# Course

**ELEC4543**

# Project Title

**Assignment Instruction for Face Recognition**

# Name

**ZHU Zicong**

# UID

**3035142132**

# Submission Files

* **ELEC4543\_FinalReport.docx**
* **demo\_RBF.m & demo\_RBF\_40.m**
* **demo\_SVM.m & demo\_SVM\_40.m**
* **demo\_euclidean\_distance.m & demo\_euclidean\_distance\_40.m**
* **demo\_BP.m & demo\_BP\_40.m**
* **fcn\_convert\_to\_labels.m**

# Section A

[Introduction to Face Recognition & Abstraction]

Face Recognition is a popular research problem, which means making it possible for machine, mostly like computers to recognize people’s face. In fact, there are multiple solutions related to Artificial Neural Network (ANN) field nowadays. This individual project mainly focuses on Eigen-face approach related to different ANN models and some relevant influence factors. Four models will be talked about: Radial Basis Function Networks (RBF), Support Vector Machine (SVM), Euclidean Distance Method and Backpropagation ANN (BP).

Eigen-face approach is a very useful method for face recognition. Pictures of face are processed and then an Eigen-face vector is generated for each picture of face. Furthermore the Eigen-face vector will be considered as the representation of each face and then all the Eigen-face vectors will be used as the data sets for ANN model training and testing. Multiple factors will be evaluated in each implementation including learning rate, structure of network, number of used Eigen-faces and even the proportions of the trained and test images.

[RBF Model]

* Run demo\_RBF.m
* The output of Command Window:

1. >> run('C:\Users\user\Desktop\ELEC4543\Student\_ ELEC4543 Assignment\Student\_ ELEC4543 Assignment\demo\_RBF.m')
2. ImgPath =
3. C: \Users\ user\ Desktop\ ELEC4543\ Student\_ ELEC4543 Assignment\ Student\_ ELEC4543 Assignment\ att\_faces\_10\_people
4. k95 =
5. 18
6. ans =
7. The number of eigenvalues is 50
8. ans =
9. Keep the index from 18 to 50
10. ans =
11. The last 33 are kept
12. winner =
13. 5 5 5 5 5 5 5 5 5 5
14. ans =
15. The number of testing images is 50
16. ans =
17. The total error of RBF method is 0

* Run the RBF Model for 10 times
  + Error Table:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Times** | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th | 10th |
| **Error** | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

* + As we can see, the error of RBF model under this situation is 0 or 1, and according to this 10 times test, we can get the accuracy p = 98.6%

[SVM Model]

* Run demo\_SVM.m
* The output of Command Window:

1. >> run('C:\Users\user\Desktop\ELEC4543\Student\_ ELEC4543 Assignment\Student\_ ELEC4543 Assignment\demo\_SVM.m')
2. ImgPath =
3. C: \Users\ user\ Desktop\ ELEC4543\ Student\_ ELEC4543 Assignment\ Student\_ ELEC4543 Assignment\ att\_faces\_10\_people
4. ans =
5. The number of eigenvalues is 50
6. ans =
7. Keep the index from 18 to 50
8. ans =
9. The last 33 are kept
10. ans =
11. The number of testing images is 50
12. ans =
13. The total error of SVM method is 2

* Run the SVM Model for 10 times
  + Error Table:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Times** | 1st | 2nd | 3rd | 4th | 5th | 6th | 7th | 8th | 9th | 10th |
| **Error** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |

* + Therefore, the accuracy of SVM Model p = 96%, this is not influenced by iteration times.

# Section B

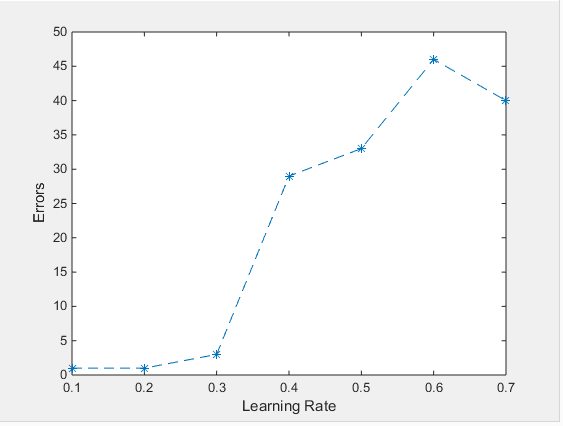
[Euclidean Distance Method]

* Run demo\_euclidean\_distance.m
  + Default maxiteration = 200
  + Default learnrate = 0.3
* The output of Command Window:

1. >> run('C:\Users\user\Desktop\ELEC4543\Student\_ ELEC4543 Assignment\Student\_ ELEC4543 Assignment\demo\_euclidean\_distance.m')
2. ImgPath =
3. C: \Users\ user\ Desktop\ ELEC4543\ Student\_ ELEC4543 Assignment\ Student\_ ELEC4543 Assignment\ att\_faces\_10\_people
4. ans =
5. The number of eigenvalues is 50
6. ans =
7. Keep the index from 18 to 50
8. ans =
9. The last 33 are kept
10. winner =
11. 5 5 4 6 6 6 5 5 4 4
12. ans =
13. The number of testing images is 50
14. ans =
15. The total error of Euclidean method is 3

* Under situation of maxiteration = 100 && learnrate = 0.3, the accuracy of Euclidean Distance Method p = 94%
* Influence of factor Learning Rate: (set maxiteration = 200)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Learning Rate** | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 |
| **Errors** | 1 | 1 | 3 | 29 | 33 | 46 | 40 |



# Section C

[Backpropagation ANN]

* Run demo\_BP.m
  + Default node number of first layer = 15
  + Default node number of second layer (output) = 10
  + Default maxiteration = 100
  + Default learnrate = 0.3
* Model Training Status (do 10 times)

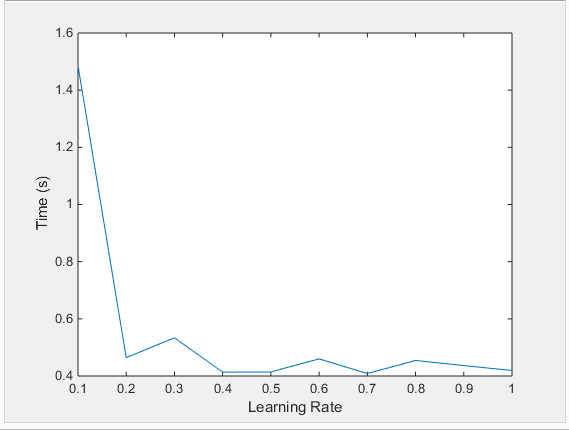
|  |  |  |  |
| --- | --- | --- | --- |
|  | Epoch | Time(s) | Error |
| 1st | 12 | 1.467411 | 0 |
| 2nd | 10 | 1.354476 | 1 |
| 3rd | 10 | 1.393208 | 0 |
| 4th | 10 | 1.379411 | 1 |
| 5th | 10 | 1.361914 | 5 |
| 6th | 10 | 1.367212 | 0 |
| 7th | 10 | 1.478751 | 1 |
| 8th | 10 | 1.361707 | 1 |
| 9th | 11 | 1.421458 | 1 |
| 10th | 11 | 1.408601 | 0 |
| Average | 10.4 | 1.3994 | 1 (p = 98%) |

* Influence by different Learning Rate
  + Default node number of first layer = 15
  + Default node number of second layer (output) = 10
  + Default maxiteration = 100
  + Run the following script:

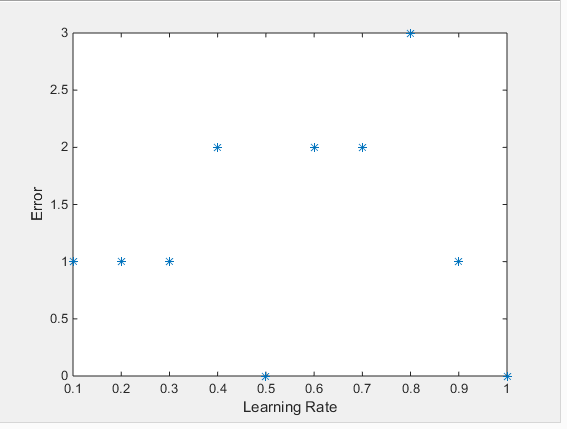
1. clear all close all result\_iter = [1 2 3 4 5 6 7 8 9 10;
2. 0 0 0 0 0 0 0 0 0 0 ;
3. 0 0 0 0 0 0 0 0 0 0 ]
4. digits(4)
5. for iter = 1: 10 learnrate = 0.1 \* iter;
6. run('demo\_BP.m');
7. result\_iter(2, iter) = vpa(time);
8. result\_iter(3, iter) = total\_errors;
9. end
   * Data Table:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Learning Rate | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
| Time (s) | 1.486 | 0.4642 | 0.5336 | 0.4136 | 0.4142 | 0.4600 | 0.4086 | 0.4543 | 0.4365 | 0.4192 |
| Error | 1 | 1 | 1 | 2 | 0 | 2 | 2 | 3 | 1 | 0 |

* + Graph
    - Learning Rate – Time (s)



* + - Learning Rate – Error

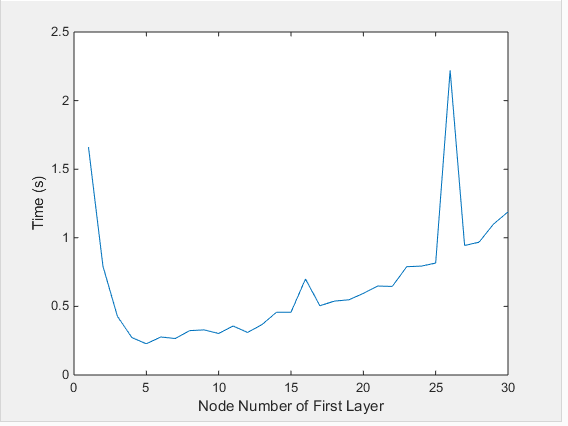


* Influence by different node number of first layer
  + Default node number of second layer (output) = 10
  + Default maxiteration = 100
  + Default learnrate = 0.3
  + Run the following script:

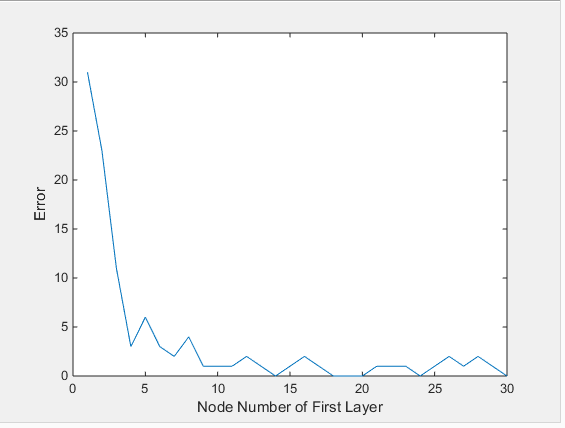
1. clear all
2. close all
3. result\_iter = [1: 30;zeros(1, 30);zeros(1, 30)]
4. digits(4)
5. for iter = 1: 30
6. first\_layer\_node\_num = iter;
7. run('demo\_BP.m');
8. result\_iter(2, iter) = vpa(time);
9. result\_iter(3, iter) = total\_errors;
10. end
    * Data Table (30 sets of data)



* + Graph:
    - Node Number of First Layer – Time (s)



* + - Node Number of First Layer – Error

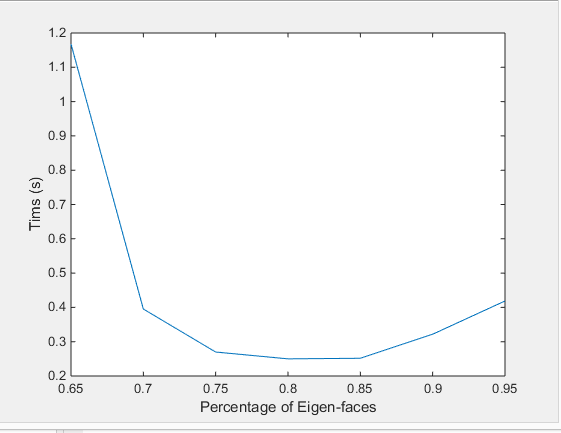


* Influence by different number of Eigen-faces
  + Default node number of first layer = 15
  + Default node number of second layer (output) = 10
  + Default maxiteration = 100
  + Default learnrate = 0.3
  + Run the following script:

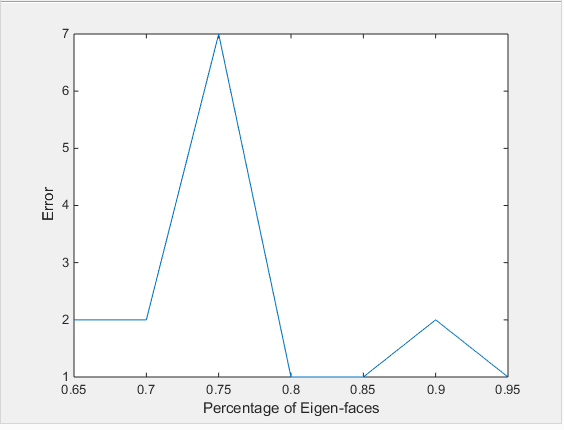
1. clear all close all result\_iter = [1: 7;0.65: 0.05: 0.95;zeros(1, 7);zeros(1, 7)]
2. digits(4)
3. for iter = 1: 7
4. p\_eigenfaces = result\_iter(2, iter);
5. run('demo\_BP.m');
6. result\_iter(3, iter) = vpa(time);
7. result\_iter(4, iter) = total\_errors;
8. end
   * Data Table:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Percentage of Eigen-faces | 0.65 | 0.7 | 0.75 | 0.8 | 0.85 | 0.90 | 0.95 |
| Time (s) | 1.155 | 0.296 | 0.2869 | 0.257 | 0.2521 | 0.3057 | 0.4348 |
| Error | 2 | 2 | 7 | 1 | 1 | 2 | 1 |

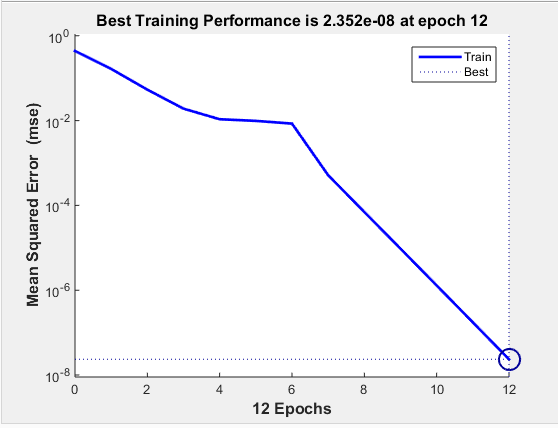
* + Graph
    - Percentage of Eigen-faces – Time



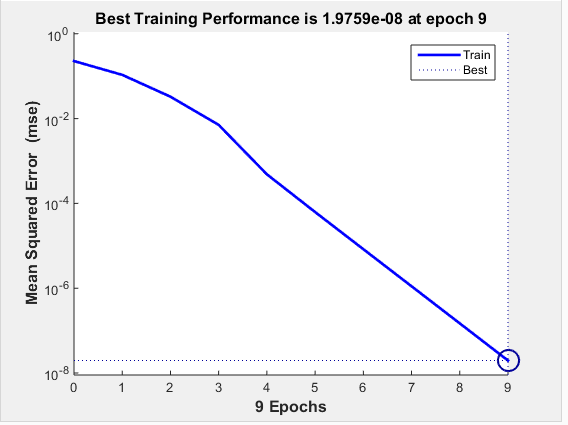
* + - Percentage of Eigen-faces – Error



* Optimal Setting of Parameter according to data above:
  + node number of first layer = 20
  + node number of second layer (output) = 10
  + maxiteration = 100
  + learnrate = 0.2
  + percentage of Eigen-faces = 0.85
* Performance Graph of the BP ANN:
  + Parameter Setting: Default Setting at first & Optimal Setting
  + Default Setting:
    - node number of first layer = 15
    - node number of second layer (output) = 10
    - maxiteration = 100
    - learnrate = 0.3
    - Percentage of Eigen-faces = 0.95
    - Output: time = 1.4869; error = 1
    - Performance Graph:



* + Optimal Setting:
    - node number of first layer = 20
    - node number of second layer (output) = 10
    - maxiteration = 100
    - learnrate = 0.2
    - percentage of Eigen-faces = 0.85
    - Output: time = 1.3078; error = 1
    - Performance Graph:



# Section D

[Comparisons]

* Run the four methods one by one with default parameter setting
* 10 TIMES for Average for each model
* Data Table (95% Eigen-faces are used):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Type** | **Time (s)** | **Performance** | **Error (in 50)** | **Accuracy** |
| RBF | 2.3895 | 0.4187 | 0.7 | 98.6% |
| SVM | 0.6365 | 1.5711 | 2 | 96% |
| Euclidean Distance | 0.0769 | 13.0039 | 3 | 94% |
| BP ANN | 1.3994 | 0.7146 | 1 | 98% |

* Data Table (65% Eigen-faces are used):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Type** | **Time (s)** | **Performance** | **Error (in 50)** | **Accuracy** |
| RBF | 2.3618 | 0.4234 | 3 | 94% |
| SVM | 0.5543 | 1.8041 | 1 | 98% |
| Euclidean Distance | 0.0724 | 13.8122 | 7 | 86% |
| BP ANN | 1.1979 | 0.8348 | 2.6 | 94.8% |

# Section E

[For 40 People]

* Run the four methods one by one with default parameter setting
* Data Table (95% Eigen-faces are used):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Type** | **Time (s)** | **Performance** | **Error (in 200)** | **Accuracy** |
| RBF | 121.7749 | 0.0082 | 149 | 25.5% |
| SVM | 1.1032 | 0.9065 | 35 | 82.5% |
| Euclidean Distance | 1.0336 | 0.9675 | 185 | 7.5% |
| BP ANN | 134.2904 | 0.0074 | 27 | 86.5% |

* For BP ANN:
  + Default node number of first layer = 40
  + Default node number of second layer (output) = 40
  + Default maxiteration = 100
  + Default learnrate = 0.3

# Section G

[Discussions & Conclusions]

* Following the Eigen-face approach, and as we know a double type in MatLab takes 64 bits (8 bytes) in storage
* N for number of people; P for number of Eigen-faces used

|  |  |  |  |
| --- | --- | --- | --- |
| Matrix Name | Number of Components | Storage Size (bytes) | Remark |
| template | 112\*92\*10\*N | 824320N | Including train and test sets |
| mean\_image | 10304 | 82432 |  |
| D | 5N\*5N | 200N2 |  |
| V | 5N\*5N | 200N2 |  |
| Ur | 10304\*50\*P | 4121600P |  |
| Dr | 50P\*50P | 20000P2 |  |

* Therefor:

Storage = 400N2 + 824320N + 20000P2 + 4121600P + 82432 bytes

* Conclusion:

|  |  |  |  |
| --- | --- | --- | --- |
|  | N = 10 | N = 40 | N = 10000 |
| P = INT(5N\*0.65) | 0.15GB | 0.85GB | 19844GB |
| P = INT(5N\*0.95) | 0.23GB | 1.43GB | 42253GB |