Copyright Notice

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.

User Instruction

Note: This document is associated with the paper "A Deep Learning Approach to Estimate Stress Distribution: A Fast and Accurate Surrogate of Finite Element Analysis" authored by: Liang Liang, Minliang Liu, Caitlin Martin, and Wei Sun, and published at Journal of The Royal Society Interface, 2018.

1. We have provided the following files

Data: ShapeData.mat, StressData.mat

Code of DL-model: DLStress.py, im2patch.m, UnsupervisedLearning.m,

Code for visualization: ReadMeshFromVTKFile.m, ReadPolygonMeshFromVTKFile.m,

WritePolygonMeshAsVTKFile.m, Visualization.m

Template meshes for visualization: TemplateMesh3D.vtk, TemplateMesh2D.vtk

Note: *.m and *.py files were converted to pdf files for documentation purpose. (e.g. *.m -> *.m.pdf). You need to convert them back if you want to run the code.

2. System Requirement

OS: Windows (64bit) 7 or 10

Hardware**:** Intel quad-core CPU, 32G RAM

3. Software Requirement

Anaconda: https://www.anaconda.com/download/; select the python 3.6 version

Keras: https://github.com/fchollet/keras

Keras can be install from Anaconda Cloud: https://anaconda.org/anaconda/keras

Tensorflow: https://www.tensorflow.org/

Tensorflow CPU version can be installed from Anaconda Cloud: https://anaconda.org/conda-forge/tensorflow

Matlab (at least 2016b): https://www.mathworks.com/products/matlab.html

MatConvNet: http://www.vlfeat.org/matconvnet/; version 1.0-beta24 (backward compatibility not guaranteed)

Paraview: https://www.paraview.org/download/; https://www.paraview.org/paraview-guide/

Spyder: https://spyder-ide.github.io/

Note: we used Tensorflow 1.1.0, Keras 2.0.4, and Python 3.5.

4. Install - Must follow the sequence:

4.1 Install Matlab and then MatConvNet

4.2 Install Anaconda

4.3 Install Tensorflow in Anaconda

First, open anaconda prompt (a cmd window), create an environment, and then install Tensorflow. See the instructions on https://www.tensorflow.org/install/install_windows to create an environment. Then install from tensorflow.org or Anaconda Cloud: https://anaconda.org/conda-forge/tensorflow; note: we used python 3.5

4.4 Install Keras in Anaconda

You must install Keras in the same environment that has Tensorflow. First, open anaconda prompt, then activate the environment in the cmd window, and type *conda install -c anaconda keras*

4.5 Install Spyder in Anaconda: https://anaconda.org/conda-forge/spyder

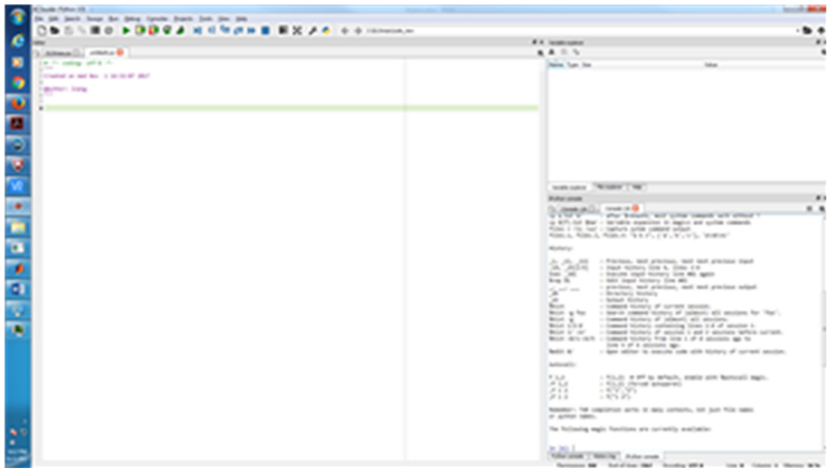
4.6 Setup Matlab engine for python

First, open anaconda prompt (a cmd window), activate the environment in the cmd window, and then follow the steps in https://www.mathworks.com/help/matlab/matlab_external/install-the-matlab-engine-for-python.html

4.7 Run examples of Keras in Spyder to make sure that software applications have been installed correctly.

5. Usage

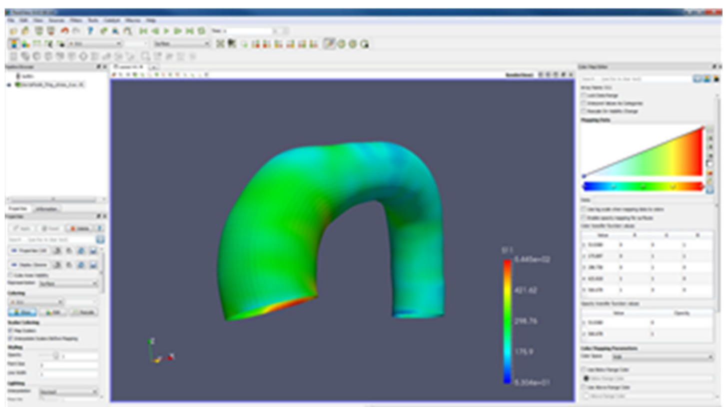
Activate the anaconda environment in a cmd window, and type spyder. Then you should see something like this. Spyder is a Python IDE. The current directory of Spyder is shown on top right.



Open DLStress.py in Spyder, and run the code. You need to change the current directory of Spyder so that it contains DLStress.py. Change the path of MatConvnet in UnsupervisedLearning.m

Once you save the result to mat files, open Visualization.m, and then convert the result to vtk files.

Open the vtk files in Paraview. You will see the ground-truth and predicted stress fields on 2D/3D meshes.



Do not be surprised if you get a slightly different result, which may be caused by software version differences, random initialization of neural network weights, and stochastic optimization.