Computational Modelling for Materials Engineering

Project: Artificial Neural Network (ANN) for Prediction of Homogenized Properties of Polysilicon Thin-Films

Background

The silicon lattice i.e., the periodic arrangement of the atoms in the crystal, is formed by diamond-like unit cells, that displays the face-centered cubic structure. Due to its crystalline structure, silicon displays an anisotropic behavior, that can be defined as a function of the crystallographic orientation. For polysilicon (aggregate of crystalline domains or grains), at small length scales, the grain morphology and lattice orientation have important effects on the *effective or homogenized elastic properties*: the effective elastic moduli computed over domains that are equal in terms of overall size, are described by statistical distributions that are functions of grain morphology and lattice orientations. Figure 1 shows illustrations of 2-dimensional digitally generated silicon aggregates featuring the same overall size, but all characterized by different values in their *effective elastic properties*. For each silicon grain, the simulated in-plane crystal lattice orientations are shown at each crystal centroid.

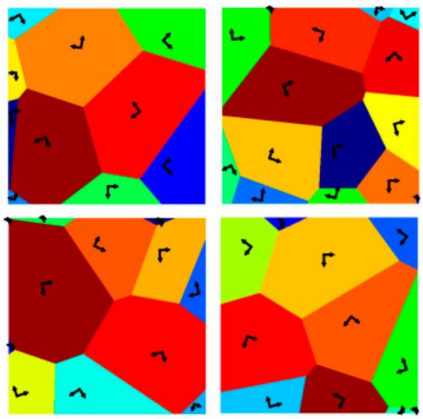

Figure 1. Four different 2 × 2 μm² digitally generated polysilicon realizations

These representations exemplify the morphology observed in epitaxially grown polysilicon thin films, transversally to the growth direction. For each grain the growth direction remains perpendicular to the substrate and aligned with one of the axes of elastic symmetry.

Fixing the values for the overall length and average grain size, it is possible to adopt a Monte Carlo analysis and carry out numerical homogenization of a large enough number of realizations, to obtain the corresponding statistical distributions describing the *effective elastic properties*. A more compact representation for these realizations can be achieved if the colored images (RGB) are converted to single channel (grayscale), where the gray level is correlated to the lattice

orientation (indicated previously by the rotated reference system depicted at each crystal centroid). In this way the polysilicon realizations can be handled as shown in Figure 2:

Figure 2. Compact representation of digitally generated polysilicon realizations

In this compact representation, a value of 0 is assigned to pixels corresponding to lattice rotation angles of 0° while a value of 255 is assigned for a rotation of 45°; any other angle outside [0°, 45°] is re-mapped to this range by subtracting 90° a suitable (integer) number of times. This is possible given the symmetry displayed by the elastic properties as a function of the in-plane lattice orientation in epitaxially grown silicon.

Instructions for the Project

- Develop and implement an ANN architecture capable to learn the mapping between the microstructure of polysilicon and the respective homogenized property of interest (to be chosen by you) as shown conceptually in Figure 3:

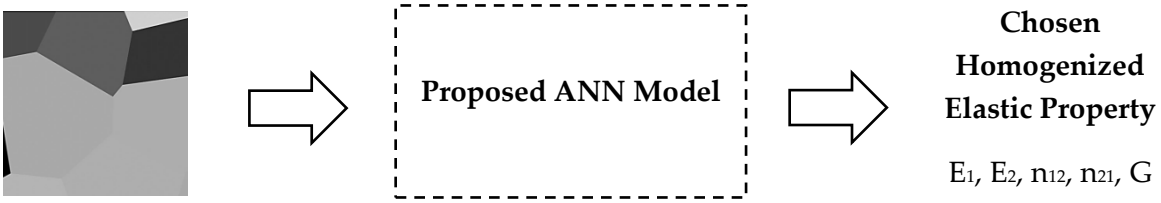

Figure 3: Scheme of the required implementation

- Download all files available at https://github.com/josepabloquesadamolina/CMME/tree/main/Project%20CMME and place them in a common folder in the default working directory of Jupyter Notebook.
- Extract the .zip files before attempting to run the Jupyter Notebook code titled base.ipynb.
- The file called base.ipynb is provided as the starting point for the project development and contains the required code for importing the input data (training, validation and test images) and corresponding target values or ground-truth data (E_1, E_2, n_{12}, n_{21}, G), which has been obtained via FE homogenization.
- In the section **STEP 2: DATA SPLITTING AND GROUND-TRUTH DATA STATISTICS** available within base.ipynb, three image datasets are assembled in the form

of independent arrays for their use during the implementation of the model. The shape of the arrays is shown below:

```
Training dataset:(2889, 128, 128, 1)
Validation dataset:(689, 128, 128, 1)
Test dataset:(300, 128, 128, 1)
```

Similarly, for each dataset, five independent arrays are available containing the target va lues of the homogenized elastic properties i.e., E_1, E_2, n_{12}, n_{21}, G. The statistical indicators (mean and standard deviation) of each property are also reported for each dataset.

- Start by carefully selecting the homogenized property to be predicted by the model. This is done in the section **STEP 3: SELECT THE ELASTIC CONSTANT TO PREDICT WITH THE ANN** available within base.ipynb
- base.ipynb suggest a possible structure to develop the project by showing a series of Markdown cells with the recommended contents within each section, followed by empty Code cells.
- Continue to create the model architecture. Select an architecture for the base model (to be used as the feature extractor). It can be one of the available architectures at https://keras.io/api/applications/ or can be a convolutional neural network architecture developed from scratch by the student. Report the main features of the proposed base model. Hint: Original papers for models imported from Keras applications are typically available in arXiv.org (Keras website provides the link that redirects to the respective paper).
- Add on top of the base model one or more layers for learning the specific mapping of the homogenized property. Hint: In case of importing the base model from https://keras.io/api/applications/, since the input data consists of single-channel images it the weights of the pre-trained base model can be discarded.
- Report the total number of parameters of the finally proposed architecture and draw a diagram to illustrate the structure of the implemented model including both base model and configuration of the layers added on top.
- When performing the training make sure to implement ModelCheckpoint callback function to monitor the "val_loss" and seek to minimize its value.
- Report the minimum validation loss and corresponding training loss as well as the number of epochs required to achieve these values.
- Assess if the model suffers from overfitting or underfitting by reporting the corresponding Loss vs Epoch plot and discussing the observed trend. Adjust correspondingly as required to decrease/eliminate these scenarios.
- After the training (and retraining to determine the optimal hyperparameters), evaluate the performance of the model by performing predictions of the selected effective property of interest. Report the mean and standard deviation values computed from the set of predictions, for each of the datasets: Training. Validation and Test Set.

- Compare the statistical indicators (mean and standard deviation) of the predictions with those computed based on the ground-truth data. Report the absolute percentage errors.
- Report and interpret the coefficient of determination between the set of predictions in each dataset and the corresponding ground-truth data.
- Generate a parity plot to summarize the results. In this plot, the y-axis must contain the predictions of the ANN model and the x-axis the corresponding ground-truth values (targets) of the chosen homogenized property of interest. Elaborate one parity plot for each dataset or one single parity plot for the three datasets making use of clarifying legends for the data. Analyze the performance of the model from the observed results. Hint: For an ideal model, the parity plot must reproduce the identity function $y = x$ in each dataset.
- Restrict the range of values of the x-axis and y-axis in the parity plot according to the range observed in the ground-truth of the selected homogenized elastic property.

General Note: All plots must include a title, axes description and units (when applicable). Legends must be added when multiple data is collected in a single plot.

Evaluating the effect of hyperparameters

Below there is a list of hyperparameters that should be considered and explored to achieve the best performing model:

1. Base model (feature extractor architecture).
2. Number of additional "Dense" layers added on top of the feature extractor
3. Number of units (neurons) in each of the "Dense" layers added on top of the feature extractor to produce the output prediction
4. Activation function to be used in the neurons of the "Dense" layers added on top of the feature extractor
5. Optimizer
6. Learning rate
7. Loss function
8. Number of epochs
9. Batch size

Report the results only for the best performing model (with the final set of hyperparameters chosen) but describe the impact of the different hyperparameters explored during the training of the ANN. The lack of this information will be penalized as it indicates a poor tuning/optimization of the model.

Submission and Evaluation

- A complete report compliant with the specified requirements must be submitted.
- The final Jupyter Notebook file (.ipynb extension) and the corresponding weights file (.h5 extension) of the best performing model must be submitted.
- The assessment of the model will account for the performance observed over a separate test dataset available only for the evaluator.