Assignment 5: Facial recognition with CNN
Harkirat Gill
301219830
CMPT 412

```
%% defining CNN parameters
% defining layers
layers = [imageInputLayer([size(img,1) size(img,2) 1])
    %middle layers
    convolution2dLayer(5,3,'Padding', 2, 'Stride',3)
    reluLayer
    maxPooling2dLayer(3,'Stride',3)
    %final layers
    fullyConnectedLayer(40)
    softmaxLayer
    classificationLayer()];

% options to train the network
options = trainingOptions('sgdm', ...
    'MiniBatchSize', 40, ...
    'InitialLearnRate', 1e-4, ...
    'MaxEpochs', 25, ...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropFactor', 0.875, ...
    'LearnRateDropPeriod', 12, ...
    'VerboseFrequency', 5);

% training the network
convnet = trainNetwork(trainFaceData,layers,options);
```

The above picture shows my layers and options used to train my convent. I have the following layers:

1) Input layer
2) Convolution layer
    a. Size = 5, Filters# = 3, Padding of 2 on each edge, and stride of 3
3) reluLayer activation layer
4) pooling layer
    a. pools max from [3,3] and has stride of 3
5) Fully connected layer to have 40 results
6) Softmaxlayer to convert the probabilities from the fully connected layer to convert to labels
7) Classification layer

The options used are as follows:

1) Minibatchsize = 40
    a. A mini-batch is a subset of the training set that is used to evaluate the gradient of the loss function and update the weights
    b. 40 seems to be a perfect size. 40 gives me 1 image per person to run the batch test. Default of 128 is too high (it would sacrifice 3 out of 5 images to test)
2) MaxEpochs = 25
    a. Accuracy starts to plateu around epoch 20
3) LearnRateSchedule
    a. 'LearnRateDropFactor', 0.875
    b. 'LearnRateDropPeriod', 12

c. Above combination gives the highest accuracy

Past tries of layers:

- I tried multiple layers of upto size 14(manual made) and alexnet. They both gave accuracy lower than 10%.
- Realized that training-options are as much important as layers
    - I have attached a picture from a test at the end of report to showcase what kind of results I was getting while using default training methods
- Changing 'MiniBatchSize' made the largest difference in acquiring higher accuracy
    - Jumped from below 35 to over 80

Tests run:

- Ran tests on images as they are and got 89% accuracy
- Ran test by reducing brightness to 90% of actual brightness and accuracy remained within same range
- Ran test by rotating the test images upto 7degs, accuracy went down to upto 75%

Thought on this method:

- This method is very useful to have detection/recognition
- It can be used for detecting (humans vs dogs vs cars etc)
- It can also be used for recognition (human 1 vs human 2 vs human 3)
- Finding what layers/options work the best is the hardest thing about this method

Accuracy from older tests

```
trainingNumFiles = 5;
rng(1)
[trainDigitData,testDigitData] = splitEachLabel(digitData, ...
    trainingNumFiles,'randomize');

layers = [imageInputLayer([size(img,1) size(img,2) 1])
          convolution2dLayer(5,2)
%           reluLayer
          maxPooling2dLayer(2,'Stride',1)

%         convolution2dLayer(3,1)
%         reluLayer
%
%         convolution2dLayer(3,10)
%         reluLayer
%         maxPooling2dLayer(2,'Stride',2)

          fullyConnectedLayer(40)
          softmaxLayer
          classificationLayer()];

options = trainingOptions('sgdm','MaxEpochs',50, ...
    'InitialLearnRate', 0.0001000);

convnet = trainNetwork(trainDigitData,layers,options);

YTest = classify(convnet,testDigitData);
```

```
ommand Window
Initializing image normalization.
|=========================================================================================|
|     Epoch    |   Iteration  | Time Elapsed |  Mini-batch  |  Mini-batch  | Base Learning|
|              |              |  (seconds)   |     Loss     |   Accuracy   |     Rate     |
|=========================================================================================|
|           1 |           1 |         0.67 |       5.7728 |        4.69% |     1.00e-04 |
|          50 |          50 |        31.02 |       0.0002 |      100.00% |     1.00e-04 |
|=========================================================================================|

accuracy =

    0.6800

>>
```

Latest Accuracy

```
main.m
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
    trainingNumFiles,'randomize');

layers = [imageInputLayer([size(img,1) size(img,2) 1])
    %middle layers
    convolution2dLayer(5,3,'Padding', 2, 'Stride',3)
    reluLayer
    maxPooling2dLayer(3,'Stride',3)
    %final layers
    fullyConnectedLayer(40)
    softmaxLayer
    classificationLayer()];

options = trainingOptions('sgdm', ...
    'MiniBatchSize', 40, ...
...%    'Momentum', .9, ...
    'InitialLearnRate', 1e-4, ...
    'MaxEpochs', 25, ...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropFactor', 0.88, ...
    'LearnRateDropPeriod', 8, ...
    'VerboseFrequency', 5);

convnet = trainNetwork(trainFaceData,layers,options);

YTest = classify(convnet,testFaceData);
TTest = testFaceData.Labels;

%%
% Calculate the accuracy.
```

```
ommand Window
|          20 |          100 |         10.98 |       0.0248 |      100.00% |     7.74e-05 |
|          21 |          105 |         11.64 |       0.0208 |      100.00% |     7.74e-05 |
|          22 |          110 |         12.26 |       0.0180 |      100.00% |     7.74e-05 |
|          23 |          115 |         12.82 |       0.0161 |      100.00% |     7.74e-05 |
|          24 |          120 |         13.35 |       0.0146 |      100.00% |     7.74e-05 |
|          25 |          125 |         13.89 |       0.0134 |      100.00% |     6.81e-05 |
|=========================================================================================|

accuracy =

    0.8900

>>
```

Name	Value
accuracy	0.8900
convnet	1x1 SeriesNetwork
faceData	1x1 ImageDatasto
faceDatasetP...	'orl_faces'
img	112x92 uint8
layers	7x1 Layer
options	1x1 TrainingOptio
testFaceData	1x1 ImageDatasto
trainFaceData	1x1 ImageDatasto
trainingNum...	5
TTest	200x1 categorical
YTest	200x1 categorical

Data from training CNN

Command Window

```
Training on single CPU.
Initializing image normalization.
|=========================================================================================|
|     Epoch     |   Iteration   | Time Elapsed |  Mini-batch  |  Mini-batch  | Base Learning|
|               |               |  (seconds)   |     Loss     |   Accuracy   |     Rate     |
|=========================================================================================|
|           1 |           1 |         0.16 |       3.9113 |        2.50% |     1.00e-04 |
|           1 |           5 |         0.60 |       3.7560 |        5.00% |     1.00e-04 |
|           2 |          10 |         1.19 |       3.7463 |        0.00% |     1.00e-04 |
|           3 |          15 |         1.77 |       3.7661 |        2.50% |     1.00e-04 |
|           4 |          20 |         2.39 |       3.7277 |        2.50% |     1.00e-04 |
|           5 |          25 |         2.95 |       3.6793 |        2.50% |     1.00e-04 |
|           6 |          30 |         3.51 |       3.6343 |        2.50% |     1.00e-04 |
|           7 |          35 |         4.08 |       3.5619 |        7.50% |     1.00e-04 |
|           8 |          40 |         4.69 |       3.4247 |       10.00% |     1.00e-04 |
|           9 |          45 |         5.31 |       3.1566 |       10.00% |     1.00e-04 |
|          10 |          50 |         5.95 |       2.6958 |       27.50% |     1.00e-04 |
|          11 |          55 |         6.52 |       2.0626 |       42.50% |     1.00e-04 |
|          12 |          60 |         7.09 |       1.3742 |       57.50% |     1.00e-04 |
|          13 |          65 |         7.66 |       0.8008 |       80.00% |     8.75e-05 |
|          14 |          70 |         8.24 |       0.4239 |       92.50% |     8.75e-05 |
|          15 |          75 |         8.84 |       0.2051 |       97.50% |     8.75e-05 |
|          16 |          80 |         9.60 |       0.1063 |      100.00% |     8.75e-05 |
|          17 |          85 |        10.33 |       0.0640 |      100.00% |     8.75e-05 |
|          18 |          90 |        10.92 |       0.0427 |      100.00% |     8.75e-05 |
|          19 |          95 |        11.48 |       0.0311 |      100.00% |     8.75e-05 |
|          20 |         100 |        12.07 |       0.0243 |      100.00% |     8.75e-05 |
|          21 |         105 |        12.63 |       0.0201 |      100.00% |     8.75e-05 |
|          22 |         110 |        13.20 |       0.0174 |      100.00% |     8.75e-05 |
|          23 |         115 |        13.76 |       0.0155 |      100.00% |     8.75e-05 |
|          24 |         120 |        14.32 |       0.0141 |      100.00% |     8.75e-05 |
|          25 |         125 |        14.88 |       0.0130 |      100.00% |     7.66e-05 |
|=========================================================================================|

accuracy =

    0.9100
```

All of code!

```
%% variables
trainingNumFiles = 5;
rng(1)
%% importing images for database & doing fft
faceDatasetPath = fullfile('orl_faces');
faceData = imageDatastore(faceDatasetPath,...
    'IncludeSubfolders',true, 'LabelSource', 'foldernames');

% read one image to get pixel size
img = readimage(faceData,1);

% splitting the testing and training data
[trainFaceData,testFaceData] = splitEachLabel(faceData, ...
    trainingNumFiles,'randomize');
```

```
%% defining CNN parameters
% defining layers
layers = [imageInputLayer([size(img,1) size(img,2) 1])
    %middle layers
    convolution2dLayer(5,3,'Padding', 2, 'Stride',3)
    reluLayer
    maxPooling2dLayer(3,'Stride',3)
    %final layers
    fullyConnectedLayer(40)
    softmaxLayer
    classificationLayer()];

% options to train the network
options = trainingOptions('sgdm', ...
    'MiniBatchSize', 40, ...
    'InitialLearnRate', 1e-4, ...
    'MaxEpochs', 25, ...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropFactor', 0.875, ...
    'LearnRateDropPeriod', 12, ...
    'VerboseFrequency', 5);

% training the network
convnet = trainNetwork(trainFaceData,layers,options);
```

```
%% classifying
YTest = classify(convnet,testFaceData);
TTest = testFaceData.Labels;

%% Calculate the accuracy.
accuracy = sum(YTest == TTest)/numel(TTest)
```