# Data Agent for Dataset Design and Curation for Anomaly Detection

**Yinan Wang**
Rensselaer Polytechnic Institute

# Motivation & Outline

- ML models that monitor and control physical systems (manufacturing, accelerators) are only as reliable as the data they are trained on — and the data they see in deployment.

- This talk: three questions along the data lifecycle —
  1. **Trust:** can a deployed model recognize data it was never trained on? → **WOOD (OOD detection)**
  2. **Design:** with scarce labels and distribution mismatch, which data to select? → **ADs (active data selection)**
  3. **Automate:** can an agent design and curate datasets autonomously? → **Data Agent (preliminary investigation)**

**Trust the data**
*WOOD*

→

**Design the dataset**
*ADs*

→

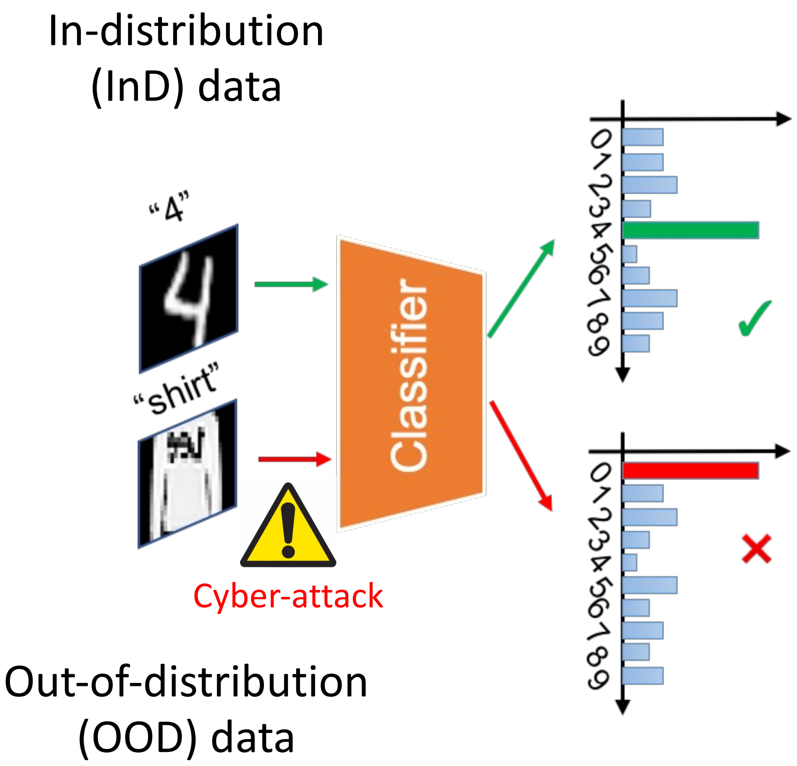
**Automate the loop**
*Data Agent*

# WOOD: Wasserstein-based Out-of-Distribution Detection

- **Yinan Wang**, Wenbo Sun, Jionghua (Judy) Jin, Zhenyu (James) Kong, Xiaowei Yue, 2024, "WOOD: Wasserstein-based Out-of-Distribution Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

- *Code is available at https://github.com/wyn430/WOOD*

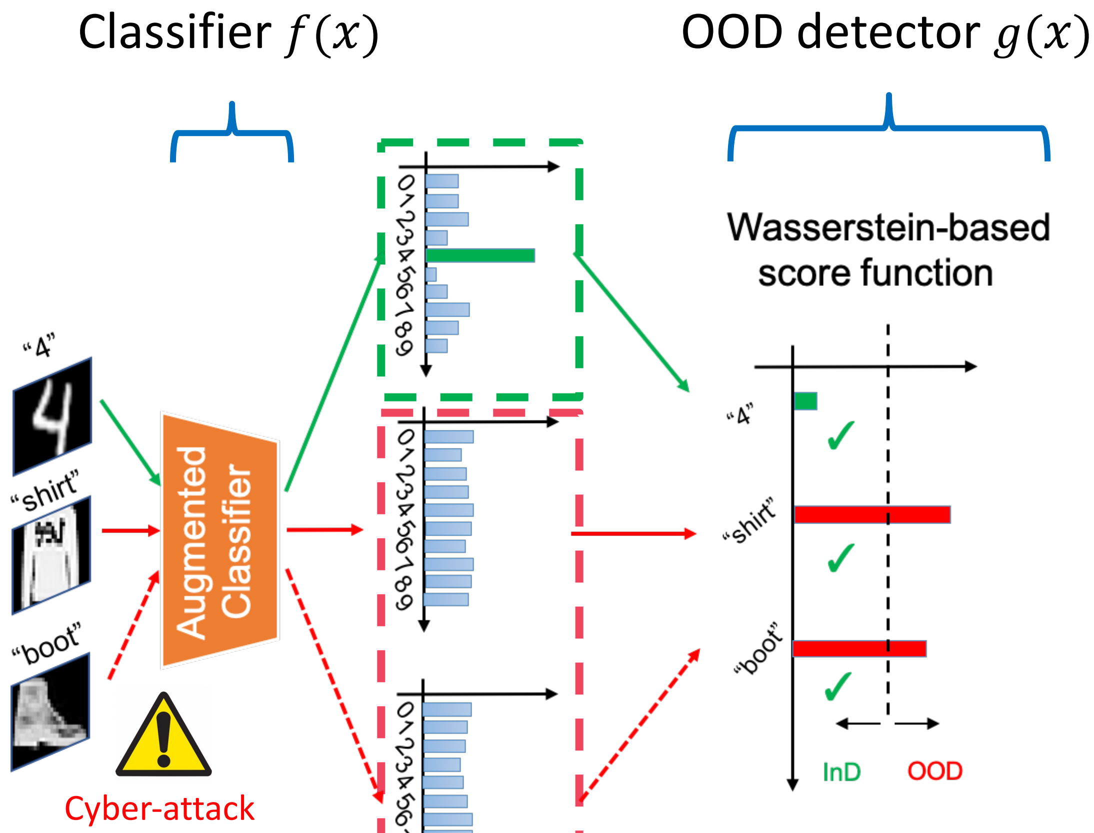
# Motivations




- In Deep Neural Networks (DNNs), the training and test data are possibly from different distributions.
- Classifier tends to overconfidently predict OOD data with class labels of InD data.
- OOD data threatens the reliability of classifiers in practice.
- **Goal:** Strengthen the classifier with the ability to detect OOD samples.

# Methodology: Overview

- Wasserstein-based OOD Detection (WOOD)




WD is adapted to train the $f(x)$ (1) correctly classify InD samples; (2) keep the OOD samples away from any InD classes.

The score function in $g(x)$ is designed by using WD to evaluate the dissimilarities between InD and OOD samples (outputs from the classifier are discrete distributions).

# Experiment Setting

- Datasets

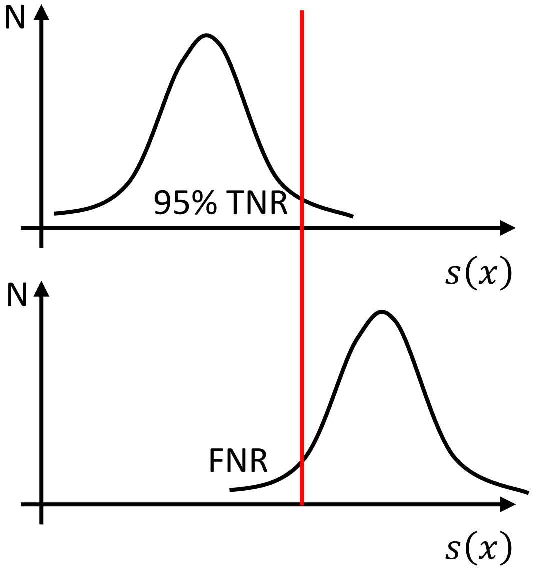
Table 1.1 Basic Information of Dataset

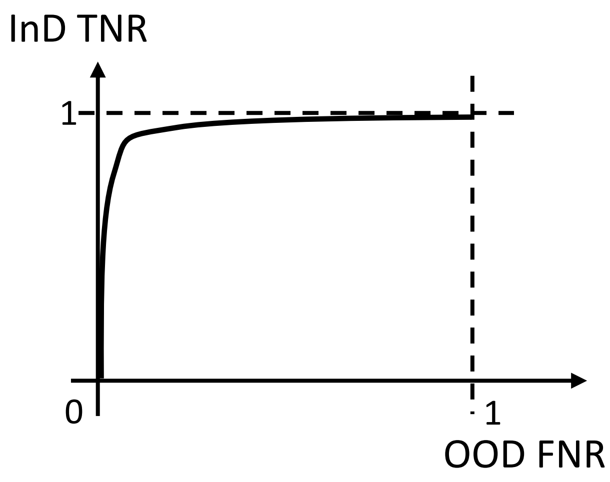
| Datasets | Number of Class | Image Dimension | Number of Training | Number of Testing |
|---|---|---|---|---|
| MNIST | 10 | $28 \times 28$ | 60000 | 10000 |
| FashionMNIST | 10 | $28 \times 28$ | 60000 | 10000 |
| CIFAR-10 | 10 | $3 \times 32 \times 32$ | 50000 | 10000 |
| CIFAR-100 | 100 | $3 \times 32 \times 32$ | 50000 | 10000 |
| SVHN | 10 | $3 \times 32 \times 32$ | 73257 | 26032 |
| TinyImageNet-r | 200 | $3 \times 32 \times 32$ | 100000 | 10000 |
| TinyImageNet-c | 200 | $3 \times 32 \times 32$ | 100000 | 10000 |
| Gaussian | N/A | $3 \times 32 \times 32$ | 10000 | 10000 |
| Uniform | N/A | $3 \times 32 \times 32$ | 10000 | 10000 |

- Evaluation Metrics

$$g(x) = \begin{cases} 1, \text{if } x \in \mathcal{X}_{\mathrm{OOD}} \\ 0, \text{if } x \in \mathcal{X}_{\mathrm{InD}} \end{cases}$$

False Negative Rate
(95% True Negative Rate)




AUROC

# Experiment Results: Evaluation Metrics

Table 1.2 Experiment Results of the Proposed WOOD Framework and Baseline Methods

| InD | OOD | FNR (95% TNR) ↓ | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | ODIN [8] | Maha [10] | ODIN* [9] | Energy [20] | 1D [15] | WOOD-binary | WOOD-dynamic |
| MNIST | FashionMNIST | 0.0589 | < 0.00001 | 0.0008 | 0.0005 | 0.0157 | < 0.00001 | < 0.00001 |
| FashionMNIST | MNIST | 0.0287 | 0.1657 | 0.1045 | 0.0010 | 0.2428 | < 0.00001 | < 0.00001 |
| CIFAR-10 | SVHN | 0.1693 | 0.1927 | 0.1017 | 0.0030 | 0.1765 | 0.0005 | 0.0011 |
| | TinyImageNet-r | 0.0430 | 0.3637 | 0.2479 | 0.5055 | 0.4269 | 0.0095 | 0.0019 |
| | TinyImageNet-c | 0.0001 | 0.3709 | 0.2568 | 0.0275 | 0.2128 | < 0.00001 | 0.0002 |
| | Gaussian | < 0.00001 | < 0.00001 | < 0.00001 | 0.0005 | 0.0088 | < 0.00001 | < 0.00001 |
| | Uniform | < 0.00001 | < 0.00001 | < 0.00001 | < 0.00001 | 0.0047 | < 0.00001 | < 0.00001 |
| CIFAR-100 | SVHN | 0.4973 | 0.4082 | 0.1723 | 0.0235 | 0.5683 | 0.0020 | 0.0020 |
| | TinyImageNet-r | 0.2524 | 0.3158 | 0.1818 | 0.5740 | 0.6962 | 0.0036 | 0.0058 |
| | TinyImageNet-c | 0.2534 | 0.3209 | 0.1846 | 0.0975 | 0.6702 | 0.0042 | 0.0002 |
| | Gaussian | < 0.00001 | < 0.00001 | < 0.00001 | 0.0010 | 0.3299 | < 0.00001 | < 0.00001 |
| | Uniform | < 0.00001 | < 0.00001 | < 0.00001 | 0.0010 | 0.4028 | < 0.00001 | < 0.00001 |

**Note:** The proposed WOOD consistently outperforms baseline methods and receives outstanding performance in all test cases.

# ADs: Active Data-selection for Data Quality Assurance in Data-sharing over Advanced Manufacturing Systems

- Zhao, Y., Li, Y., Liu, C., & Wang, Y. (2025). ADs: Active Data-selection for Data Quality Assurance in Data-sharing over Advanced Manufacturing Systems. *IEEE Transactions on Automation Science and Engineering*.

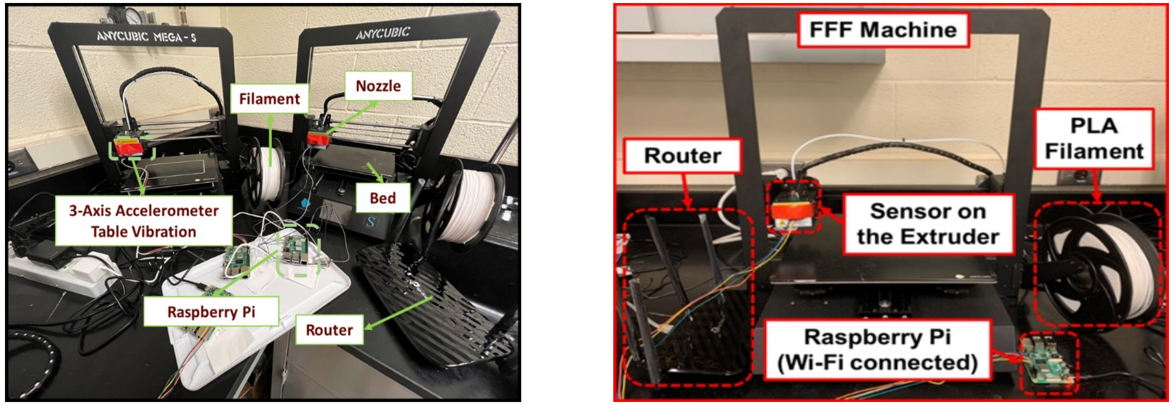
- *Code is available at https://github.com/zhaoy23/ADs*

# Motivation

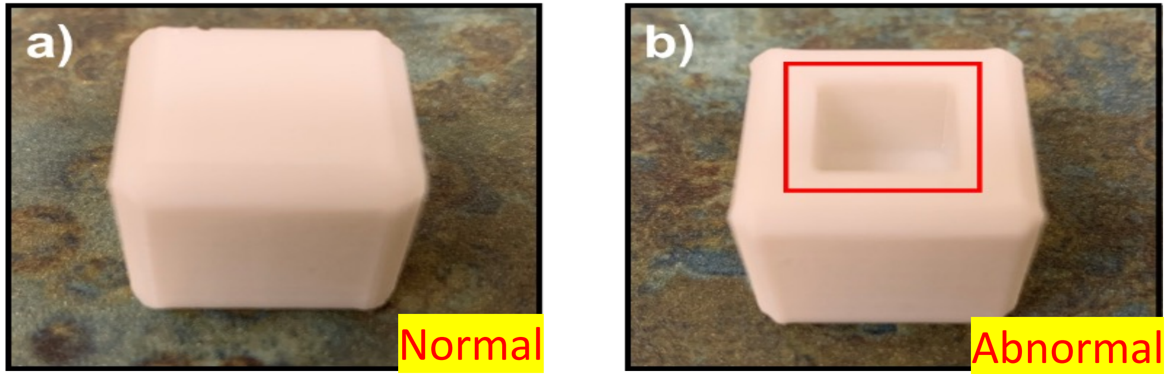
- Data collection needs **extensive time costs and investments** in the manufacturing system, and data scarcity commonly exists.

- Data-sharing among multiple machines with similar functionality has been widely enabled, **but distribution mismatch inevitably exists**

- **Goal:** Propose an intelligent data-selection framework to ensure the quality of the shared data

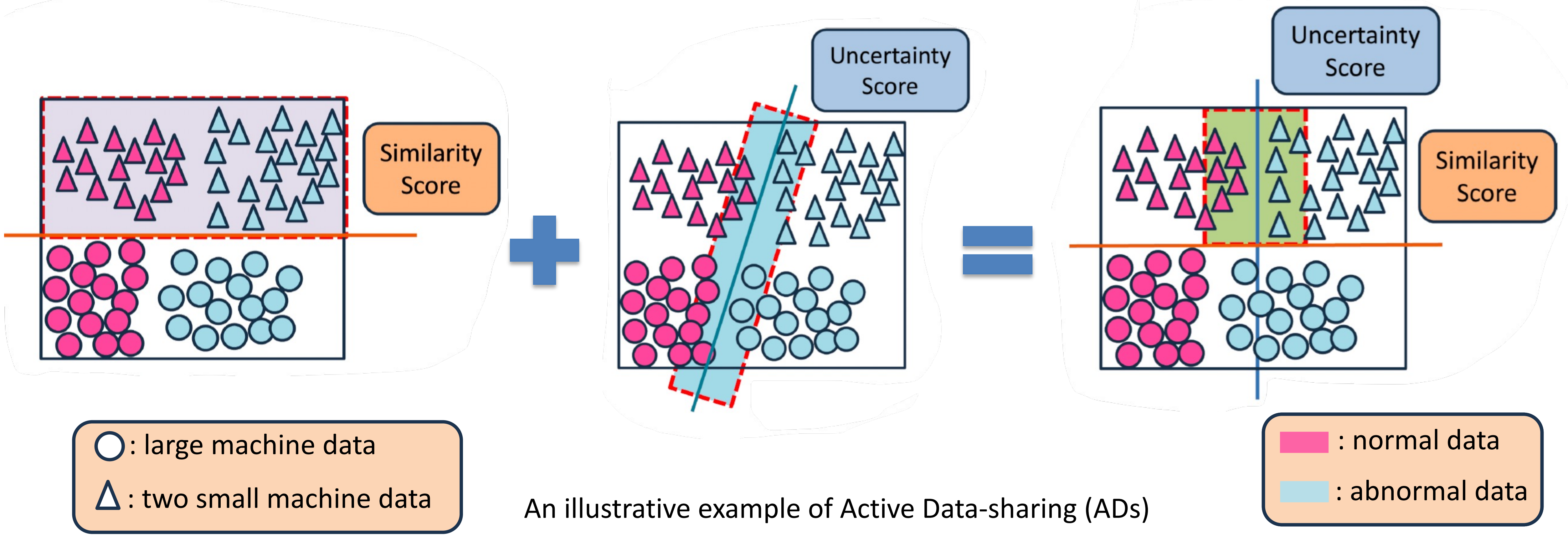



Experimental platform by using Fused filament fabrication (FFF) machine (a) From Two small printer (same model) (b) From one large printer




Printing normal Product: (a)
8 $cm^3$ cube abnormal Product: (b)
8 $cm^3$ cube with a small square void

# Methodology : Active Data-sharing (ADs)




An illustrative example of Active Data-sharing (ADs)

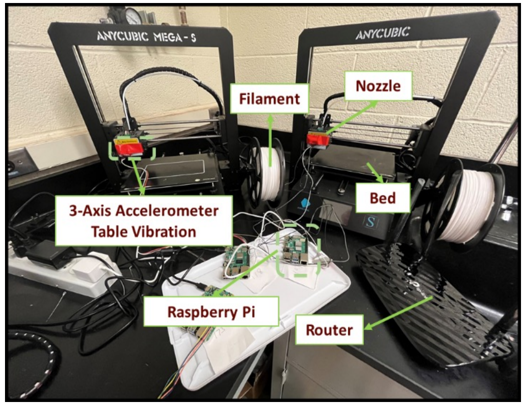
- Extract similar and dissimilar features by contrastive learning
- Use entropy as an uncertainty measure
- Combines the Similarity Score $S_{\text{similarity}}$ and Uncertainty score $S_{\text{Uncertainty}}$ to select samples for labeling

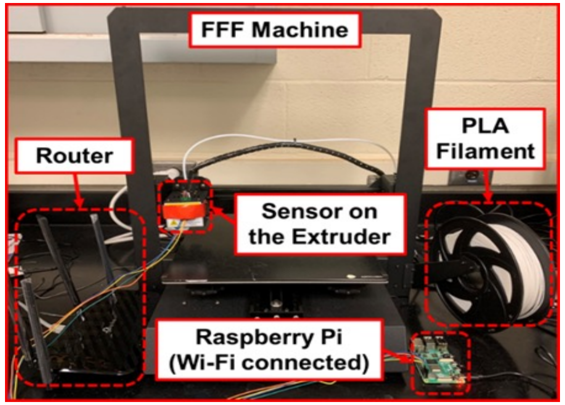
# Case Study: Dataset

- Real world in-situ monitoring data of the same additive manufacturing process from three different machines.

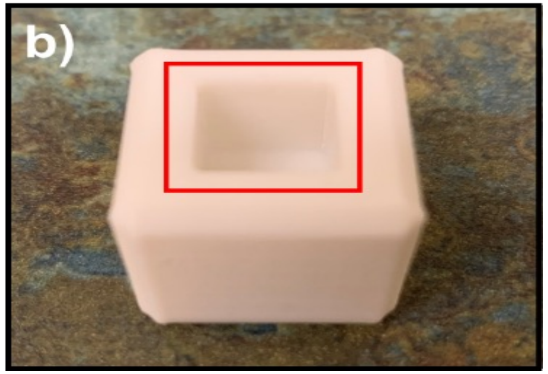
- Two smaller machines are the same make and model (similar distribution)

- A larger machine is from a separate manufacturer (dissimilar distribution)




Experimental platform by using Fused filament fabrication (FFF) machine (a) From Two small printer (same model) (b) From one large printer


Printing normal Product: (a) 8 $cm^3$ cube
abnormal Product: (b) 8 $cm^3$ cube with a small square void

# Case Study: ADs vs. Supervised Learning

**Results of Case Study**

| | Initial | Queried | Accuracy | F1 Score | $\%L$ |
|---|---|---|---|---|---|
| Supervised | 100 | n/a | 0.944 | 0.945 | n/a |
| Random Pick S | 10% | 800 | 0.902 | 0.900 | n/a |
| Random Pick S+L | 10% | 800 | 0.600 | 0.614 | 47 |
| ADs Without CL | 10% | 800 | 0.866 | 0.869 | 3.7 |
| **ADs** | **10%** | **800** | **0.935** | **0.944** | **0.0** |

- ADs Framework:
  - Efficiently queried high-quality data
  - Did not require a full dataset to achieve promising performance
- ADs (22% of labeling data) beats supervised learning (100% of labeling data) with higher accuracy

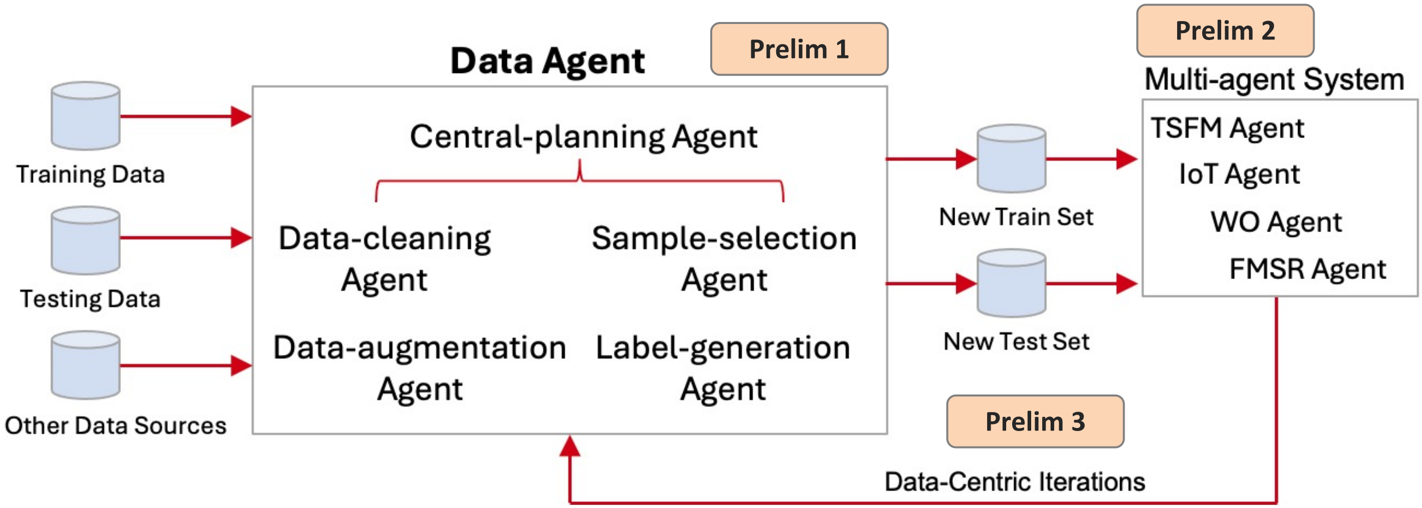
# From Manual Curation to a Data Agent

- WOOD and ADs are effective — but each is a **hand-crafted pipeline**: score functions, acquisition criteria, and training schemes designed by experts for one task.
- Every new dataset, sensor, or task restarts the engineering loop: inspect data, pick criteria, tune, validate.
- LLM-based agents can now plan and execute this loop — but an autonomous curation agent needs three building blocks first:
  1. **Curation actions** that measurably help the downstream task → **Preliminary 1**
  2. **A detector** that exploits the curated data → **Preliminary 2**
  3. **A ground-truth testbed** to evaluate the agent → **Preliminary 3**

**Note:** What follows is a preliminary investigation — the actions and the testbed come first; the planning / orchestration layer that ties them together is the next step.

# Data Agent: Objective, Architecture & Status

- **Goal:** an industrial data agent that automates data curation for downstream anomaly detection.
- **How:** a central-planning agent (LLM) invokes curation actions — cleaning, sample selection, augmentation, label generation — and iterates with the downstream task as the judge.
- **Works with task agents:** time-series foundation models (TSFM), IoT data access, maintenance work orders.
- **Human in the loop** for decisions the data cannot settle (e.g., synthesis, relabeling).




**Note:** Status — curation actions (Preliminary 1–2) and the testbed (Preliminary 3) are built; the planner / orchestration layer that closes the loop is next.

# Preliminary 1 — Action: Sample Selection

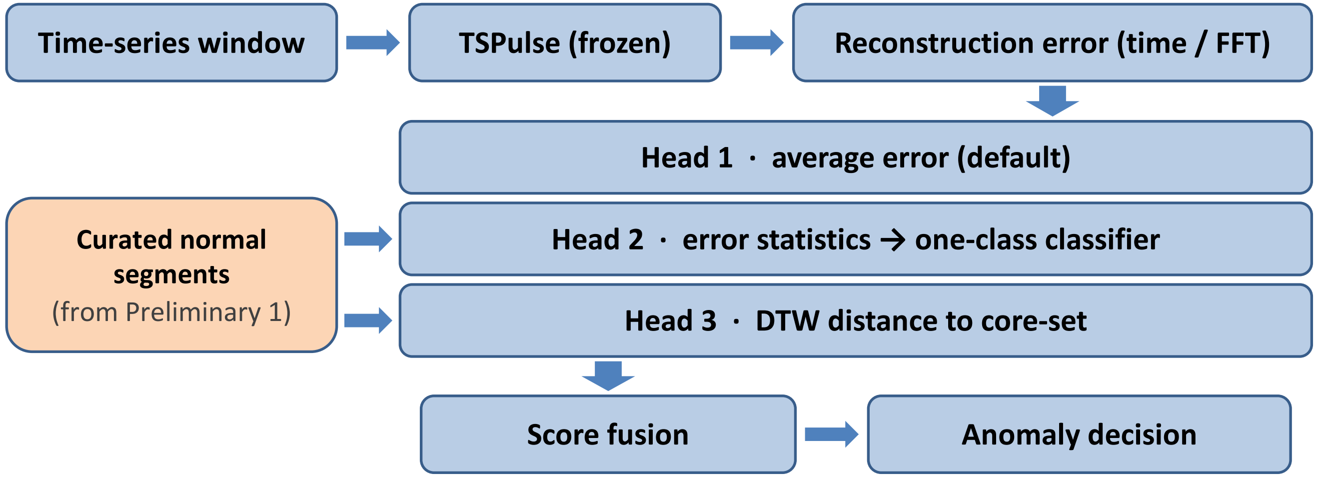
- **Foundation model:** TSPulse — a time-series foundation model (TSFM) for anomaly detection; the default few-shot recipe finetunes on the first 25% of each series.
- **Question:** can informed sample selection beat the default split?
- **Strategies:** rank segments by zero-shot anomaly score; or cluster segments (DTW + K-Medoids) and select representative normal samples.

| Method | Avg. VUS-PR ↑ |
|---|---|
| Zero-shot (TSPulse) | 0.780 |
| Few-shot, default first-25% split | 0.808 |
| Few-shot, selection by anomaly score | 0.815 |
| **Few-shot, selection by clustering (DTW + K-Medoids)** | **0.817** |

**Note:** Which 25% you finetune on matters even for a foundation model — consistent but modest gains (+0.009 over the default split) at zero extra labeling cost.

# Preliminary 2 — Action: Detection Heads

- **Problem:** the foundation model (TSPulse) only outputs a reconstruction error; the default detector thresholds its window average — one generic score that ignores the error’s shape and never uses the curated normal data.
- **Idea:** learn the decision rule from the curated normal segments — (1) statistics of the reconstruction-error sequence (time & FFT: mean, std, percentiles) → one-class classifier (iForest / OC-SVM / LOF); (2) DTW distance to a core-set of representative normal segments; (3) fuse the scores after rescaling.




| Dataset | Default (avg. error) | + Classifier head | Fusion (all heads) |
|---|---|---|---|
| **NAB** | 0.506 | 0.542 | **0.544 (+0.038)** |
| **Exathlon** | 0.827 | **0.895** | 0.872 (+0.045) |
| **SMD** | 0.787 | 0.797 | **0.798 (+0.011)** |
| **MSL** | 0.643 | 0.606 | **0.660 (+0.017)** |
| **SMAP** | 0.708 | 0.706 | **0.750 (+0.042)** |

*VUS-PR ↑ per dataset (avg. over series), TSB-AD-U benchmark — NAB (IT/web metrics), Exathlon (Spark traces), SMD (server metrics), MSL & SMAP (NASA spacecraft telemetry). Frozen zero-shot TSPulse; classifier head = best of iForest / LOF / OC-SVM.*

**Note:** Fusion beats the default detector on 5 of 5 datasets (+0.011 to +0.045; the classifier head alone on 3 of 5). Curated data pays twice — finetuning and a nearly free detection head. Complementary to Preliminary 1; combining both is next.

# Preliminary 3 — Testbed: Physics-Based Scenarios

- Faulty data are rare and expensive — so **simulate them**: MATLAB/Simulink physics-based assets (PMSM motor, battery system, gear, bearing, EV charging cable) with controlled fault injection → labeled time-series.
- Each scenario packs a question, data manifest, ground truth and hints; agents (Claude, Codex, Stirrup, direct LLM) must fetch, clean and analyze the data to answer.
- Evaluation covers the whole process, not just the final answer — pass rate, partial match, tool calls / cost; joint benchmark effort with IBM Research (AssetOpsBench).
- **Early result:** on the first 10 time-series scenarios, frontier coding agents pass only 0–1 of 10 (MiniMax-M3: 0/10; Claude Opus 4.8: 1/10) — the data work is the bottleneck.

**MATLAB physics-based assets**
PMSM motor · battery · gear · bearing · EV cable
→
**Time-series data**
with injected faults
→
**Scenarios / tasks**
question · manifest · ground truth · hints
→
**Agents**
Codex · Claude · Stirrup · direct LLM
→
**Performance evaluation**
pass rate · partial match · tool calls / cost

**Note:** Fault injection gives unlimited ground-truth anomalies — the same role a digital twin can play for an accelerator. Today's agents fail most scenarios: exactly the gap a data agent must close.

# What Transfers to EIC / Accelerator Data?

| Our setting (manufacturing) | EIC / accelerator analogue |
|---|---|
| OOD detection for sensor streams (WOOD) | Off-normal beam states, drifting diagnostics, unseen failure modes |
| Active selection under scarce labels (ADs) | Limited beam time; expert labeling of anomalies is expensive |
| Data sharing across similar machines with distribution mismatch | Reusing data across runs, injectors, and facilities |
| Simulink assets + fault injection for ground truth | Digital twins as fault-data generators for AD benchmarks |
| Agentic curation workflows over TSFM / IoT agents | Agentic AI + data-quality workflows for archived & streaming EIC data |

**Note:** The methods are sensor-agnostic — happy to explore dataset-design and curation benchmarks on EIC data and digital twins.

# Summary

- **WOOD (TPAMI 2024):** a Wasserstein-based score and unified training that let deployed classifiers flag out-of-distribution inputs.
- **ADs (T-ASE 2025):** jointly informative + high-quality data selection for data sharing — 22% of labels outperforms 100% supervised.
- **Data Agent (preliminary):** curation actions — sample selection and detection heads — measurably improve TSFM-based anomaly detection; a physics-simulated testbed shows today's agents fail most data scenarios.
- **Next:** the planning / orchestration layer that closes the loop — an autonomous curation agent — and transfer of these methods to accelerator time series.

*Data-agent investigation: [Student Name(s) — TO FILL]; AssetOpsBench collaboration with IBM Research.* · *yinanw.rpi@gmail.com*