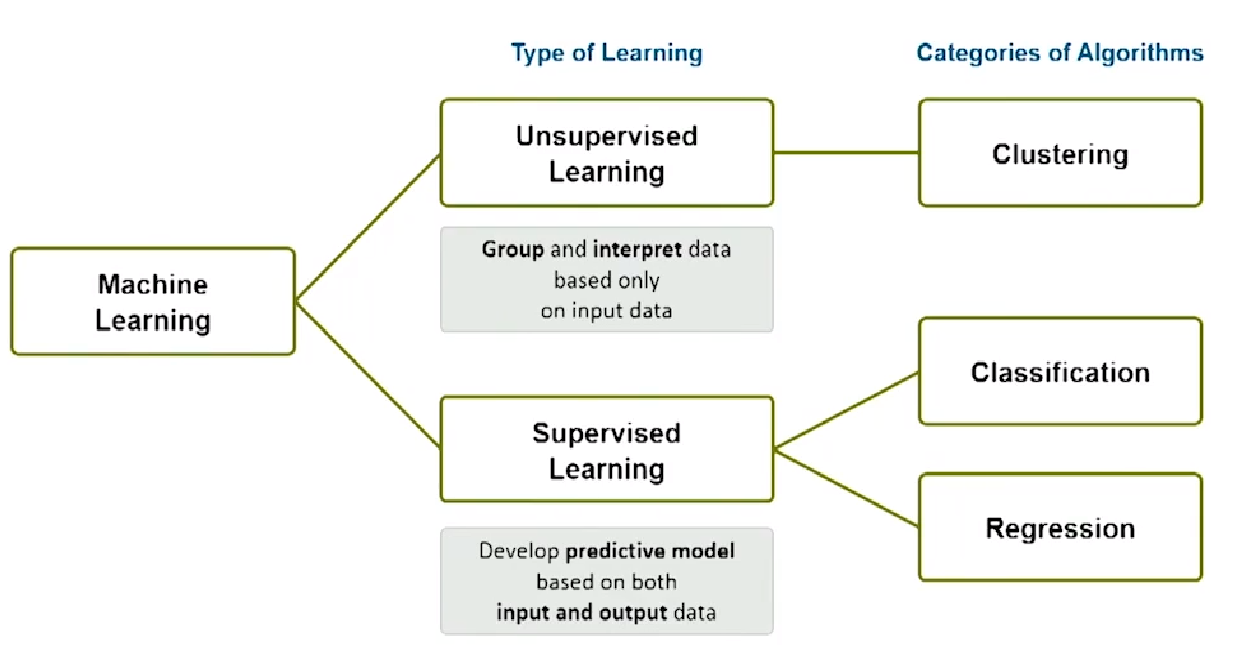
1. **Các dạng rút trích đặc trưng**

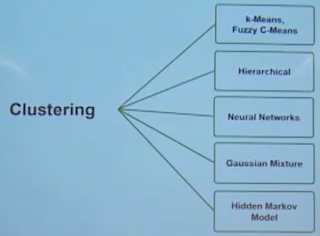
* *Histogram of oriented gradients (HOG)*
* Local binary patterns (LBP)
* Alexnet (Deep Learning)
* Bag of visual Words (BoW)
* Haar wavelets
* Color histograms, ect

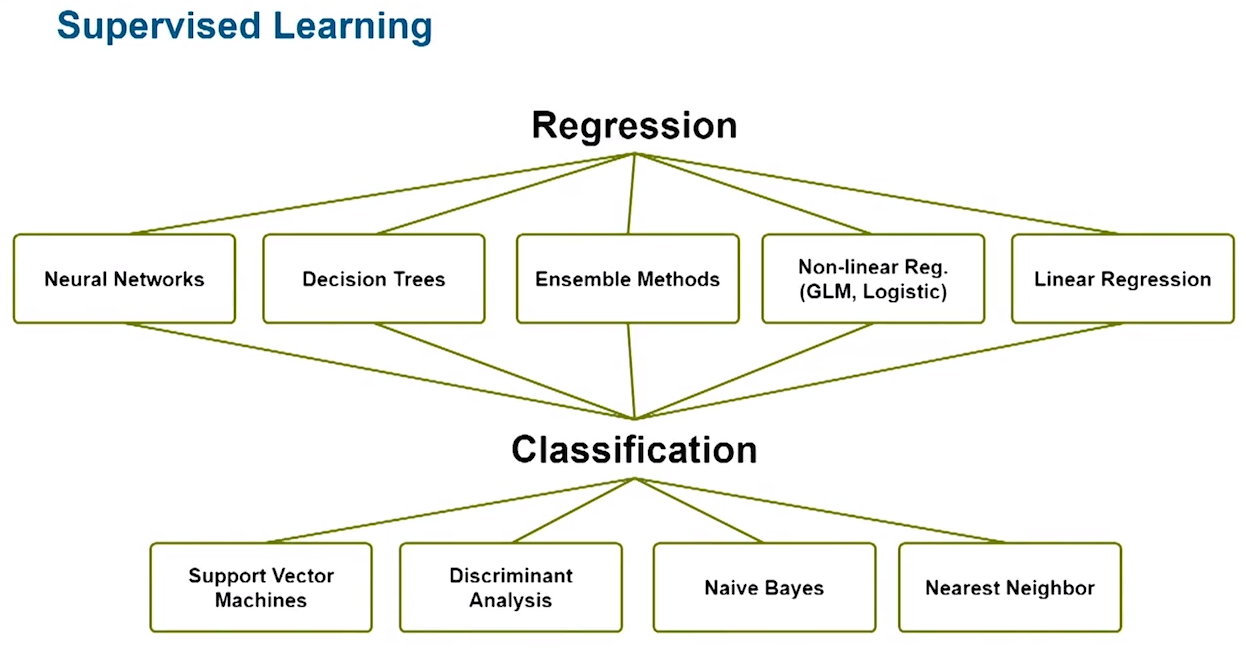
Ngoài ra, còn có các dạng phát hiện các điểm đặc trưng như: SURF, HARRIS, ect

1. **Các loại thuật toán học máy**

(mô hình tổng thể về các thuật toán học máy như bên dưới)

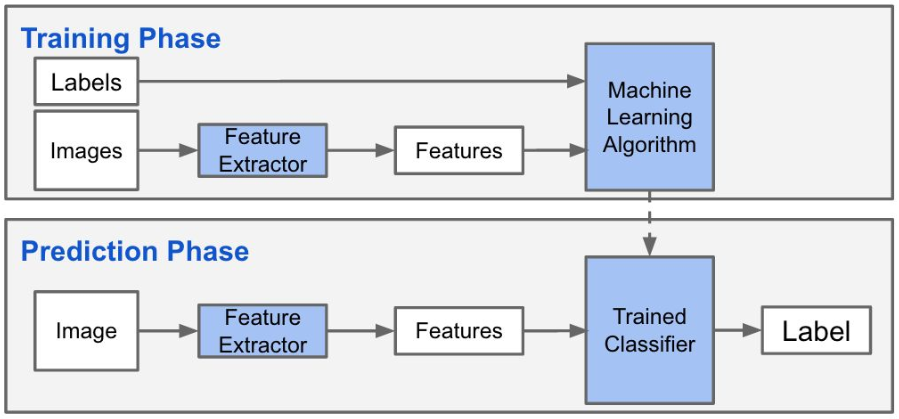






Ngoài các thuật toán như mô hình tổng thể trên còn có thuật toán khá nổi tiếng gần đây đó là Deep Learning.

1. **Mô hình bài toán VRA**



Trong các mô hình nhận dạng thị giác đa phần chúng trải qua các bước sau đây:

* Xây dựng bộ dữ liệu data train
* Huấn luyện dữ liệu (load data train)
* *Rút trích đặc trưng (chọn một loại rút trích phù hợp)*
* *Xây dựng model (chọn một thuật toán học máy phù hợp)*
* Nạp dữ liệu test (load data test, có thể là Image hoặc video)
* Dựa vào model để phán đoán

1. **Bài toán phân lớp**

Để mô tả cho bài toán phân lớp chúng tôi cũng không biết giải thích như thế nào là hợp lí và ngắn gọn. Do đó, chúng tôi xin trình bày phần này bằng một bài toán: Image Classification with Bag of Visual Words

Image Classification with Bag of Visual Words

Ta có thể mô tả ngắn gọn phần này bằng Matlab qua các bước sau:

**Step 1: Set Up Image Category Sets**

Chia bộ dữ liệu ban đầu (DataSet) thành hai phần con là training và test. Sử dụng hàm [imageDatastore](https://www.mathworks.com/help/matlab/ref/matlab.io.datastore.imagedatastore.imagedatastore.html) lưu trữ tất cả các image của DataSet. Chúng ta có thể chia DataSet đó ra thành hai phần con là training và test bằng hàm [splitEachLabel](https://www.mathworks.com/help/matlab/ref/datastore.spliteachlabel.html).

Read the category images and create image sets.

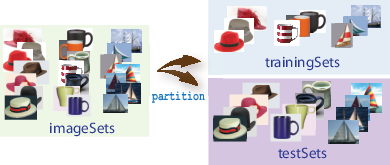
setDir = fullfile(toolboxdir('vision'),'visiondata','imageSets');

imds = imageDatastore(setDir,'IncludeSubfolders',true,'LabelSource',...

'foldernames');

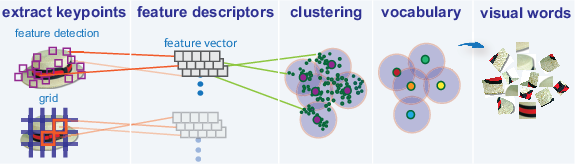
Chia DataSet đó thành hai phần con là training và test theo tỉ lệ % tương ứng.

[trainingSet,testSet] = splitEachLabel(imds,0.3,'randomize');



**Step 2: Create Bag of Features**

Khởi tạo một visual vocabulary hoặc bag of features bằng việc trích xuất feature descriptors từ các image tương ứng của mỗi category. Hàm [bagOfFeatures](https://www.mathworks.com/help/vision/ref/bagoffeatures-class.html) định nghĩa features, or visual words, sử dụng thuật toán [k-means clustering](https://www.mathworks.com/help/stats/k-means-clustering.html#bq_679x-19)

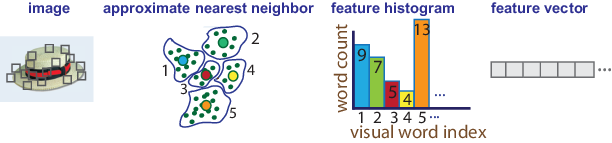


**Step 3: Train an Image Classifier With Bag of Visual Words**

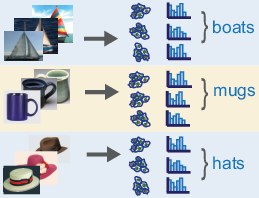
Hàm [trainImageCategoryClassifier](https://www.mathworks.com/help/vision/ref/trainimagecategoryclassifier.html) trả về một lớp image. Hàm này huyến luyện cho nhiều lớp image và sử dụng error-correcting output codes (ECOC) framework with binary support vector machine (SVM) classifiers. Hàm trainImageCategoryClassfier được trả về bởi đối tượng [bagOfFeatures](https://www.mathworks.com/help/vision/ref/bagoffeatures-class.html) để mã hóa image trong image set đến histogram of visual words.

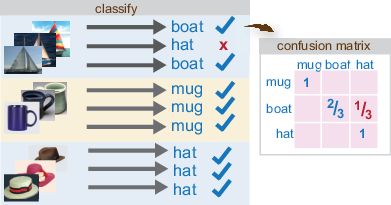
Hàm [bagOfFeatures](https://www.mathworks.com/help/vision/ref/bagoffeatures-class.html) sử dụng phương pháp mã hóa để mã hóa cho từng image trong data set. Chúng có chức năng detects and extracts features từ image và sau đó sử dụng thuật toán approximate nearest neighbor để xây dựng một feature histogram cho mỗi image. Histogram này trở thành một feature vector cho image.

1. Hàm [bagOfFeatures](https://www.mathworks.com/help/vision/ref/bagoffeatures-class.html) sử dụng phương pháp mã hóa để mã hóa cho từng image trong data set. Chúng có chức năng detects and extracts features từ image và sau đó sử dụng thuật toán approximate nearest neighbor để xây dựng một feature histogram cho mỗi image. Histogram này trở thành một feature vector cho image



1. Lặp lại step 1 cho mỗi image trong training set để tạo training data.



1. Đánh giá chất lượng của việc phân loại. Sử dụng hàm [imageCategoryClassifier](https://www.mathworks.com/help/vision/ref/imagecategoryclassifier-class.html) để kiểm tra việc phân loại đối với image set hợp lệ. 

**Step 4: Classify an Image or Image Set**

Sử dụng hàm [imageCategoryClassifier](https://www.mathworks.com/help/vision/ref/imagecategoryclassifier-class.html) phán đoán một hình ảnh mới thuộc category nào.

II. THỰC NGHIỆM CÁC LOẠI RÚT TRÍCH TRONG MATLAB

1. **Histogram of oriented gradients (HOG)**

%Read the image of interest.

function HOGFeatureCellsize()

%Read the image of interest.

I1 = imread('ThanhPhuong.jpg');

%Extract HOG features.

[hog1,visualization] = extractHOGFeatures(I1,'CellSize',[128 128]);

%Display the original image and the HOG features.

subplot(1,2,1);

imshow(I1);

subplot(1,2,2);

plot(visualization);

end

Thay đổi cellsize ta lần lượt thu được các kết quả như sau:

|  |  |  |
| --- | --- | --- |
| **CellSize = [32 32]** | **CellSize = [64 64]** | **CellSize = [128 128]** |
|  |  |  |

function HOGFeatureAroundCorner()

%Read in the image of interest.

I2 = imread('GaTrong.jpg');

%Detect and select the strongest corners in the image.

corners = detectFASTFeatures(rgb2gray(I2));

strongest = selectStrongest(corners,3);

%Extract HOG features.

[hog2, validPoints,ptVis] = extractHOGFeatures(I2,strongest);

%Display the original image with an overlay of HOG features around the strongest corners.

figure;

imshow(I2);

hold on;

plot(ptVis,'Color','blue');

Thay đổi corner lần lượt thu được các kết quả như sau:

|  |  |  |
| --- | --- | --- |
| **Corner = 3** | **Corner = 5** | **Corner = 7** |
|  |  |  |

1. **Local binary patterns (LBP)**

function LBPFeature()

%Read in a sample image and convert it to grayscale.

I = imread('ThanhPhuong.jpg');

I = rgb2gray(I);

%Extract unnormalized LBP features so that you can apply a custom normalization.

lbpFeatures = extractLBPFeatures(I,'CellSize',[64 64],'Normalization','None');

%Reshape the LBP features into a number of neighbors -by- number

% of cells array to access histograms for each individual cell.

numNeighbors = 8;

numBins = numNeighbors\*(numNeighbors-1)+3;

lbpCellHists = reshape(lbpFeatures,numBins,[]);

%Normalize each LBP cell histogram using L1 norm.

lbpCellHists = bsxfun(@rdivide,lbpCellHists,sum(lbpCellHists));

%Reshape the LBP features vector back to 1-by- N feature vector.

lbpFeatures = reshape(lbpCellHists,1,[]);

%Display the original image and the LBP features.

subplot(1,2,1);

imshow(I);

subplot(1,2,2);

plot(lbpFeatures);

end

Thay đổi cellsize ta lần lượt thu được các kết quả như sau:

|  |  |  |
| --- | --- | --- |
| **CellSize = [64 64]** | **CellSize = [128 128]** | **CellSize = [256 256]** |
|  |  |  |

1. **Create Bag of Visual Words (BoW)**

function CreateABagOfVisualWordsFeature()

%% Create a Bag of Visual Words

% Load two image sets.

setDir = fullfile('AnimalData');

imgSets = imageSet(setDir,'recursive');

%% Pick the first two images from each image set to create training sets.

trainingSets = partition(imgSets,10);

%% Create the bag of features.

bag = bagOfFeatures(trainingSets,'Verbose',false);

% Compute histogram of visual word occurrences for one of the images.

% Store the histogram as feature vector.

img = read(imgSets(1),1);

featureVector = encode(bag,img);

%% Display a figure for Bag of Visual Words

figure

bar(featureVector)

title('Visual word occurrences')

xlabel('Visual word index')

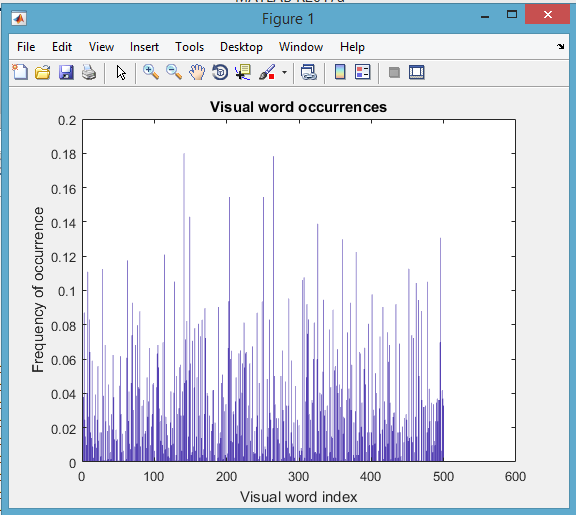
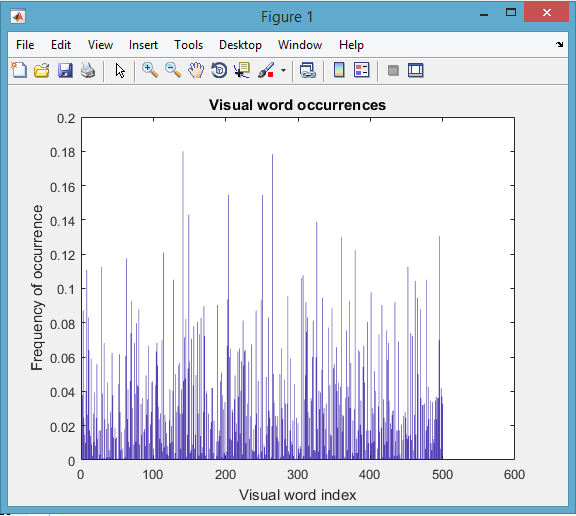
ylabel('Frequency of occurrence')

end

Thay đổi số lượng data train là : 10, 30 cho mỗi lớp 50 images, lần lượt thu được các kết quả như sau:

( trainingSets = partition(imgSets,10);)

( trainingSets = partition(imgSets,30);)

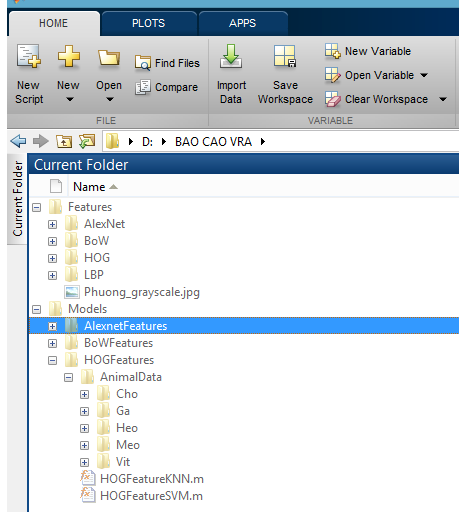


III. MÔ TẢ BỘ DỮ LIỆU VÀ MÔ HÌNH PHÂN LOẠI ẢNH

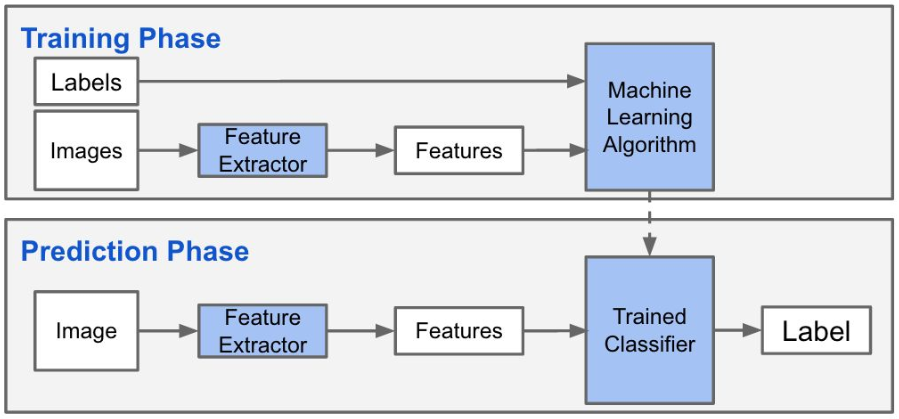
1. **Bộ dữ liệu thực nghiệm**

Bộ dữ liệu thực nghiệm do chúng tôi tạo ra, chúng có một thư mục gốc AnimalData và 5 thư mục mục con, mỗi thư mục đại diện cho một lớp đối tượng trong đó chứa 50 image và được tổ chức như hình sau

Kích thước image: 227x227 fixel, Cellsize: 8bit. Sử dụng phần mềm Adobe PhotoShop để xử lí image được download từ internet và dữ liệu được tổ chức như hình sau.



1. **Mô hình phân loại ảnh**



Trong đồ án phân loại ảnh, để đánh giá mô hình chúng tôi kết hợp rút trích đặc trưng và thuật toán học máy lần lượt có tên như sau:

* **Mô hình BoW: như đã giới thiệu phần trên (BoW)**
* Mô hình HOG: sử dụng rút trích đặc trưng Histogram of oriented gradients (HOG) và thuật toán K-nearest neighbor (k-NN)
* Mô hình HOG: sử dụng rút trích đặc trưng Histogram of oriented gradients (HOG) và thuật toán [Support Vector Machine](https://viblo.asia/p/support-vector-machine-trong-hoc-may-mot-cai-nhin-don-gian-hon-XQZkxoQmewA)(SVM)
* **Mô hình Alexnet: sử dụng rút trích đặc trưng Alexnet và thuật toán** [**Support Vector Machine**](https://viblo.asia/p/support-vector-machine-trong-hoc-may-mot-cai-nhin-don-gian-hon-XQZkxoQmewA)**(k-NN)**
* **Mô hình Alexnet: sử dụng rút trích đặc trưng Alexnet và thuật toán** [**Support Vector Machine**](https://viblo.asia/p/support-vector-machine-trong-hoc-may-mot-cai-nhin-don-gian-hon-XQZkxoQmewA)**(SVM)**

IV. THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ CỦA 3 MÔ HÌNH

1. **HOG Features**

function HOGFeatureKNN()

%% Load Image Information from AnimalData Directory

faceDatabase = imageSet('AnimalData','recursive');

%% Split Database into Training & Test Sets 4680

[training,test] = partition(faceDatabase,[0.8 0.2]);

%% Extract HOG Features for training set

% 26244 = extractHOGFeatures(read(training(i),j)). L?u y?

trainingFeatures = zeros(size(training,2)\*training(1).Count,26244);

TrainingfeatureCount = 1;

for i=1:size(training,2)

for j = 1:training(i).Count

trainingFeatures(TrainingfeatureCount,:) = extractHOGFeatures(read(training(i),j));

trainingLabel{TrainingfeatureCount} = training(i).Description;

TrainingfeatureCount = TrainingfeatureCount + 1;

end

AnimaltraingIndex{i} = training(i).Description;

end

%% Create 40 class classifier using fitcecoc

**% MdKNN = fitcknn(trainingFeatures,trainingLabel); % (1)**

**MdSVM = fitcecoc(trainingFeatures,trainingLabel); % (2)**

%% Extract HOG Features for test set

% X = extractHOGFeatures(read(training(i),j)). Luy y: X = 26244

testFeatures = zeros(size(test,2)\*test(1).Count,26244);

TestfeatureCount = 1;

for i=1:size(test,2)

for j = 1:test(i).Count

testFeatures(TestfeatureCount,:) = extractHOGFeatures(read(test(i),j));

testLabel{TestfeatureCount} = test(i).Description;

TestfeatureCount = TestfeatureCount + 1;

end

AnimaltestIndex{i} = test(i).Description;

end

%% Classify Test Images and count of number of sample

**% predictedLabels = predict(MdKNN,testFeatures); % (1)**

**predictedLabels = predict(MdSVM,testFeatures); % (2)**

lblPredicted = predictedLabels';

nResult = strcmp(testLabel,lblPredicted);

nCount = sum(nResult);

nTest = size(testLabel,2);

accuracy = nCount/nTest

fprintf('\n So luong mau test :%d\n', nCount)

fprintf('\n So luong mau dung :%d\n', nCount)

fprintf('\n Ti le mau dung :%d\n', accuracy)

%% Display five sample test images with their predicted labels.

animal = 6;

idx = [1 13 25 37 49];

figure;

for i = 1:numel(idx)

label = predictedLabels(idx(i));

lbl = label';

subplot(2,3,i);imshow(read(test(i),animal));

title(char(lbl));

end

end

***HOG Features và thuật toán SVM***

Sử dụng (1) thuật toán KNN và Predict

%% Create 40 class classifier using fitcecoc

**MdKNN = fitcknn(trainingFeatures,trainingLabel); % (1)**

%% Classify Test Images and count of number of sample

**predictedLabels = predict(MdKNN,testFeatures); % (1)**

|  |  |
| --- | --- |
| **Hiển thị 5 mẫu test** | **Kết quả** |
|  |  |

***HOG Features và thuật toán SVM***

Sử dụng (2) thuật toán SVM và Predict

%% Create 40 class classifier using fitcecoc

**MdSVM = fitcecoc(trainingFeatures,trainingLabel); % (2)**

%% Classify Test Images and count of number of sample

**predictedLabels = predict(MdSVM,testFeatures); % (2)**

|  |  |
| --- | --- |
| **Hiển thị 5 mẫu test** | **Kết quả** |
|  |  |

1. **BoW Features**

function Bag\_ofWords

%% Load Image Data Train

rootFolder = fullfile('DataTrain');

categories = {'Vit', 'Ga', 'Heo', 'Cho', 'Meo'};

imds = imageDatastore(fullfile(rootFolder, categories), 'LabelSource', 'foldernames');

tbl01 = countEachLabel(imds)

minSetCount = min(tbl01 {:, 2});

imds = splitEachLabel(imds, minSetCount, 'randomize');

tbl02 = countEachLabel(imds)

bag = bagOfFeatures(imds);

img = readimage(imds,1);

featureVector = encode(bag,img);

categoryClassifier = trainImageCategoryClassifier(imds, bag);

rootFolder = fullfile('DataTest');

categories = {'Vit', 'Ga', 'Heo', 'Cho', 'Meo'};

imds = imageDatastore(fullfile(rootFolder, categories), 'LabelSource', 'foldernames');

tbl01 = countEachLabel(imds)

confMatrixTest = evaluate(categoryClassifier, imds);

mean(diag(confMatrixTest));

end

|  |
| --- |
| **Kết quả** |
|  |

1. **AlexNet Features**

function AlexnetKnn()

%% Call libraly of convolutional neural network

convnet = alexnet;

convnet.Layers

%% Set up data\_training

%Load Image Data Train

rootFolderTrain = fullfile('trainingSet');

categories = {'Vit', 'Ga', 'Heo', 'Cho', 'Meo'};

imgeTrainAll = imageDatastore(fullfile(rootFolderTrain, categories), 'LabelSource', 'foldernames');

% Extract features from the training set images

featureLayer = 'fc7';

trainingFeatures = activations(convnet, imgeTrainAll, featureLayer);

trainingLabels = imgeTrainAll.Labels;

%% Train a model

**classifier = fitcknn(trainingFeatures,trainingLabels); % (1)**

**% classifier = fitcecoc(trainingFeatures,trainingLabels); % (2)**

%% Set up data\_test

%Load Image Data Test

rootFolderTest = fullfile('testSet');

categories = {'Vit', 'Ga', 'Heo', 'Cho', 'Meo'};

imgeTestAll = imageDatastore(fullfile(rootFolderTest, categories), 'LabelSource', 'foldernames');

% Extract features from the training set images

featureLayer = 'fc7';

testFeatures = activations(convnet, imgeTestAll, featureLayer);

testLabels = imgeTestAll.Labels;

%% Classify Test Images and count of number of sample

predictedLabels = predict(classifier,testFeatures);

nResult = (predictedLabels == testLabels);

nCount = sum(nResult);

accuracy = mean(predictedLabels == testLabels)

fprintf('\n So luong mau dung :%d\n', nCount)

fprintf('\n Ti le mau dung :%d\n', accuracy)

%% Display four sample test images with their predicted labels.

idx = [1 13 25 37 49];

figure

for i = 1:numel(idx)

subplot(2,3,i)

I = readimage(imgeTestAll,idx(i));

label = predictedLabels(idx(i));

imshow(I)

title(char(label))

end

end

***AlexNet Features và thuật toán KNN***

Sử dụng (1) thuật toán KNN và Predict

%% Train a model

**classifier = fitcknn(trainingFeatures,trainingLabels); % (1)**

**% classifier = fitcecoc(trainingFeatures,trainingLabels); % (2)**

|  |  |
| --- | --- |
| **Hiển thị 5 mẫu test** | **Kết quả** |
|  |  |

***AlexNet Features và thuật toán SVM***

Sử dụng (2) thuật toán SVM và Predict

%% Train a model

**% classifier = fitcknn(trainingFeatures,trainingLabels); % (1)**

**classifier = fitcecoc(trainingFeatures,trainingLabels); % (2)**

|  |  |
| --- | --- |
| **Hiển thị 5 mẫu test** | **Kết quả** |
|  |  |

**So sánh các features và thuật toán cho mô hình phân loại ảnh**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Features** | **HOG** | **HOG** | **BoW** | **AlexNet** | **AlexNet** |
| **Algorithm** | KNN | SVM | B- SVM | SVM | KNN |
| **Classifier** | 5 | 5 | 5 | 5 | 5 |
| **Data train** | 250 | 250 | 250 | 250 | 250 |
| **Data test** | 50 | 50 | 50 | 50 | 50 |
| **Accuracy** | 36% | 52% | 90% | 100% | 100% |
|  |  |  |  |  |  |

Bảng so sánh kết quả của mô hình

Trên đây là bảng thống kê kết quả đạt được, kết quả này chỉ mang tính chất tham khảo vì bộ dữ liệu thực nghiệm quá ít mẫu nên cũng không thể hiện hết được tính chính xác. Trong tương lai nếu có điều kiện về thiết bị nghiên cứu chúng tôi sẽ thử nghiệm trên bộ dữ liệu lớn hơn trên image và video.

V. KẾT LUẬN

Qua những tháng ngày ăn ngủ với Matlab và VRA, bản thân rút ra một số kinh nghiệm như sau:

Rút trích đặc trưng: đối với ảnh màu có độ phân giải ngày càng cao nên trong phần tùy biến các thông số chúng ta nên chọn cellsize lớn hơn.

Rút trích đặc trưng AlexNet Feature cho tỉ lệ vượt trội. Tuy nhiên gặp khó khăn trong vấn đề cài đặt thêm công cụ (Neural Network Toolbox Model for AlexNet Network) và áp dụng vào bộ dữ liệu mới do chúng ta tạo ra.

Thời gian trích xuất các features: cùng một bộ dữ liệu, HOGFeature chạy nhanh hơn (theo định tính). Tuy nhiên kết quả thì không mấy khả thi.

Thuật toán: k-NN, SVM thì thuật toán SVM cho kết quả chính xác hơn.

**Tài liệu tham khảo**

1. Bài giảng và thực hành Nhận dạng thị giác và ứng dụng của Ts. Lê Đình Duy và Ts. Nguyễn Tấn Trần Minh Khang
2. <https://www.mathworks.com/help/vision/ref/extracthogfeatures.html>
3. <https://www.mathworks.com/help/vision/ref/extractlbpfeatures.html>
4. <https://www.mathworks.com/help/vision/ref/bagoffeatures-class.html>
5. <https://www.mathworks.com/help/vision/ug/image-classification-with-bag-of-visual-words.html>
6. <https://www.mathworks.com/examples/matlab/community/22754-deep-learning-example-feature-extraction-using-alexnet-and-cifar-10-dataset>