**Lâm Hàn Vương**

**Bài Tập Thực Hành 3**

**Source Code Bài 1 và 2:**

function showImageAndLabelByIndex(data, index)

fprintf('Load data');

allImage = loadMNISTImages(data);

TotalImage = size(allImage,2);

figure;

if(index <= TotalImage)

img = allImage(:, index);

img = reshape(img,28,28)

index = num2str(index);

imshow(img);

title(index);

end

end

**Q1. Hiển thị ảnh dataset-train với thứ tự index – n (Tham số):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n = 1** | **n=500** | **n=5000** | **n=10000** | **n=59000** |
|  |  |  |  |  |

function showImageFromDataTrain(index)

showImageByIndex('train-images.idx3-ubyte', 'train-labels.idx1-ubyte',index)

end

**Q2. Hiển thị ảnh dataset-test với thứ tự index – n (tham số):**

|  |  |  |  |
| --- | --- | --- | --- |
| **n = 1** | **n=500** | **n=5000** | **n=9000** |
|  |  |  |  |

function showImageFromDataTest(index)

showImageByIndex('t10k-images.idx3-ubyte', 't10k-labels.idx1-ubyte',index)

end

**Source Code Q3 và Q4**

function statictis=statisticLables(dataLabel)

allLabels = loadMNISTLabels(dataLabel);

lstNumber = unique(allLabels);

statictis = [lstNumber,histc(allLabels(:),lstNumber)];

bar(statictis)

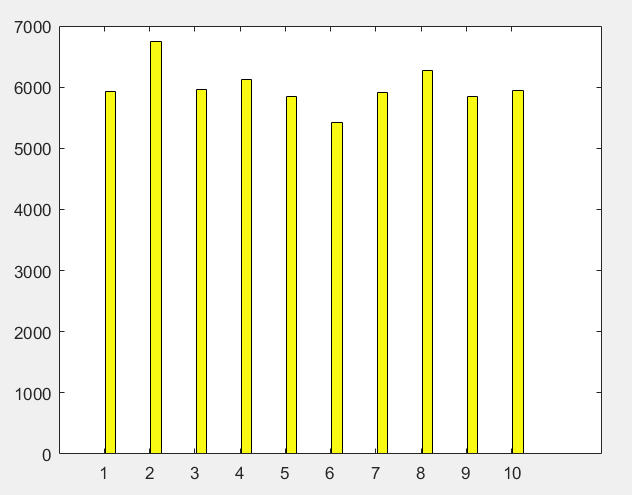
end

**Q3. Thống kê số lượng label trong tập huấn luyện train**

* Gồm 10 labels, được đánh dấu từ 0 đến 9:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Labels** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Số lượng** | 5923 | 6742 | 5958 | 6131 | 5842 | 5421 | 5918 | 6265 | 5851 | 5949 |

* **Lược đồ thể hiện:**

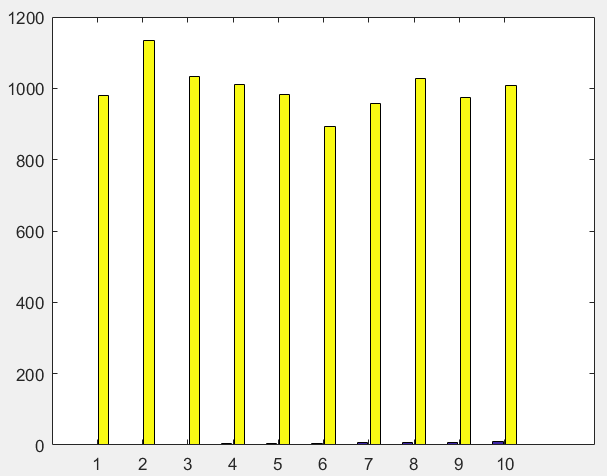


**Q4.Thống kê số lượng label trong tập test**

* Gồm 10 labels, được đánh dấu từ 0 đến 9:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Labels** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **Số lượng** | 980 | 1135 | 1032 | 1010 | 982 | 892 | 958 | 1028 | 974 | 1009 |

* Lược đồ thể hiện



**Q5. Trả về kết quả nhận dạng của ảnh trong tập test có thứ tự là n[1 ,10000]**

function strLabelImage = RecognitionDigits(index)

imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');

lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');

Mdl = fitcknn(imgTrainAll', lblTrainAll);

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');

lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');

mTestImgs = size(imgTestAll,2 );

nNumber =index;

imgTest = imgTestAll(:,nNumber);

lblPredictTest = predict(Mdl, imgTest');

lblImageTest =lblTestAll(nNumber);

figure;

img2D = reshape (imgTest,28,28);

imshow(img2D);

strLabelImage = 'Ban Dau ';

strLabelImage = [strLabelImage, num2str(lblTestAll(nNumber)),'.'];

strLabelImage = [strLabelImage, ' Du doan: '];

strLabelImage = [strLabelImage, num2str(lblPredictTest), '.'];

if(lblPredictTest == lblImageTest)

strLabelImage = [strLabelImage, 'Ket qua dung.'];

else

strLabelImage = [strLabelImage, 'ket qua sai '];

end

title(strLableImage)

end

|  |  |  |
| --- | --- | --- |
| **N=5** | **N = 500** | **N =900** |
|  |  |  |

**Q6. Hiển thị ảnh tương ứng với ảnh tìm được trong tập train**

function ShowSimilarImage(index)

imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');

lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');

Mdl = fitcknn(imgTrainAll', lblTrainAll);

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');

lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');

mTestImgs = size(imgTestAll,2 );

nNumber =index;

imageInput = imgTestAll(:, index);

imageInput = reshape(imageInput,[28,28]);

imgTest = imgTestAll(:,nNumber);

lblPredictTest = predict(Mdl, imgTest');

figure;

indexSimilarImage = find(lblTrainAll==lblPredictTest);

value = 1;

lstImage=zeros(0,0,'double');

for i = 1: 4

lst = imgTrainAll(:, indexSimilarImage(value));

lst = reshape(lst,[28,28]);

value= value +1;

for j = 1:10

img = imgTrainAll(:, indexSimilarImage(value));

img = reshape(img,[28,28]);

lst = cat(2,lst,img);

value =value +1;

end

lstImage = cat(1,lst,lstImage);

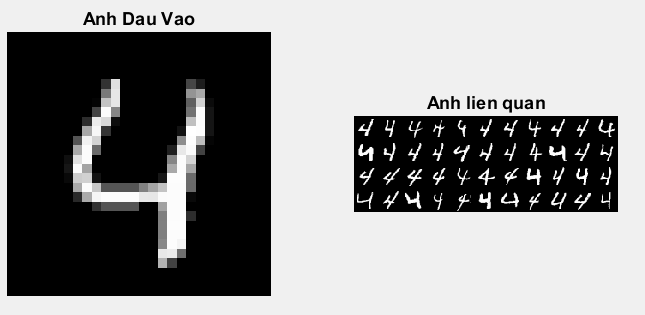
end

subplot(1,2,1), imshow(imageInput); title('Anh Dau Vao');

subplot(1,2,2), imshow(lstImage); title('Anh lien quan');

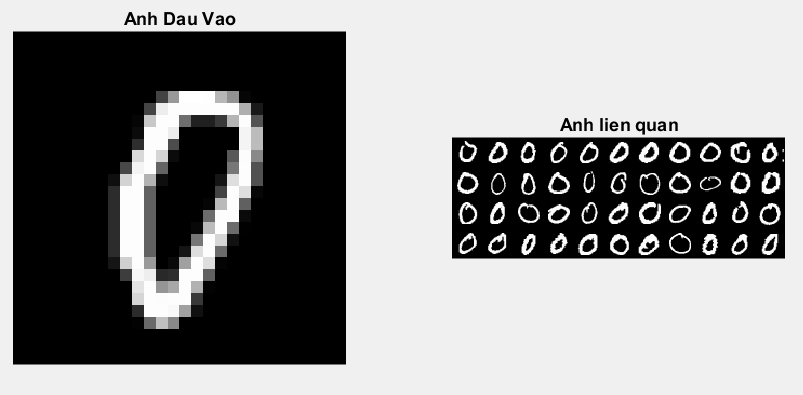
end

* Kết quả chạy
  + **Index = 5 – dataset test – predict: 4**



Ảnh trong data train liên quan

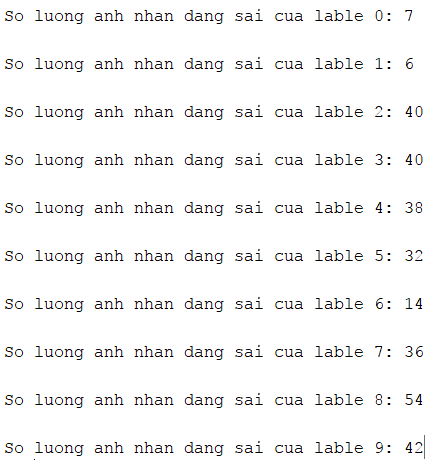
* + **Index = 499 từ dataset test- predict: 0**



Ảnh trong data train liên quan

**Q7. Đếm số lượng label ( n = 0… 9) bị nhận dạng sai**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N=0** | **N=1** | **N=2** | **N=3** | **N=4** | **N=5** | **N=6** | **N=7** | **N=8** | **N=9** |
| **7** | **6** | **40** | **40** | **38** | **32** | **14** | **36** | **54** | **42** |



function CountNumberImageFaile()

imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');

lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');

Mdl = fitcknn(imgTrainAll', lblTrainAll);

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');

lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');

fprintf('dang chay:\n');

lblPredictTest = predict(Mdl, imgTestAll');

for i = 0: 9

flag = (lblPredictTest == i);

mark = (lblTestAll == i);

nCount = (flag==mark & flag ~=0 & mark ~= 0);

nCount = sum(mark)- sum(nCount);

fprintf('\nSo luong anh nhan dang sai cua lable %d: %d\n',i,nCount);

end

end

**Q.7\* Lập bản confusion matrix**

function cfmatrix = confusionmatrix()

imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');

lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');

Mdl = fitcknn(imgTrainAll', lblTrainAll);

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');

lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');

fprintf('dang chay:\n');

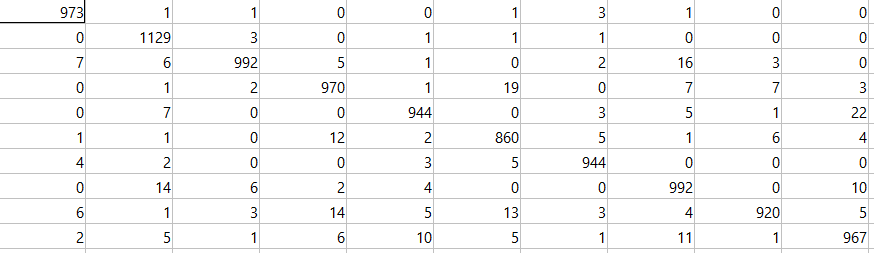
lblPredictTest = predict(Mdl, imgTestAll');

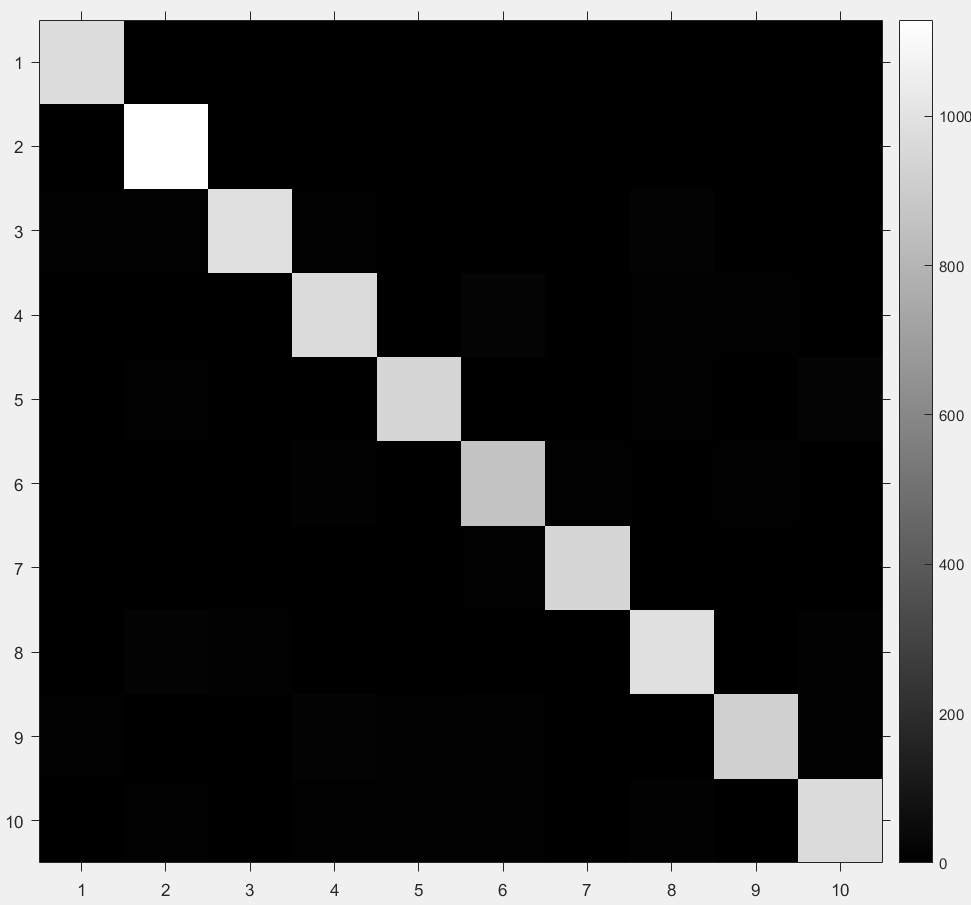
cfmatrix = confusionmat(lblTestAll, lblPredictTest);

imshow(cfmatrix, [], 'InitialMagnification', 10000);

end

* **Kết quả của Confusionmatrix:**



* Hiển thị:

**Q.8 Viết function tính độ chính xác của Knn với các tham số về neighbor và distance.**

function result=BT8()

imgTrainAll = loadMNISTImages('train-images.idx3-ubyte');

lblTrainAll = loadMNISTLabels('train-labels.idx1-ubyte');

imgTestAll = loadMNISTImages('t10k-images.idx3-ubyte');

lblTestAll = loadMNISTLabels('t10k-labels.idx1-ubyte');

fprintf('dang chay:\n');

distance = {'cosine','euclidean','minkowski','cosine','jaccard','chebychev','seuclidean'};

neighbor = [3 6 10 15 20];

confusionmatrix = zeros(length(neighbor),size(distance,2),'uint32');

a = 0;

for i = 1: length(neighbor)

for j = 1: length(distance)

fprintf('so hinh da xong: %d\n', a);

fprintf('NumNeighbors: %d | Distance: %s |', neighbor(i),distance{j});

Mdl = fitcknn(imgTrainAll', lblTrainAll,'NumNeighbors',neighbor(i),'Distance',distance{j});

lblPredictTest = predict(Mdl, imgTestAll');

confusionmatrix(i,j)= sum(lblPredictTest==lblTestAll);

a = a+1;

fprintf('gia tri: %d\n', confusionmatrix(i,j));

end

end

result=confusionmatrix;

imshow(confusionmatrix, [], 'InitialMagnification', 10000);

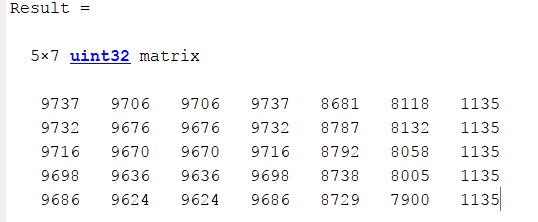
end

Kết quả chạy với các tham số:

* Cách tính distance: {'cosine','euclidean','minkowski','cosine','jaccard','chebychev','seuclidean'};
* Các neighbor: [3 6 10 15 20];

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Consine | Euclidean | Minkowski | Jaccard | Chebychev | seuclidean |
| 3 | 9737 | 9706 | 9706 | 8681 | 8118 | 1135 |
| 6 | 9732 | 9676 | 9676 | 8787 | 8132 | 1135 |
| 10 | 9716 | 9670 | 9670 | 8792 | 8058 | 1135 |
| 15 | 9698 | 9636 | 9636 | 8738 | 8005 | 1135 |
| 20 | 9686 | 9624 | 9624 | 8729 | 7900 | 1135 |

* Ma trận confusion Matrix:



* Thuật toán chạy tốt vói tham số đầu vào là: neighbor = 3 và cách tính khoảng cách là cosine