```
%==========================================================================================
%Copyright (c) 2018 by Georgia Tech Research Corporation.
%All rights reserved.
%
%The files contain code and data associated with the paper titled
%"A Deep Learning Approach to Estimate Stress Distribution: A Fast and
%Accurate Surrogate of Finite Element Analysis".
%
%The paper is authored by Liang Liang, Minliang Liu, Caitlin Martin,
%and Wei Sun, and published at Journal of The Royal Society Interface, 2018.
%
%The file list: ShapeData.mat, StressData.mat, DLStress.py, im2patch.m,
%UnsupervisedLearning.m, ReadMeshFromVTKFile.m, ReadPolygonMeshFromVTKFile.m,
%WritePolygonMeshAsVTKFile.m, Visualization.m, TemplateMesh3D.vtk, TemplateMesh2D.vtk.
%Note: *.m and *.py files were converted to pdf files for documentation purpose.
%
%THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES,
%INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
%FOR A PARTICULAR PURPOSE.
%==========================================================================================
%%
function Result=UnsupervisedLearning(OutputDataFile, ShapeDataFile, StressDataFile,
IdxList_train, IdxList_test)
%%
Result=0;
load(ShapeDataFile)
load(StressDataFile)
IdxList_train=IdxList_train(:)';
IdxList_test=IdxList_test(:)';
%%
ShapeData_train=ShapeData(:,IdxList_train);
ShapeData_test=ShapeData(:,IdxList_test);
MeanShape=mean(ShapeData_train,2);
X=zeros(size(ShapeData_train));
for k=1:length(IdxList_train)
    X(:,k)=ShapeData_train(:,k)-MeanShape;
end
X=X/sqrt(length(IdxList_train));
[U, S, V]=svd(X);
Lambda=diag(S);
V123=sum(Lambda(1:3).^2)/sum(Lambda.^2);
PC={};
PC_count=3;
for k=1:PC_count
    PC{k}=U(:,k);
end
Proj=[];
for k=1:PC_count
    Proj(:,k)=U(:,k)/Lambda(k);
end
ShapeCode_train=zeros(PC_count,length(IdxList_train));
ShapeError_train=zeros(1,length(IdxList_train));
for k=1:length(IdxList_train)
```

```
    temp=ShapeData_train(:,k)-MeanShape;
    c=zeros(1,PC_count);
    for n=1:PC_count
        c(n)=sum(PC{n}(:).*temp(:))/Lambda(n);
    end
    ShapeCode_train(:,k)=c;
end
ShapeCode_test=zeros(PC_count,length(IdxList_test));
ShapeError_test=zeros(1,length(IdxList_test));
for k=1:length(IdxList_test)
    temp=ShapeData_test(:,k)-MeanShape;
    c=zeros(1,PC_count);
    for n=1:PC_count
        c(n)=sum(PC{n}(:).*temp(:))/Lambda(n);
    end
    ShapeCode_test(:,k)=c;
end
%%
StressData_train=StressData(:,:,IdxList_train);
StressData_test=StressData(:,:,IdxList_test);
S11_train=reshape(StressData_train(1,:,:), [5000, length(IdxList_train)]);
S22_train=reshape(StressData_train(2,:,:), [5000, length(IdxList_train)]);
S12_train=reshape(StressData_train(4,:,:), [5000, length(IdxList_train)]);
Sdata_train=zeros(50, 100, 3, length(IdxList_train));
for k=1:length(IdxList_train)
    Sdata_train(:,:,1,k)=reshape(S11_train(:,k), [50, 100]);
    Sdata_train(:,:,2,k)=reshape(S22_train(:,k), [50, 100]);
    Sdata_train(:,:,3,k)=reshape(S12_train(:,k), [50, 100]);
end
S11_test=reshape(StressData_test(1,:,:), [5000, length(IdxList_test)]);
S22_test=reshape(StressData_test(2,:,:), [5000, length(IdxList_test)]);
S12_test=reshape(StressData_test(4,:,:), [5000, length(IdxList_test)]);
Sdata_test=zeros(50, 100, 3, length(IdxList_test));
for k=1:length(IdxList_test)
    Sdata_test(:,:,1,k)=reshape(S11_test(:,k), [50, 100]);
    Sdata_test(:,:,2,k)=reshape(S22_test(:,k), [50, 100]);
    Sdata_test(:,:,3,k)=reshape(S12_test(:,k), [50, 100]);
end
%%
Data1=[];
for k=1:length(IdxList_train)
    tempPatch = im2patch(Sdata_train(:,:,:,k), [10, 20], [10, 20]);
    for n=1:size(tempPatch, 4)
        temp=tempPatch(:,:,:,n);
        Data1(:,end+1)=temp(:);
    end
end                    Whynot normalization?
Data1=single(Data1);   reduce storage
C1=Data1*Data1'/size(Data1, 1);   size: 600 X 600
[P1, L1, V1]=svd(C1);
L1=sqrt(diag(L1));
%% setup MatConvnet
run('Z:\matconvnet-1.0-beta24\matlab\vl_setupnn.m')
```

```
%%
W1=[];
Ps1=P1(:,1:256);
for k=1:size(Ps1, 2)
    W1(:,:,:,k)=reshape(Ps1(:,k), [10, 20, 3]);   10 x 20 x 3 x 256
end
Y1 = vl_nnconv(Sdata_train, W1, zeros(1,size(W1,4)),'stride', [10, 20]);   5 x 5 x 256 x train
Data2=[];          50 x 100 x 3 x train
for k=1:size(Y1,4)
    temp=Y1(:,:,:,k);
    Data2(:,end+1)=temp(:);
end
Data2=single(Data2);    6400 x train
C2=Data2*Data2'/size(Data2, 1);     6,400 x 6400
[P2, L2, V2]=svd(C2);
L2=sqrt(diag(L2));
W2=[];
Ps2=P2(:,1:64);
for k=1:size(Ps2, 2)
    W2(:,:,:,k)=reshape(Ps2(:,k), [size(Y1,1), size(Y1,2), size(Y1,3)]);   5 x 5 x 256 x 64
end
Y2 = vl_nnconv(Y1, W2, zeros(1,size(W2,4)),'stride', 1);   1 x 1 x 64 x train
Y1_t = vl_nnconv(Sdata_test, W1, zeros(1,size(W1,4)),'stride', [10, 20]);
Y2_t = vl_nnconv(Y1_t, W2, zeros(1,size(W2,4)),'stride', 1);
Y2n_train=zeros(size(Y2));
for k=1:64
    Y2n_train(:,:,k,:)=Y2(:,:,k,:)/L2(k);
end
Y2n_test=zeros(size(Y2_t));
for k=1:64
    Y2n_test(:,:,k,:)=Y2_t(:,:,k,:)/L2(k);
end
%%
Stress_train=[S11_train; S22_train; S12_train];
Stress_test=[S11_test; S22_test; S12_test];
try
delete(OutputDataFile)
catch
end
save(OutputDataFile, 'MeanShape', 'Proj', 'ShapeCode_train', 'ShapeCode_test', 'Stress_train',
'Stress_test', ...
    'Y2n_train', 'Y2n_test', 'L2', 'W1', 'W2', ...
    'OutputDataFile', 'ShapeDataFile', 'StressDataFile', 'IdxList_train', 'IdxList_test');
Result=1;
```