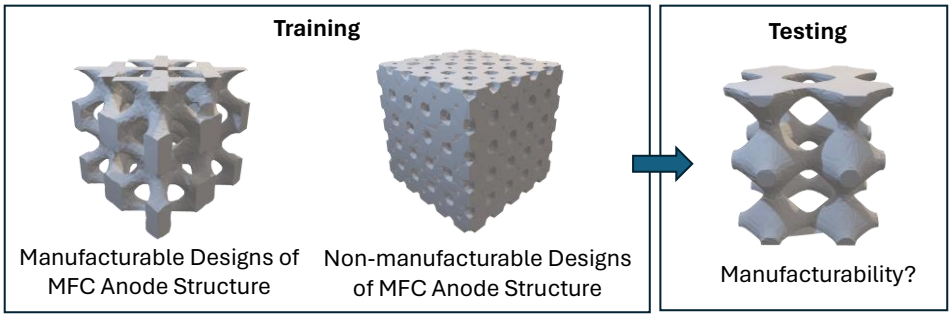
— Call for Participation —

Manufacturing AI Competition

Sponsored by Virginia Tech Academy of Data Science and
INFORMS Quality, Statistics, and Reliability Section
2024 INFORMS Annual Meeting (October 20 – 23, 2024)
Seattle, Washington, USA

Problem Statement

A manufacturing designer generates feasible designs and infeasible designs of Microbial Fuel Cell Anode Structure [ref??]. The design variables, generated 3D geometry data in STL files, and the manufacturability are provided, refer to the appendix for dataset structure. Please use ALL samples to train an AI model to classify the manufacturability indicator.



There are no limitations on AI models to be used but methodological innovation is ~~highlight~~ highly encouraged. Please use the 3D data from STL files to predict the binary manufacturablility~~e~~ response. Please also discuss your feature engineering and the modeling strategies to process the 3D data. Data can be downloaded here [link??] after user registration. Sample code and the environment information can be found in the appendix. Some 3D data pre-processing package can be found in the appendix. To reference the data source, please use [ref??]. Data reuse and redistribution is not allowed without written consent.

Submission Procedure

This is a team-based competition. Each team should have a maximum of 4 members from academia or industry. Students' participation is highly encouraged. Python code with required formats, the trained, serialized model object (see the detailed instructions in the appendix), and the result summary (up to 10 slides) should be submitted here [link??] online. Multiple submissions will be allowed for performance tests online, but only the last submission from the team will be used for competition. No email communications will be needed.

Commented [IL1]: Do we disclose evaluation metrics that will be used?

Commented [IL2]: We should nevertheless have a support email established if participants face any challenges?

Evaluation Procedure

The performance of the algorithms will be automatically evaluated on an undisclosed testing dataset generated from the same design simulation engine with the same parameter settings. Three finalists will be selected based on the performance of the algorithms and will be required to present the methodology and the results in an in-person session in INFORMS annual meeting 2024. One winning team will be selected by judges and all finalists will be recognized in the INFORMS-QSR business meeting. Finalists may be invited to participate in future activities, such as industrial engagement and research exchange, held by Virginia Tech Academy of Data Science.

Commented [IL3]: Is there support provided for finalists to attend INFORMs? Do we expect any exceptions for teams that cannot travel?

Important Dates

- Deadline of the submission online: August 19, 2024
- Notification of finalists: September 2, 2024
- Presentations as a session in INFORMS annual meeting: October 20 – 23, 2024

Organizers

Dr. Tom Woteki, Virgnia Tech, drwo@vt.edu
Dr. Xiaoyu Chen, University at Buffalo, xchen325@buffalo.edu
Dr. Ran Jin, Virginia Tech, jran5@vt.edu
Dr. Ismini Lourentzou, University of Illinois Urbana-Champaign, lourent2@illinois.edu
Dr. Hongyue Sun, University of Georgia, hongyuesun@uga.edu
Technical Support: Mr. Premith Chilukuri, Virginia Tech, cpremithkumar@vt.edu

Acknowledgement

This competition is partially supported by NSF project XXXX and XXXX.

Appendix

Programming ~~language~~Language:
ONLY Python3 programming language will be accepted by the online evaluation platform. The operating system is Ubuntu 18.04.6 LTS (Bionic Beaver).

Deep ~~learning~~ Learning ~~frameworks~~Frameworks:
We will support TensorFlow==latest, PyTorch==latest. Only CPU version will be used for evaluation.

If your submission depends on any other Python Modules, please also submit a requirements.txt document to specify the modules and the corresponding versions you would install. The platform will install these automatically. In the requirements.txt, please ~~doesn't~~ do not forget to include the libraries (along with corresponding versions) used for 3D data processing, transformation, and loading.

The test module in script should contain the relevant 3D data formatting, transformation, and loading logic which was used during the model training. Please also include the code for loading the trained model. This test script will be used to evaluate your model performance, so make sure the test script is fool proof.

Note: Please include comments wherever it is necessary. If you are planning to transform the given 3D data into a different representation for training the model, same data transformation logic should be implemented in the test script part of template.py

Warning: installing incompatible modules may crash the virtual environment.

Python ~~script~~ Script ~~format~~Format:
Please submit your Python module with one entrance script following the attached template.py template.

Dataset Format:
Training Data: A total of 1,000 designs with 70% feasible ones and 30% infeasible ones. Design variables and generated 3D geometry data in .STL format are provided. Folder and file structure is as follows:

```
./Training Data/
    ./Feasible_Designs/
        1.stl
        2.st ........
    ./Infeasible_Designs/
        35.stl
        42.stl ......
    Training_MetaData.csv
```

Test Data: To format your test script accordingly, the structure of the test data is provided. Save your model predictions in the "Save_Predictions.csv" file.

./Test Data/
    ./Test_designs/
        100.stl
        200.stl .....
    Save_Predictions.csv

Format of "Training_MetaData.csv":

Sample ID	STL FileName	X1	X2	X3	X4	X5	X6	Feasibility Class
1	305.stl	D	2	0.1	0.1	0	0	0
2	2.stl	B	1	0.8	0.1	1.576	0	1

Input: STL FileName (Main Predictor, encouraged to use this 3D data predictor only), X1, X2, X3, X4, X5, X6 (Design Variables: secondary Predictors, encouraged not to use them, but can be used to analyze the feasibility).
Output: Feasibility Class (Binary class)

Format of "Save_Predictions.csv": A skeleton place holder code logic should be implemented in the test script logic so that when the test directory path is given the code should be able to **load the model** and **navigate** to the test data directory and then make **predictions** on all .stl files present in the directory and save the prediction output to "Save_Predictions.csv". As the test data is not provided to you, you will have to implement the mentioned logic and during the evaluation phase we will run the test script to evaluate the model performance.

STL FileName	Predicted Class
713.stl	0
405.stl	1

Reference material to help you process the STL files.

- Please review the "numpy-stl" python package to process, access, modify, or save the .STL files. Sample codes related to "numpy-stl" can be found in this Link.
- Please refer to Open3D, which is another modern library for 3D data processing. Using this library, you can render, process, modify, and save the STL data, or transform them into voxel or point-cloud representations. Link
- The stl-to-voxel python package helps you to turn STL files into voxels, images, and videos.