Printer Simulator User Manual

Prepared by Rahul Gupta

Contact email – [rgupta22@binghamton.edu](mailto:rgupta22@binghamton.edu)

Contents

List of Figures ……………………….…………….……… 3

List of Tables ……………………….…………….……….. 3

1. About the dataset ……………………………... 4 - 5
2. Printer simulator process flow ………............ 6
3. Software Requirements…………………….…. 7
4. Steps to use printer simulator.………….…....... 8
5. Result in a file…………….……………….…… 9
6. Guidelines about the code ...………….……… 10 - 11

List of Figures

1. PCB board MOM4 ………………………………….……………… 4
2. PCB board EKRA ..……………………………….………………… 5
3. Flowchart of printer simulator …………………….……………… 6
4. Flowchart of generalizing simulator using transfer learning …. 7
5. Input parameters ……………………………………….................... 9
6. Input full-scale SPI data ...………………………………................. 9
7. Input new job file SPI data ...………………………………............. 9
8. The output of simulation ….………………….….……................... 9
9. The Output of simulation with transfer learning ………………. 10
10. Output Excel sheet …….……………………………….................... 11
11. Output with Transfer Learning Excel sheet ……………………... 11
12. Output file name …….…………………………………................... 11
13. Input feature for which you want simulation ………………….. 12
14. Decomposition and reconstruction code for noise …………….. 12
15. Random noise generation code ...........................………………... 12
16. Prediction code.……………………………………………………. 13
17. Transfer learning code for new job file ..……………………… 14

List of Tables

1. MOM4 pads dimension ……………………...…………………… 5
2. About the dataset
   * Collected from SPI consist of:
     + PadID, Size X, Size Y, Volume (%), OffsetX(um), OffsetY(um), PCB ID, Printing Speed, Printing Pressure, Separation Speed, Cleaning Type, Cleaning Age, Direction, Pos X, Pos Y, Rotation, AR, ASR
     + AR- Area ratio
     + ASR- Aspect ratio
   * 125 printer settings (5 printing pressure, 5 printing speed, 5 separation speed)
     + Printing pressure – 60, 65, 70, 75, 80
     + Printing speed – 25, 30, 35, 40, 45
     + Separation speed – 3, 3.5, 4, 5.5, 7
   * 5 boards printed for each setting, since used cleaning cycle length of 5 boards
   * Clean stencil before each setting changing with sequence (Wet-Dry-Dry)
   * The board used for simulation is MOM4

A green background with blue squares

Description automatically generated

*Figure1.* *PCB board MOM4*

* + - MOM4 board has pad dimensions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0402 | | 0603 | | 1005 | |
|  | x(mm) | y(mm) | x(mm) | y(mm) | x(mm) | y(mm) | |
| Pad dimension | 0.22 | 0.22 | 0.33 | 0.33 | 0.56 | 0.56 | |

*Table1.* *MOM4 pads dimension*

* + Dataset collected from SPI for Transfer learning to adapt for different board type and printer type consist of:
    - PadID, Size X, Size Y, Volume (%), PCB ID, Printing Speed, Printing Pressure, Separation Speed, Cleaning Age, Direction, Pos X, Pos Y, AR, ASR
    - 8 printer settings (PCB) (4 printing pressure, 2 printing speed, 4 separation speed)
      * Printing pressure – 60, 65, 70, 75
      * Printing speed – 25, 40
      * Separation speed – 3.5, 4, 4.5, 5.5
    - The board used for validation of simulation when used with transfer learning is EKRA printed with EKRA printer.

A green and gold printed circuit board

Description automatically generated

*Figure2.* *PCB board EKRA*

1. Printer simulator process flow

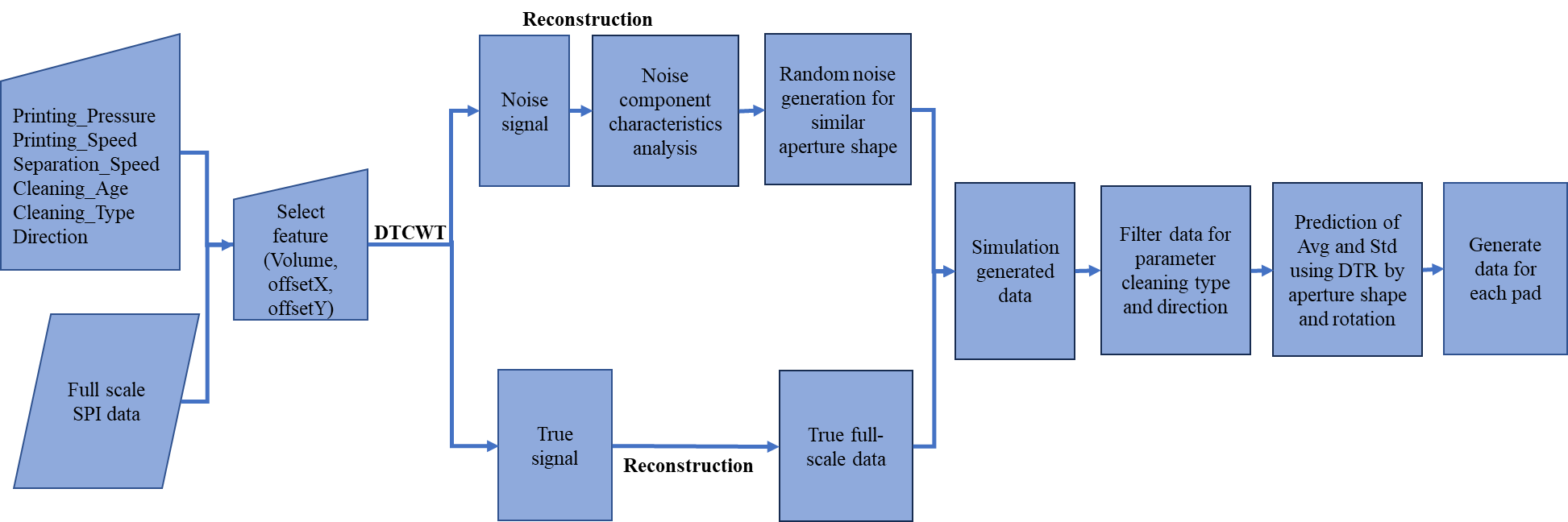


Figure3. Flowchart of printer simulator

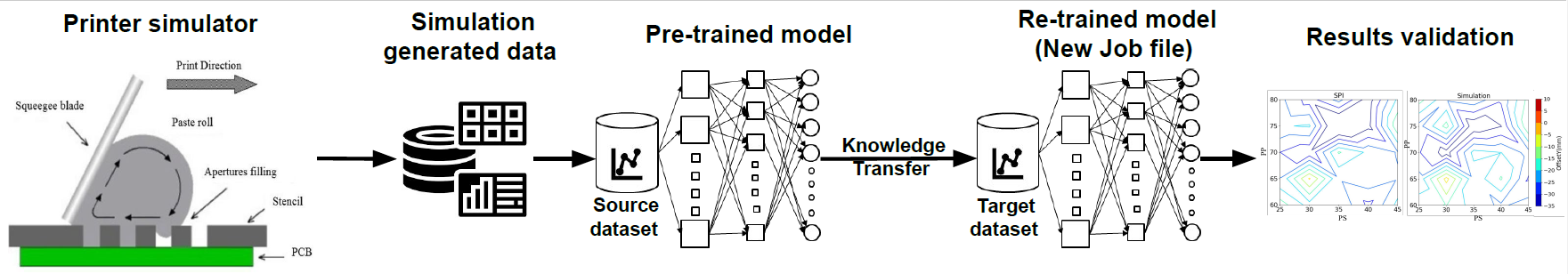
**

Figure4. Flowchart of generalizing simulator using transfer learning

1. Software Requirements
   * Software
     + Python (Version 3.9 or higher)
     + Four libraries are required
       - * pip install pywavelets
         * pip install numpy
         * pip install pandas
         * pip install scikit-learn
         * pip install tensorflow
2. Steps to use printer simulator
   * Open python file
   * Input

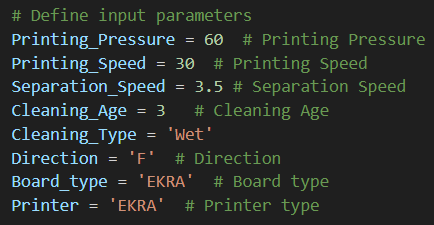


Figure5. Input parameters

* Cleaning type= [Wet, Dry]
* Direction= [F, B]
  + Input SPI data files with 624 boards information

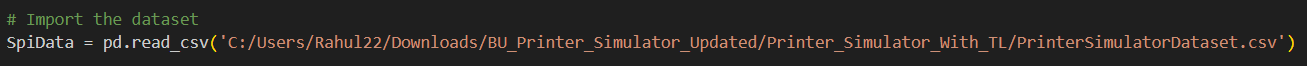


Figure6. Input full-scale SPI data

* + Input new training data files for retraining transfer learning model with 8 boards information



Figure7. Input new job file SPI data

* + Run the printer simulator code
  + Output will be generated

A picture containing text, font, menu, screenshot

Description automatically generated

Figure8. The Output of simulation

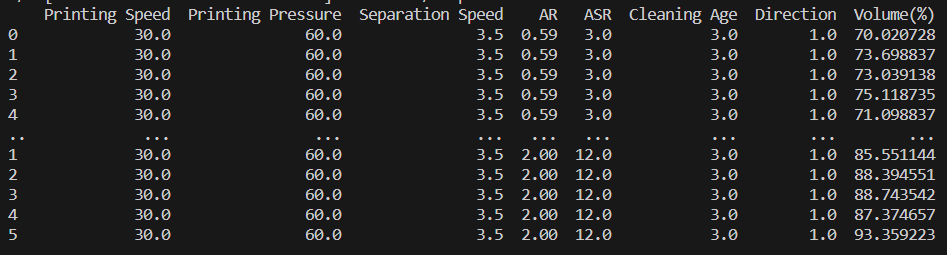


Figure9. The Output of simulation with transfer learning

1. Result in a file
   * Results are stored in an Excel file (Simulator Output.xlsx) as below. And it’s downloaded to the local system.

A screenshot of a computer

Description automatically generated

Figure10. Output Excel sheet

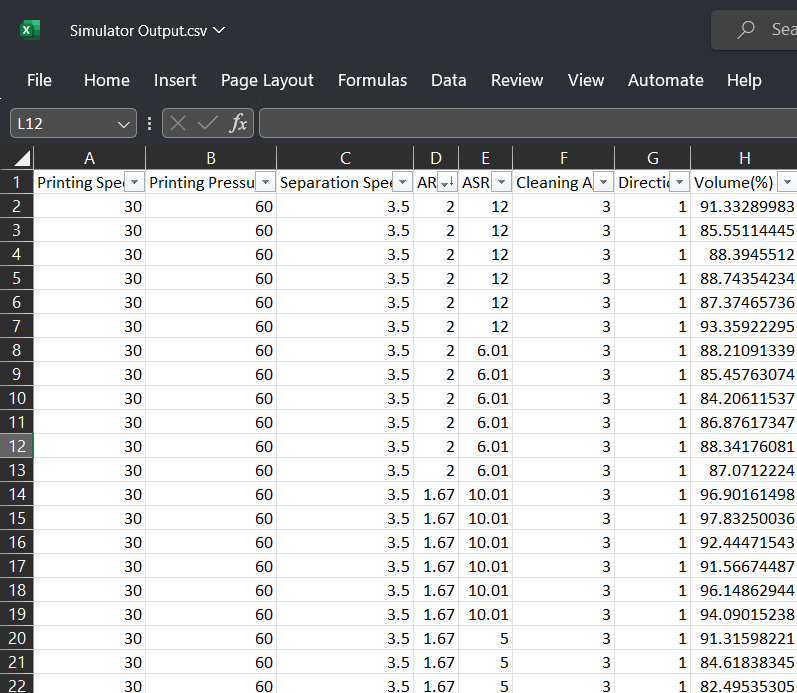


Figure11. Output with Transfer Learning Excel sheet

* + We can change this file name from the code



Figure12. Output file name

1. Guidelines about the code
   * Use Input in 3D
     + As the level of decomposition is 2, we should use even number of PCB, as in our case, it’s better to use 624



Figure13. Input feature for which you want simulation

* + DTCWT is used to get noise from the real printer. Below is the transformation of data into noise and true signal

A computer screen with colorful text

Description automatically generated A black screen with colorful text

Description automatically generated

Figure14. Decomposition and reconstruction code for noise

* + Random noise was generated for each similar pad group and aggregation with true volume to get simulated volume

A screen shot of a computer code

Description automatically generated

Figure15. Random noise generation code

* + If board type and printer used is same as input data used for simulator
    - Decision tree used for prediction for input parameters which generalize simulator
    - X – Input feature (Printing Speed, Printing Pressure, Separation Speed, Cleaning Age, AR, ASR)
    - Y – Output feature (Volume (%) / OffsetX(um) / OffsetY(um))

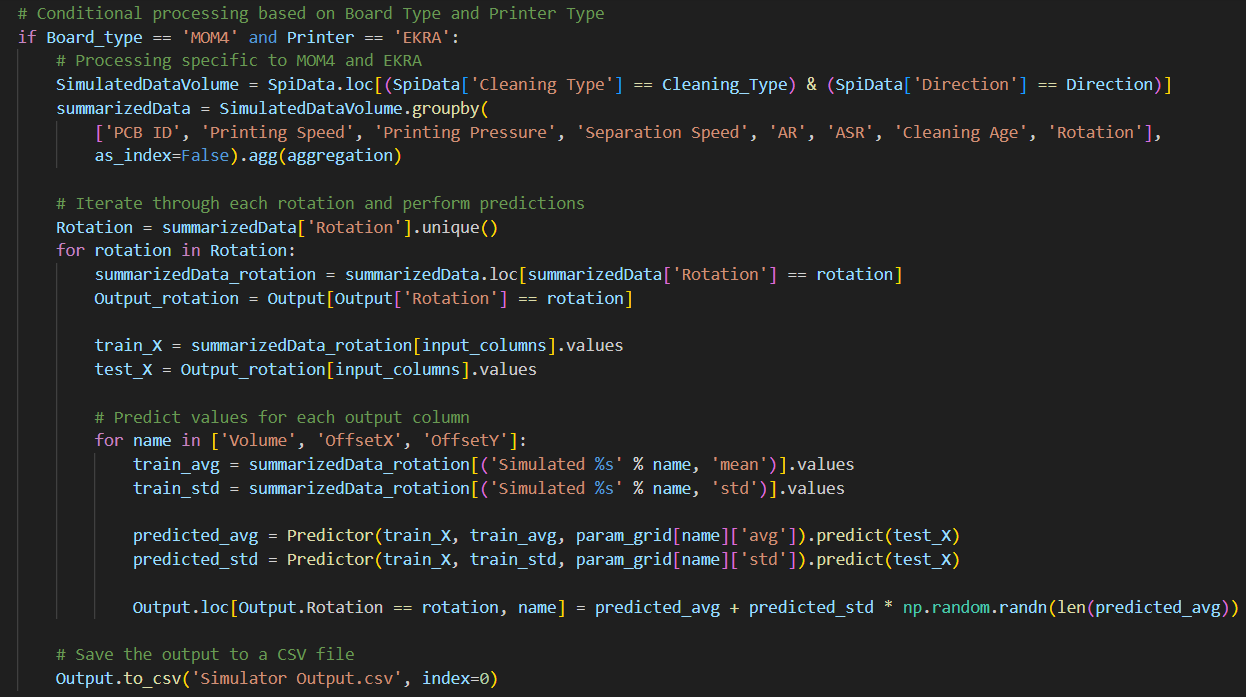


Figure16. Prediction code

* + If board type and printer type used is different from input data used for simulator: used transfer learning with neural networks
    - Pre-training: The model is initially trained (pre-trained) on a simulated dataset to learn general patterns
    - Fine-tuning: Subsequently, the pre-trained model is fine-tuned on a smaller, new job file dataset. This approach leverages the knowledge gained during pre-training to improve prediction accuracy on the new job file dataset

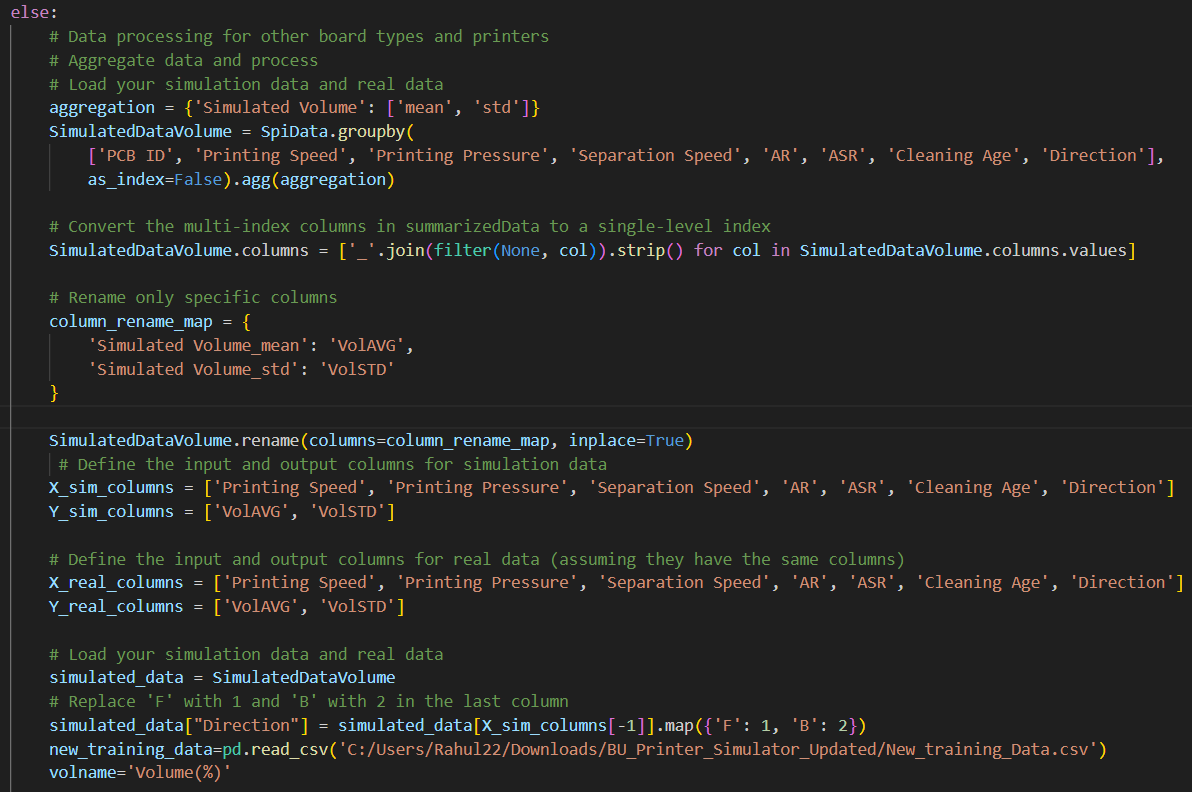
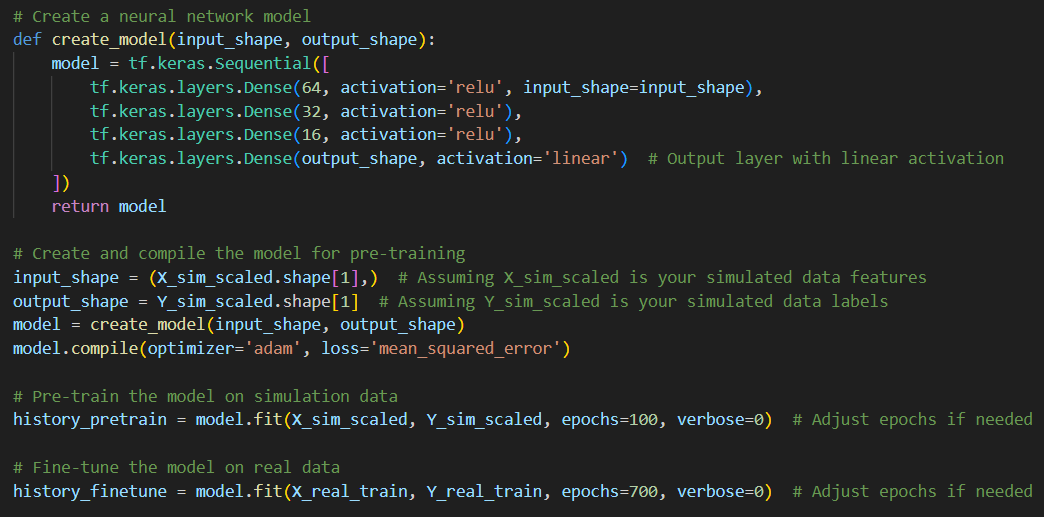


Figure17. Transfer Learning Code for new Job File

* + Noise level is controlled by the threshold of DTCWT, if required more variation increase the threshold and vice versa.
  + To know more about the Dual-Tree Complex Wavelet Transform (DTCWT) use the reference given below
    - I. W. Selesnick, R. G. Baraniuk and N. C. Kingsbury, "The dual-tree complex wavelet transform," in IEEE Signal Processing Magazine, vol. 22, no. 6, pp. 123-151, Nov. 2005, doi: 10.1109/MSP.2005.1550194.