Patch-based Analysis Sparse-coding System (PASS_v1.0)

version 1.0

Authors: Jie Zhang jzhan313@asu.edu
Qingyang Li liqingyanghappy@gmail.com
Yalin Wang ylwang@asu.edu

%%

Pre-required:

Matlab 2012a and above

g++ 4.6 and above

%%%%%%%%%%%%%%%%%%%%%%%%%%%%Example%%%5%%%

%Here we use hippocampus jfeature.m as an example.

Step 1: select the vertex information from .m file and convert .m file to .txt file

```
>>matlab -nojvm

% convert_m_txt(fileLocation, lineStart, lineEnd);
% fileLocation = '/you/jfeature.m/folder/';
% lineStart is the first vertex id;
% lineEnd is the last vertex id;

>> cd /PASSS_v1.0/
>> convert_m_txt('/your local jfeature folder/jfeature/', 1, 30000);

% Here is an example folder in the package '/PASS_v1.0/example/'
% >> cd /PASSS_v1.0/
% >> convert_m_txt('/PASS_v1.0/example/', 1, 30000);
```

Step 2: read jfeature into Matlab

```
% if there are left jfeature and right jfeature, please use function twice, each time with 15000 features
% if there is only one jfeature file, please use function once

% data = readjfeat(fileLocation, vertexDim, featNum, subjectNum)
% fileLocation = '/you/jfeature_vertex.txt/folder/';
% vertexDim is the dimension of vertex;
% featNum is the number of feature number of mTBM/TBM (featNum = 3) or MMS (featNum = 4);
% subjectNum is the number of total subjects;
>> cd /PASS_v1.0/
>> MMSdata = readjfeat('/you/jfeature_vertex.txt/folder/', 30000, 4, 8);
% Here is an example folder in the package '/PASS_v1.0/example/'
% >> cd /PASS_v1.0/
% >> MMSdata = readjfeat('/PASS_v1.0/example/', 30000, 4, 8);
% if your jfeature has two parts: left_jfeature and right_jfeature,
% please use following commands:
%   MMSdata = readjfeat('/you/left_jfeature_vertex.txt/folder/', leftvertexDim, featNum, subjectNum);
```

```
%   MMSdata(leftvertexDim+1:leftvertexDim+rightvertexDim, :, :) = readjfeat('/you/right_jfeature_vertex.txt/folder/',
rightvertexDim, featNum, subjectNum);
% e.g., the hippocampus jfeature has left_jfeature.m and right_jfeature.m
%  the left vertex dimension is 15000, the right vertex dimension is 1500
%  the command will as follows:

%  >> MMSdata = readjfeat('/you/left_jfeature_vertex.txt/folder/', 15000, 4, 4;
%  >> MMSdata(15001:30000, :, :) = readjfeat('/you/right_jfeature_vertex.txt/folder/', 15000, 4, 4);
```

Step 3: reshape the jfeature data into original template shape

```
% [left, right]=reshape_original(data, featstart, featend, leftvertexNum)
% data is the jfeature.mat
% featstart is the start feature order (1, .., 4)
% featend is the end feature order (1, .., 4)
% leftvertexNum is the number of vertexes of left_jfeature
% e.g., the MMS data is consisting of Radial Distance (RD) and mTBM,
```

```
>> cd /PASS_v1.0/
>>[leftMMS, rightMMS] = reshape_original(MMSdata, 1, 4, 15000);
```

Step 4: select 10 times 10 patches on surface data and save as patchesfile.txt

```
% [patchSamples, patchNum] = select_patches(filename, SampleNum, Sample1, Sample2, featNum);
% filename = 'you_output_patches_filename.txt';
% SampleNum is the total number of subjects
% Sample1 is the left feature data
% Sample2 is the right feature data
% featNum is the dimension of your RD/mTBM/TBM/MMS feature on each vertex
% patchNum is the number of patches for each subject

>>[patchSamples, patchNum_MMS] = select_patches('MMS_patches.txt', 8, leftMMS, rightMMS, 4);
%>> patchNum = 1008 %show on the screen
%exit the Matlab
>>exit()
```

Step 5: learning dictionary and sparse codes by stochastic coordinate coding

```
%This part is implemented by C++.
%./run featureDim sampleDim SampleFileName DictionaryFileName FeatureFileName
%featureDim is the dimension of sparses feature (usually 5 times of input patch dimension)
%sampleDim is the dimension of input features
%SampleFileName = 'MMS_patches.txt' (the name of patch file learned from step 4)
%FeatureFileName = 'MMS_sparse.txt' (the name of the output)
%DictionaryFileName = 'MMS_dict.txt' (the name of the codebook of dictionary learning)
%e.g., for hippocampus MMS data, featureDim = 2000, sampleDim = 400

>>./run 2000 400 MMS_patches.txt MMS_sparse.txt MMS_dict.txt
```

Step 6: Max-pooling

```
%This part is implemented by C++.
%./MaxPooling featureDim batchSize FeatureFileName outputFile
% FeatureFileName = 'MMS_sparse.txt' (the name of the output from step 5)
% featureDim is the dimension of sparses feature (same as step 5)
% batchSize is the number of patches for each subject
% outputFile = 'MMS_features.txt' (the name of the features which we will use do the prediction)
```

>>./MaxPooling 2000 1008 MMS_sparse.txt MMS_features.txt

Step 7: Adaboost classification

```
>> matlab
% [Acc, Sen, Spe, ppv, npv, auc] = classficationExample(sampleNum, trainNum, inputdata, inputclass, name)
% this function will recall the function adaboost, evaluation and aucscore.
% Acc, Sen, Spe, ppv, npv, auc is a 2d vector. The mean and standard deviation of the cross-validation result.
% Also, the function will generate a roc curve figure save as name.fig
% sampleNum is the number of subjects
% trainNum = N is N-fold cross validation
% inputdata is the features learnt from step6
% inputclass is the label
% name = 'roc_figure_name.fig'

>> load('/PASS_v1.0/example/label.mat');
>> load MMS_features.txt
>> [Acc, Sen, Spe, ppv, npv, auc] = classExample(8, 2, MMS_features', label, 'MMS_ROC.fig')

% the screen will show you the results of Acc, Sen, Spe, ppv, npv, auc
```