GAN-POWERED SYNTHETIC DATA FOR BATTERY RESEARCH

SUMMER SCHOOL 2025: A I , B A T T E R Y & S U S T A I N A B I L I T Y

Chahinez OUNOUGHI, Ph.D
Department of Business Administration
Tallinn University of Technology

TAL TECH

11.06.2025

AGENDA

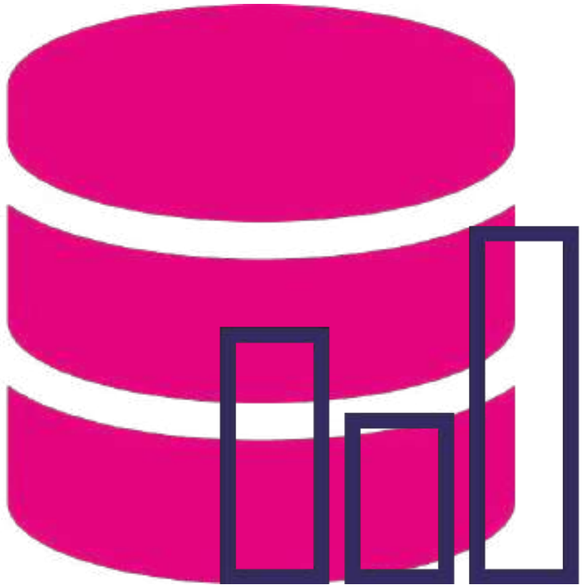
- Introduction
- Understanding GANs
- GANs in Scientific Domains
- GANs for Battery Data
- Evaluation of Sythetic Battery Data
- Challenges & Limitations
- Future Directions
- Interactive Session

INTRODUCTION – *WHY SYNTHETIC DATA FOR BATTERIES?*

Battery Research is Data-Hungry

Real-world battery testing is time-consuming and expensive

Measurements depend on specific hardware, labs, and conditions

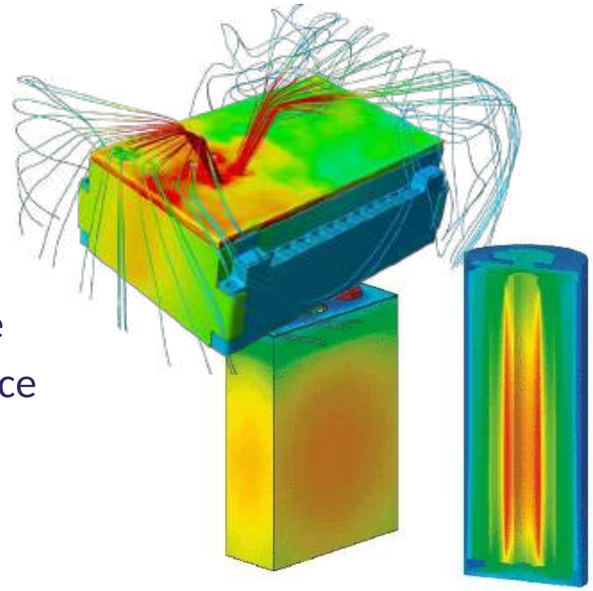


Data may be incomplete, noisy, or imbalanced (e.g., rare failure cases)

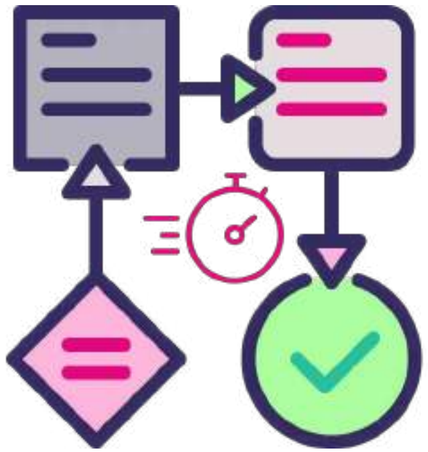
INTRODUCTION – *WHY SYNTHETIC DATA FOR BATTERIES?*

Why Synthetic Data?

Augments real data to improve deep learning model performance



Simulates rare or extreme conditions not captured in limited datasets



Enables faster prototyping and safer testing environments

Supports IP protection and anonymization in collaborative projects

TAL TECH

https://vias3d.com/blogs/how-simulations-can-help-with-battery-design-for-electric-vehicles/
https://images.app.goo.gl/K2cyhMwRKY3LwgFi9

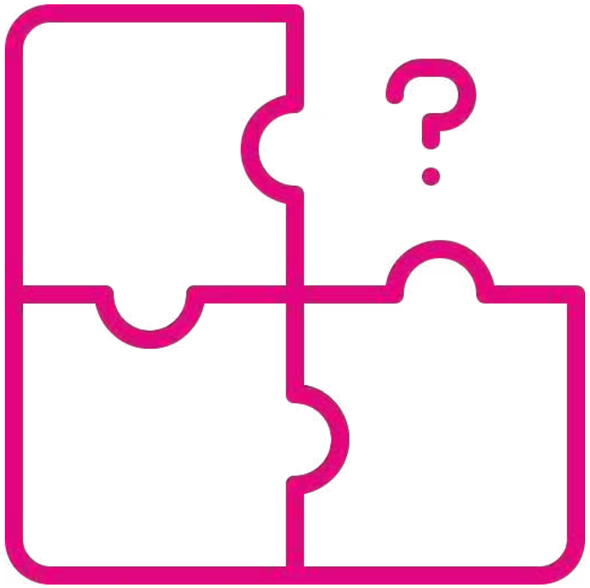
INTRODUCTION – *WHY SYNTHETIC DATA FOR BATTERIES?*

Real Collection vs. Synthetic Generation Pipeline

Battery Lab Setup	Trained GAN Model
Charge/Discharge Cycles	Input:Conditioning Variables (e.g., SoC)
Sensor Readings (Voltage, Temp, etc.)	Output: Generated Battery Data Samples
Time-Consuming (Days to Weeks per test)	Instant Sample Generation
Expensive Equipment + Safety Protocols	Requires Initial Training Only
Subject to Noise and Limited Test Scenarios	Infinite Variety & Rare Event Simulation

INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

- **High Cost & Time**
- **Data Scarcity & Imbalance**
- **Limited Diversity**
- **Data Privacy and IP Concerns**
- **Sensor Noise and Inconsistencies**



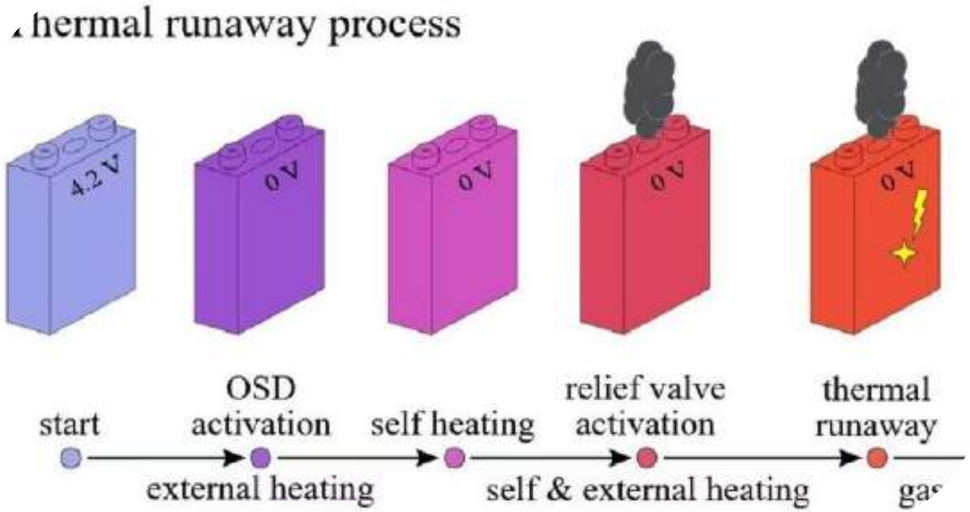
INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

- **High Cost & Time**
  - ☑ **Specialized labs**, **instrumentation**, and **safety measures**
  - ☑ Long test cycles for **charge/discharge**, **degradation**, and **thermal** events.
- **Data Scarcity & Imbalance**
- **Limited Diversity**
- **Data Privacy and IP Concerns**
- **Sensor Noise and Inconsistencies**

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.associatedenvironmentalsystems.com%2Fthe-aes-guide-to-lithium-ion-battery-test-chambers&psig=AOvVaw1OehMvShhm5EYTXpM_1y6X&ust=1747216348414000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCOCh5uOZoI0DFQAAAAAdAAAAABAE

INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

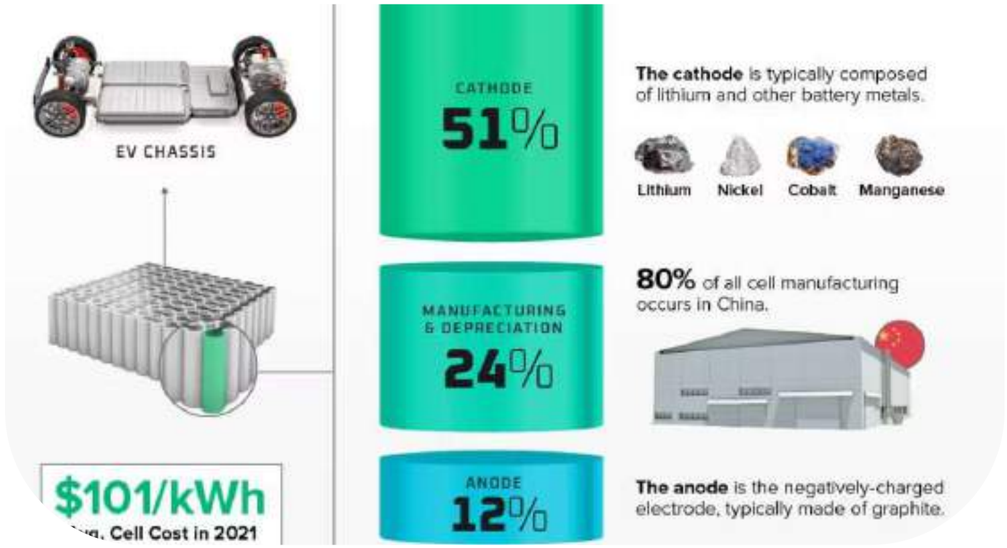
- **High Cost & Time**
- **Data Scarcity & Imbalance**
  - ☑ Rare events (e.g., thermal runaway, aging-related failures) are **hard to capture**.
  - ☑ Imbalanced datasets can lead to **biased or unreliable models**.
- **Limited Diversity**
- **Data Privacy and IP Concerns**
- **Sensor Noise and Inconsistencies**



https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mdpi.com%2F2313-0105%2F9%2F3%2F159&psig=AOvVaw2Y1hURpO7sctaQzWPhWmbC&ust=1747217196194000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCJiW1bWZoI0DFQAAAAAdAAAAABAE

INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

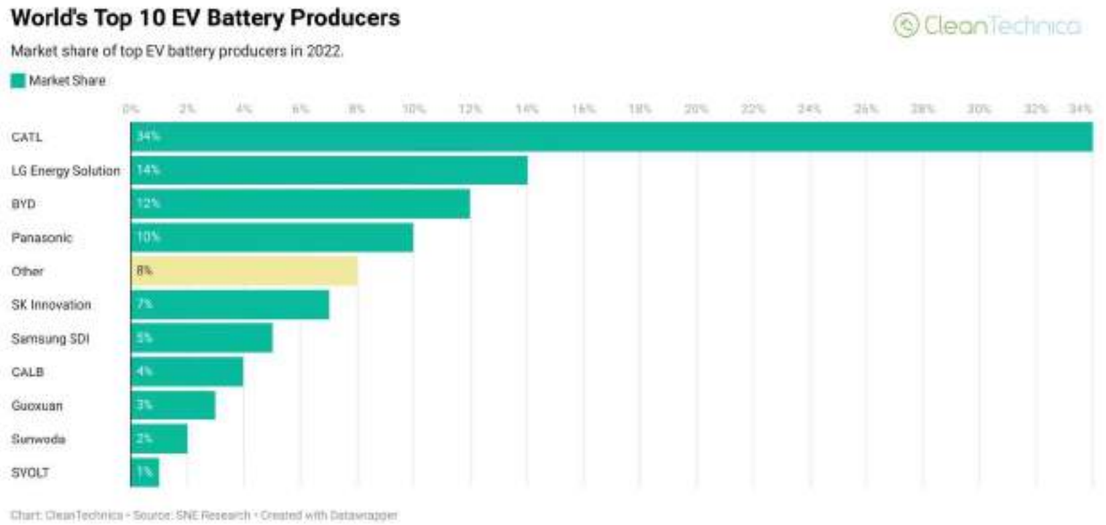
- **High Cost & Time**
- **Data Scarcity & Imbalance**
- **Limited Diversity**
  - ☑ Data may be collected from **limited chemistries, formats**, or **environmental conditions**.
  - ☑ Hard to generalize models across new cell types or use cases.
- **Data Privacy and IP Concerns**
- **Sensor Noise and Inconsistencies**



https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mdpi.com%2F2313-0105%2F9%2F3%2F159&psig=AOvVaw2Y1hURpO7sctaQzWPhWmbC&ust=1747217196194000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCJiW1bWZoI0DFQAAAAAdAAAAABAE

INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

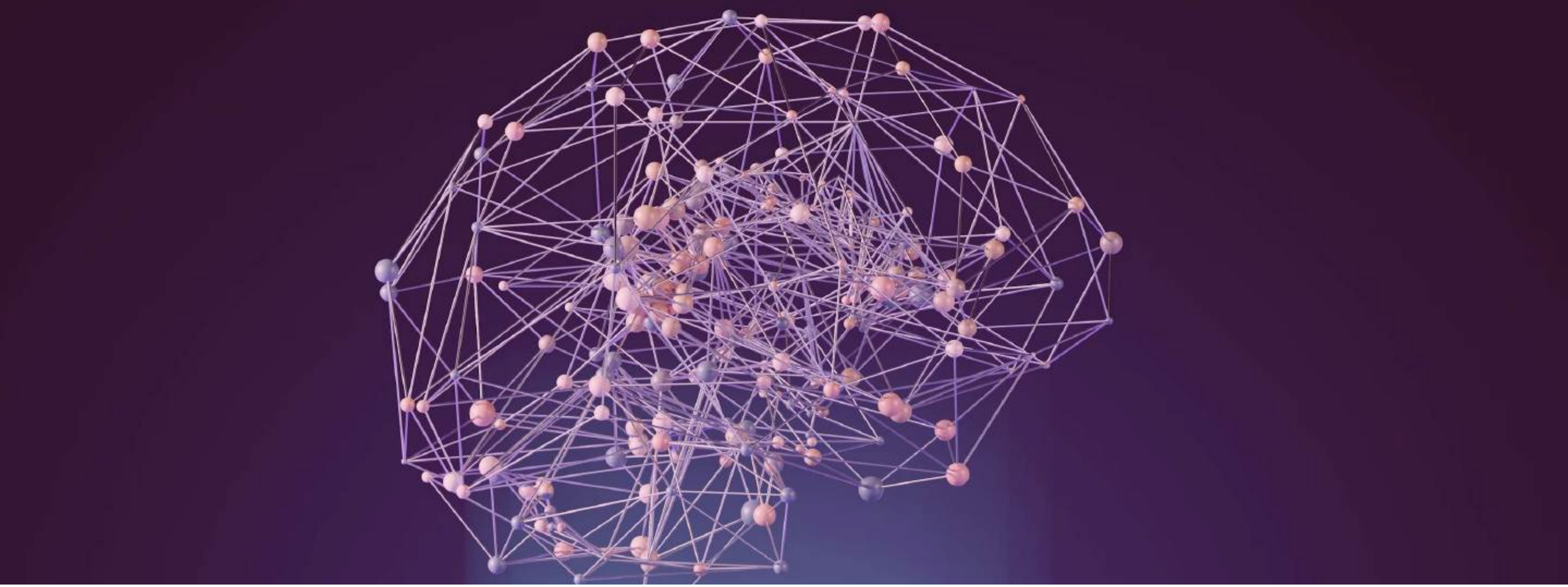
- **High Cost & Time**
- **Data Scarcity & Imbalance**
- **Limited Diversity**
- **Data Privacy and IP Concerns**
  - ☑ Battery manufacturers often **restrict data sharing** due to competitive concerns.
  - ☑ Synthetic data provides a way to collaborate **without exposing proprietary information**.
- **Sensor Noise and Inconsistencies**



https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mdpi.com%2F2313-0105%2F9%2F3%2F159&psig=AOvVaw2Y1hURpO7sctaQzWPhWmbC&ust=1747217196194000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCJiW1bWZoI0DFQAAAAAdAAAAABAE

INTRODUCTION – *CHALLENGES IN BATTERY DATA COLLECTION*

- **High Cost & Time**
- **Data Scarcity & Imbalance**
- **Limited Diversity**
- **Data Privacy and IP Concerns**
- **Sensor Noise and Inconsistencies**
  - ☑ Sensors can introduce **noise**, **drift**, or **inconsistencies** across setups.
  - ☑ Clean, consistent synthetic data can help train more robust models.

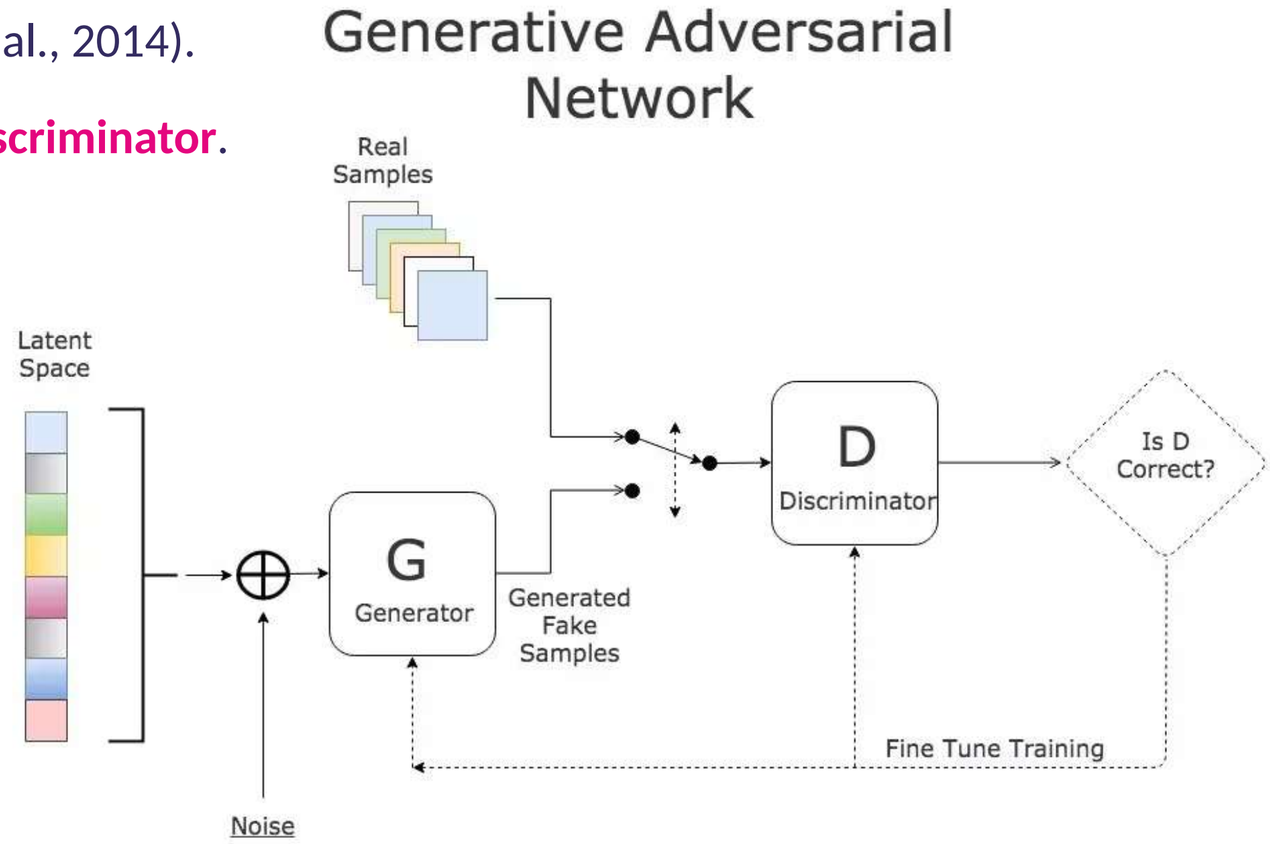


UNDERSTANDING GANS

UNDERSTANDING GANS – *CORE CONCEPTS*

What is a GAN?

- GAN = *Generative Adversarial Network* (Goodfellow et al., 2014).
- Consists of two neural networks: a **Generator** and a **Discriminator**.



Goodfellow, Ian J., et al. "Generative adversarial nets." *Advances in neural information processing systems* 27 (2014).

UNDERSTANDING GANS – *CORE CONCEPTS*

How It Works?

- **Generator** creates synthetic data (e.g., fake battery profiles). It tries to fool the discriminator.
- **Discriminator** evaluates whether the data is real or fake. It penalizes the generator for producing implausible results.
- Both models are trained simultaneously in a game-like process
- The training continues until the **generated** data becomes **indistinguishable** from **real** data.

UNDERSTANDING GANS – *CORE CONCEPTS*

How It Works?

1. When training begins, the generator produces obviously **fake data**, and the **discriminator** quickly learns to tell that it's fake:

2. As training progresses, the generator gets closer to producing output that can fool the discriminator
3. Finally, if generator training goes well, the discriminator gets worse at telling the difference between real and fake. It starts to classify fake data as real, and its accuracy decreases.

UNDERSTANDING GANS – *CORE CONCEPTS*

How It Works?

1. When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it's fake:
2. As training progresses, the generator gets **closer to producing** output that can **fool the discriminator**

3. Finally, if generator training goes well, the discriminator gets worse at telling the difference between real and fake. It starts to classify fake data as real, and its accuracy decreases.

UNDERSTANDING GANS – *CORE CONCEPTS*

How It Works?

1. When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it's fake:
2. As training progresses, the generator gets closer to producing output that can fool the discriminator
3. Finally, if generator training goes well, the **discriminator** gets **worse** at telling the difference between real and fake. It starts to classify **fake data as real**, and its **accuracy decreases**.

UNDERSTANDING GANS – *CORE CONCEPTS*

Training Dynamics

1. **Adversarial Training:** GAN training is a **min-max game**:
   - Generator tries to **maximize** the chance of fooling the Discriminator.
   - Discriminator tries to **minimize** misclassification of real vs. fake data.

Objective: Reach **Nash Equilibrium** where the Generator's output is indistinguishable from real data.

UNDERSTANDING GANS – *CORE CONCEPTS*

Loss Functions in GANs

- Binary Cross-Entropy (Standard GANs): $E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$
  - **Discriminator:** $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right]$
  - **Generator:** $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$

- Wasserstein Loss (WGAN):
  - More stable training by approximating Earth Mover's Distance.
  - Removes vanishing gradient issues in classic GANs.

UNDERSTANDING GANS – *CORE CONCEPTS*

Training Dynamics and Loss Functions in GANs

For each training iteration:

a. Sample real data **x ~ p_data(x)**

b. Sample noise **z ~ p_z(z)**

c. Generate fake data: **x_fake = G(z)**

d. Compute Discriminator Loss **L_D**

e. Update Discriminator weights to minimize **L_D**

f. Freeze Discriminator weights

g. Compute Generator Loss **L_G**

h. Update Generator weights to minimize **L_G**

Repeat until convergence

UNDERSTANDING GANS – *CORE CONCEPTS*

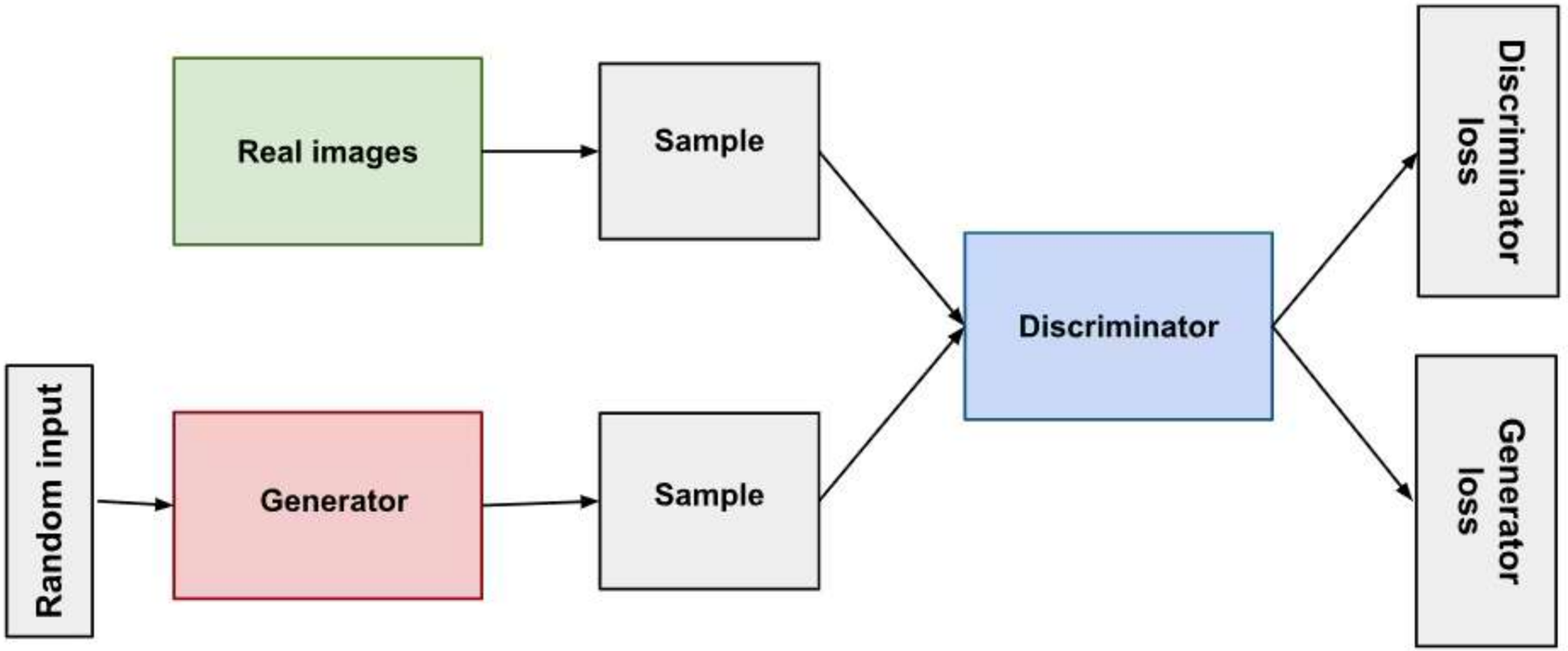
Training Challenges

- Mode collapse: Generator produces limited variations.
- Vanishing gradients: Discriminator too strong or weak.
- Training instability: Careful balancing required.

GANS IN SCIENTIFIC DOMAINS

Variants of GANs

1. **Vanilla GAN** – Basic form using only noise as input.

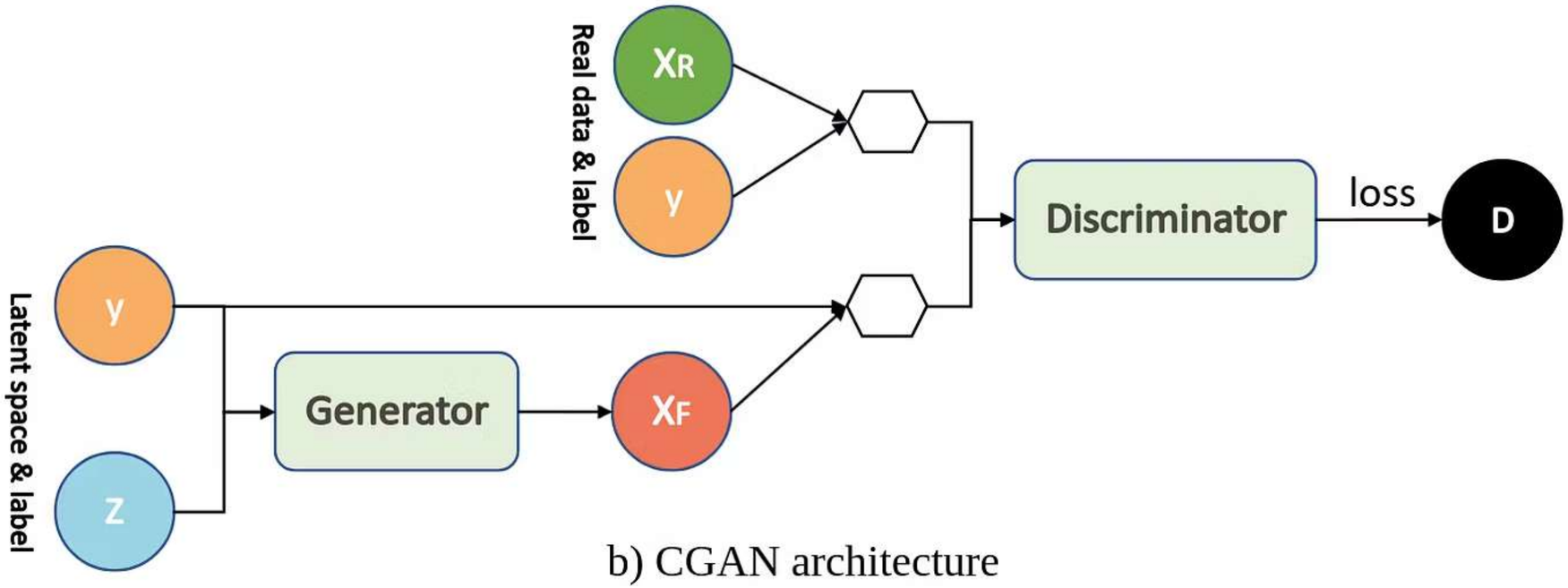


2. **Conditional GAN (cGAN)**
3. **CycleGAN**
4. **Wasserstein GAN (WGAN)**
5. **StyleGAN / BigGAN**

GANS IN SCIENTIFIC DOMAINS

Variants of GANs

1. **Vanilla GAN**
2. **Conditional GAN (cGAN)** – Uses labels or input data (e.g., sensor readings) to guide generation.
3. **CycleGAN**
4. **Wasserstein GAN (WGAN)**
5. **StyleGAN / BigGAN**

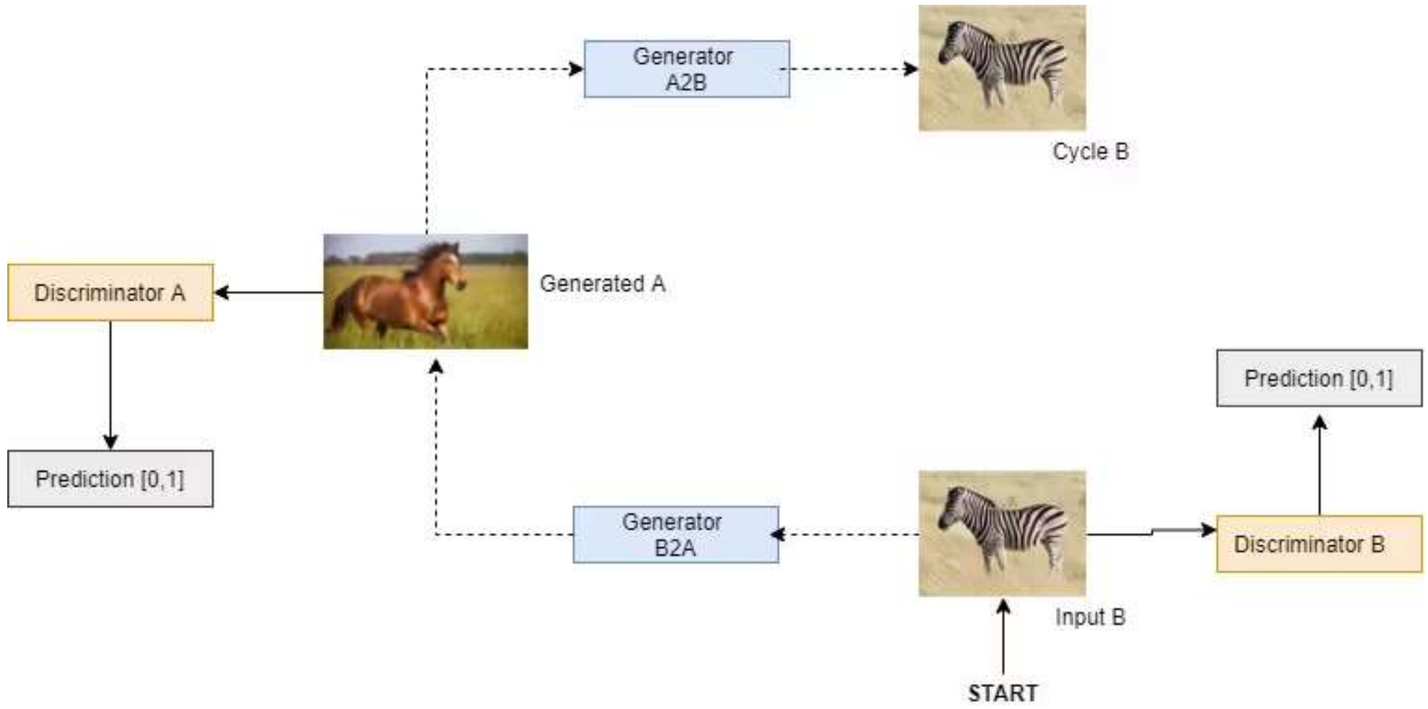


b) CGAN architecture

GANS IN SCIENTIFIC DOMAINS

Variants of GANs

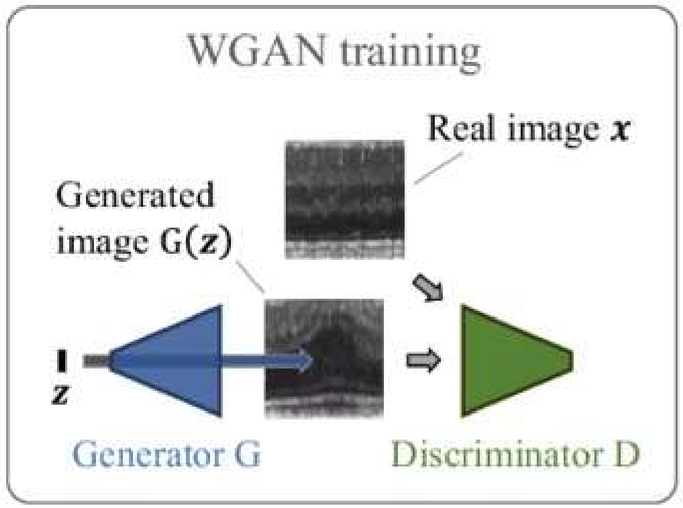
1. **Vanilla GAN**
2. **Conditional GAN (cGAN)**
3. **CycleGAN** – Translates data between two domains (e.g., low-res ↔ high-res thermal maps).
4. **Wasserstein GAN (WGAN)**
5. **StyleGAN / BigGAN**

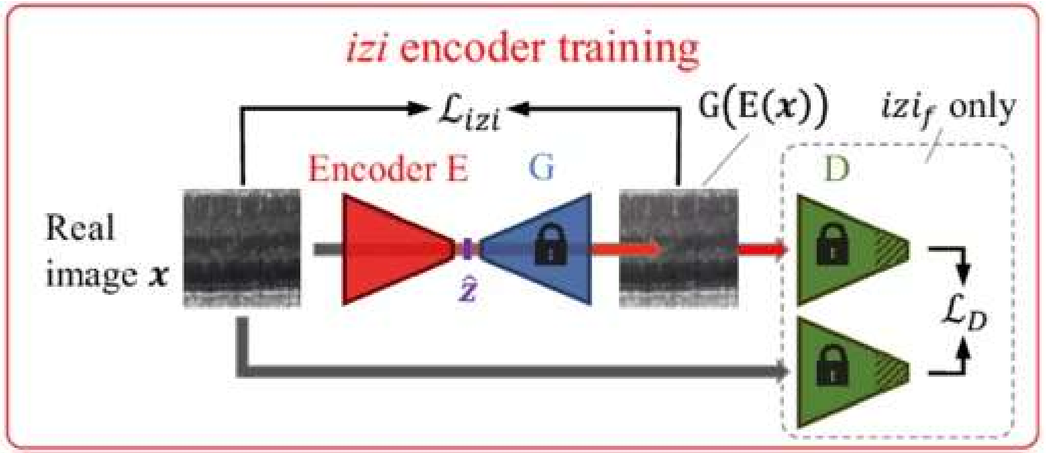


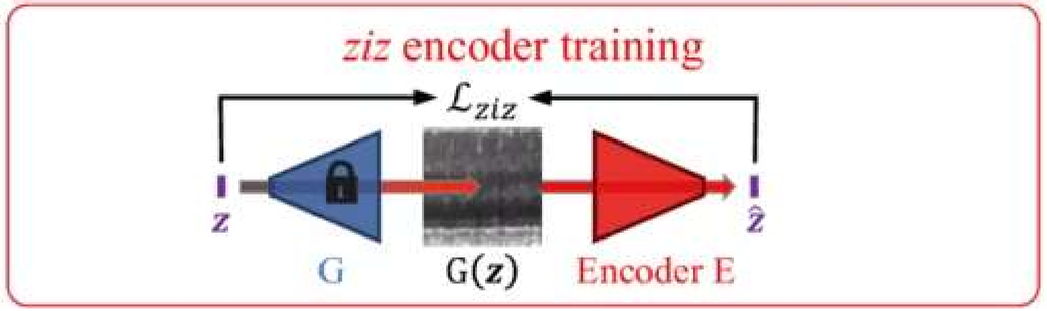
GANS IN SCIENTIFIC DOMAINS

Variants of GANs

1. **Vanilla GAN**
2. **Conditional GAN (cGAN)**
3. **CycleGAN**
4. **Wasserstein GAN (WGAN)** – More stable training using Wasserstein distance.
5. **StyleGAN / BigGAN**



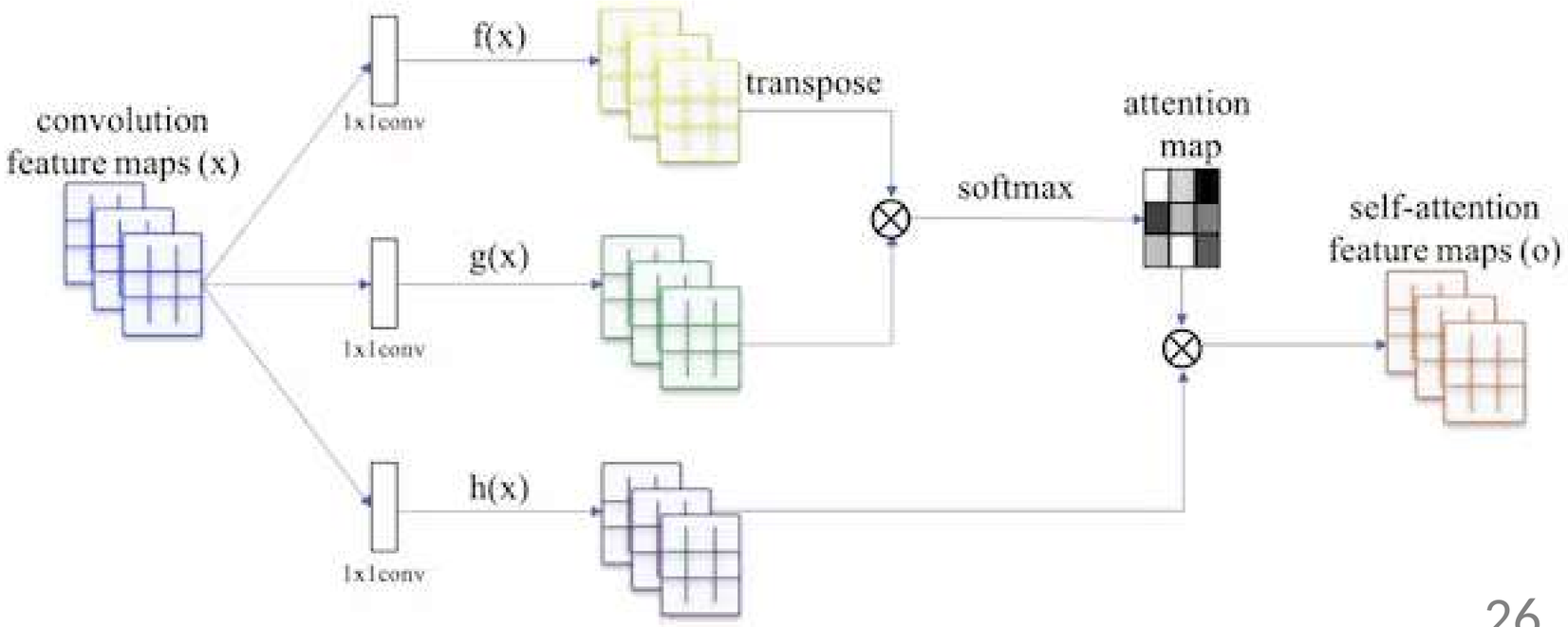




GANS IN SCIENTIFIC DOMAINS

Variants of GANs

1. **Vanilla GAN**
2. **Conditional GAN (cGAN)**
3. **CycleGAN**
4. **Wasserstein GAN (WGAN)**
5. **StyleGAN / BigGAN** – Used for high-fidelity image synthesis (less common in scientific domains).



GANS IN SCIENTIFIC DOMAINS

Applications in Science

1. **Synthetic Time-Series Generation** (e.g., voltage, current, capacity over cycles).
2. **Thermal Profile Prediction** (e.g., predicting internal cell temperature).
3. **Anomaly Detection** (e.g., rare fault generation for diagnostic models).
4. **Material Design** (e.g., predicting electrode structures).
5. **Simulation Data Augmentation** (e.g., physics-based simulations in battery R&D).

GANS FOR BATTERY DATA

DESIGNING GANS FOR BATTERY DATA GENERATION

What Does Battery Data Look Like?

- Multivariate time-series: Voltage, Current, Temperature, SoC, Capacity
- Structured as: $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}\ \boldsymbol{x} \in \mathbb{R}$
- Can also include metadata: ambient temperature, charging protocol, cell type

How Do We Use GANs?

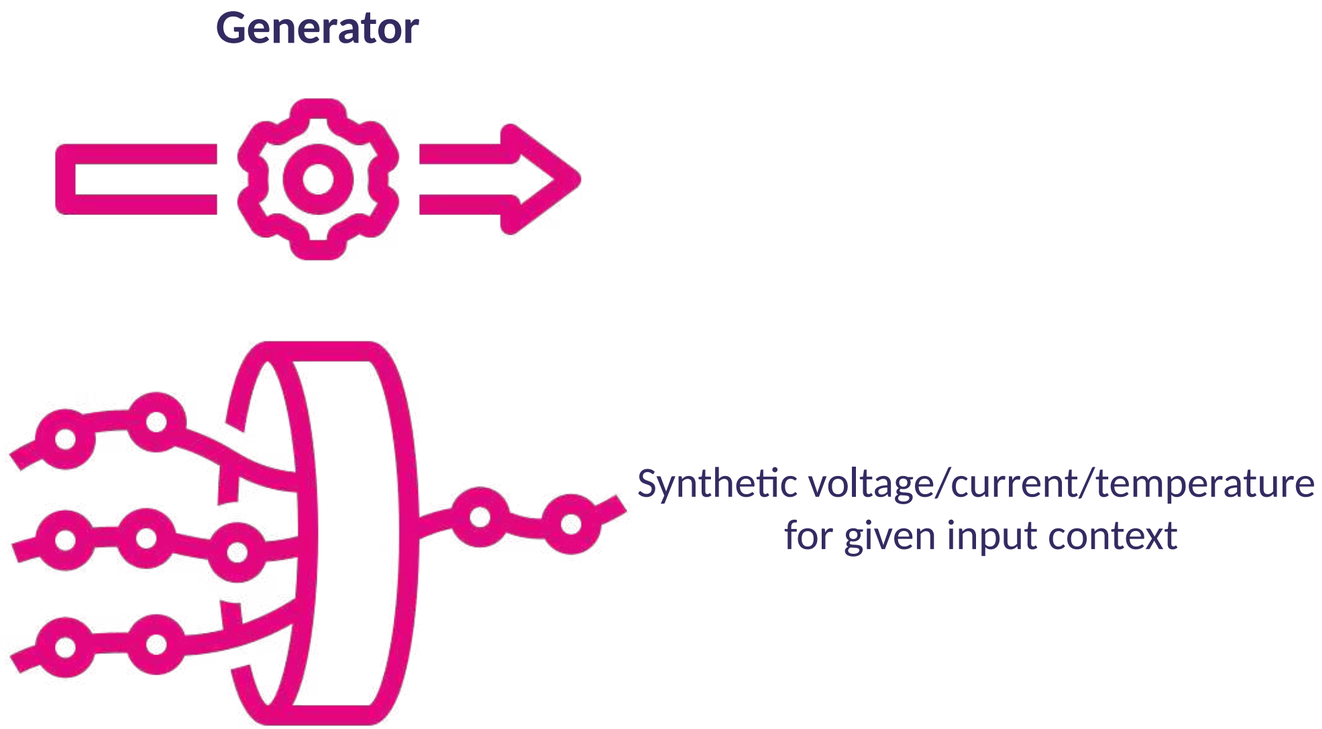
- Generator learns to **simulate realistic battery behavior over time**
- Input can be **pure noise (z)** or **conditioned on real values**

DESIGNING GANS FOR BATTERY DATA GENERATION

Conditioning the Generator (cGAN Setup)

- Noise vector z
- Conditioning vector c:

c ={SoC,Cycle Number,Temperature,Cell Type}

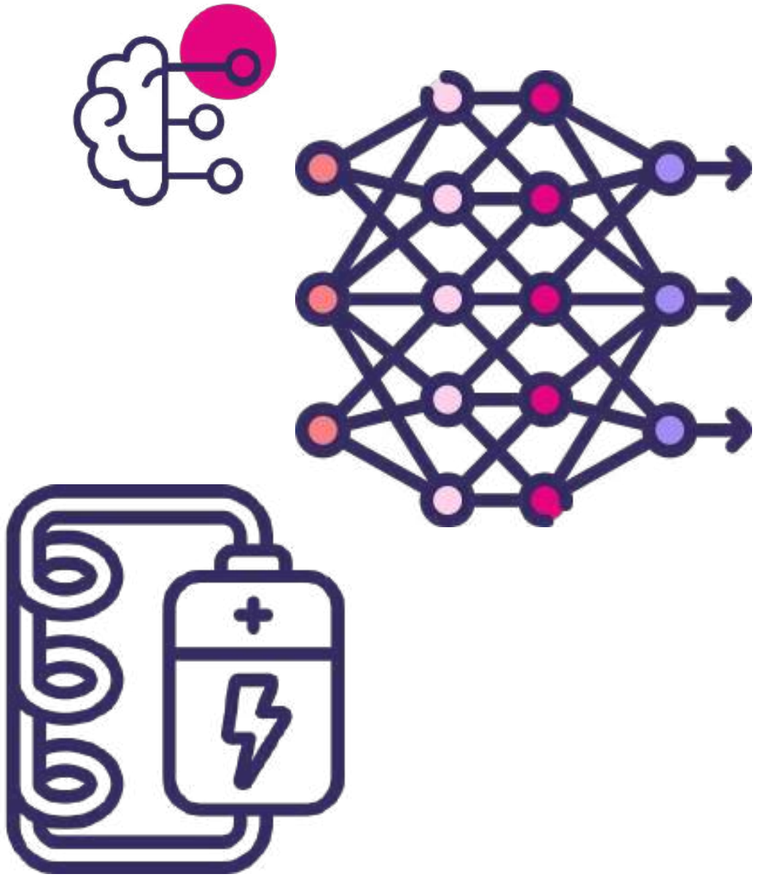


Synthetic voltage/current/temperature for given input context

DESIGNING GANS FOR BATTERY DATA GENERATION

Network Choices

- 1D CNNs / LSTMs for time-series generation
- Tabular GANs (e.g., CTGAN) for cycle-level data
- Physics-aware architecture: integrates constraints or custom losses



DESIGNING GANS FOR BATTERY DATA GENERATION

Hands-On Example – Conditional GAN for Battery Data

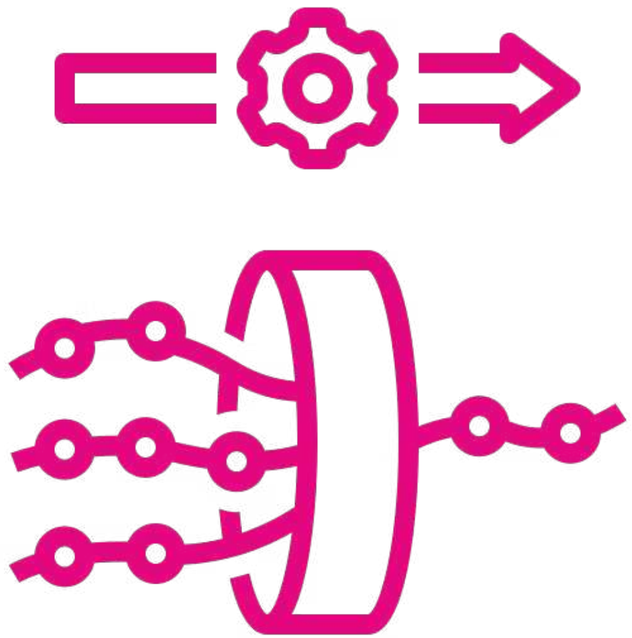
Objective

- Use a Conditional GAN (cGAN) to generate synthetic temperature profiles of a lithium-ion cell during charging.
- Conditioning inputs: *State of Charge (SoC)*, *Cycle number*, *Ambient temperature*

DESIGNING GANS FOR BATTERY DATA GENERATION

Hands-On Example – Conditional GAN for Battery Data

Generator



- Noise vector $z \in R$
- Conditioning vector c = {SoC, Cycle, Temp}

Synthetic temperature series:

$T = \{T1, T2, \ldots, Tt\}\ t \in R$

DESIGNING GANS FOR BATTERY DATA GENERATION

Hands-On Example – Conditional GAN for Battery Data

Architecture

- Generator: LSTM + Dense layers
- Discriminator: CNN + Fully Connected
- Loss: Binary Cross-Entropy + optional physics-consistency loss

```python
class LSTMGenerator(nn.Module):
    def __init__(self, condition_dim, noise_dim, hidden_dim, sequence_length):
        super().__init__()
        self.lstm = nn.LSTM(input_size=condition_dim + noise_dim, hidden_size=hidden_dim, batch_first=True)
        self.fc = nn.Linear(hidden_dim, 1)
        self.sequence_length = sequence_length

    def forward(self, noise, cond):
        batch_size = cond.size(0)
        cond_repeated = cond.unsqueeze(1).repeat(1, self.sequence_length, 1)
        noise_repeated = noise.unsqueeze(1).repeat(1, self.sequence_length, 1)
        lstm_input = torch.cat((noise_repeated, cond_repeated), dim=2)
        lstm_out, _ = self.lstm(lstm_input)
        return self.fc(lstm_out).squeeze(-1)
```

```python
def physics_consistency_loss(generated_sequence, expected_trend=None):
    penalty = torch.mean(torch.clamp(-generated_sequence, min=0))  # Penalize negative temps
    if expected_trend is not None:
        penalty += torch.mean((generated_sequence - expected_trend) ** 2)
    return penalty
```

```python
class CNNDiscriminator(nn.Module):
    def __init__(self, condition_dim, sequence_length):
        super().__init__()
        self.conv1 = nn.Conv1d(1, 16, kernel_size=3, padding=1)
        self.relu = nn.LeakyReLU(0.2)
        self.pool = nn.MaxPool1d(2)
        self.fc1 = nn.Linear(16 * (sequence_length // 2) + condition_dim, 64)
        self.fc2 = nn.Linear(64, 1)
        self.sigmoid = nn.Sigmoid()

    def forward(self, sequence, cond):
        x = sequence.unsqueeze(1)
        x = self.relu(self.conv1(x))
        x = self.pool(x)
        x = x.view(x.size(0), -1)
        x = torch.cat((x, cond), dim=1)
        x = self.relu(self.fc1(x))
        return self.sigmoid(self.fc2(x))
```

DESIGNING GANS FOR BATTERY DATA GENERATION

Hands-On Example – Conditional GAN for Battery Data

Training Data Example

SoC (%)	Cycle	Ambient Temp (°C)	Real Peak Temp (°C)
25	150	38.2	60
30	200	41.5	80

Goal!

- Ensure the synthetic outputs match the shape and range of realistic thermal behavior.
- Validate through visual plots, statistical comparison, and model-based testing.

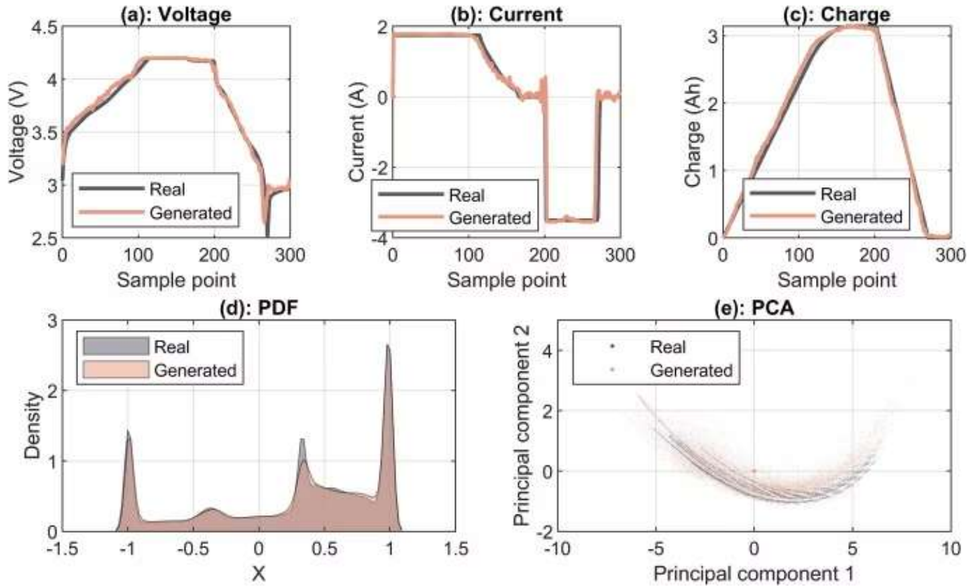
DATA EVALUATION OF SYNTHETIC BATTERY DATA

Data Evaluation of Synthetic Battery Data

1. Visual Comparison

- Overlay real vs. synthetic time-series plots (voltage, temperature, etc.)
- Check if generated curves follow similar patterns and trends

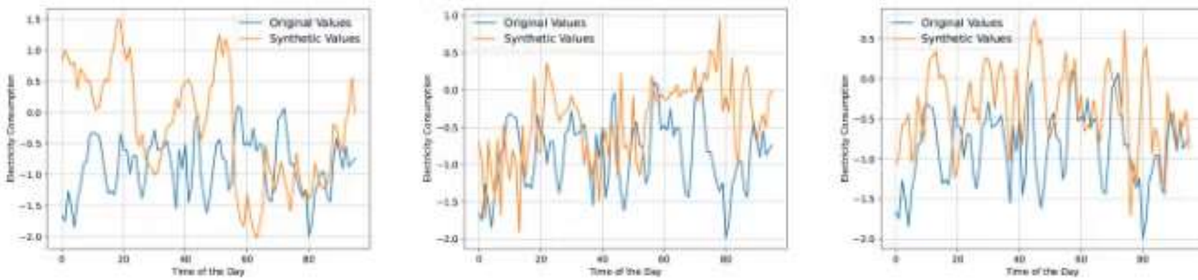
https://doi.org/10.1016/j.jechem.2024.12.010



2. Statistical Validation

- Mean, standard deviation, skewness, kurtosis
- Distribution matching using KS test or Earth Mover's Distance
- Auto-correlation comparison for time-series

https://doi.org/10.1016/j.eswa.2024.123851



(d) WGAN (e) WGAN-GP (f) DRAGAN

DATA EVALUATION OF SYNTHETIC BATTERY DATA

Data Evaluation of Synthetic Battery Data

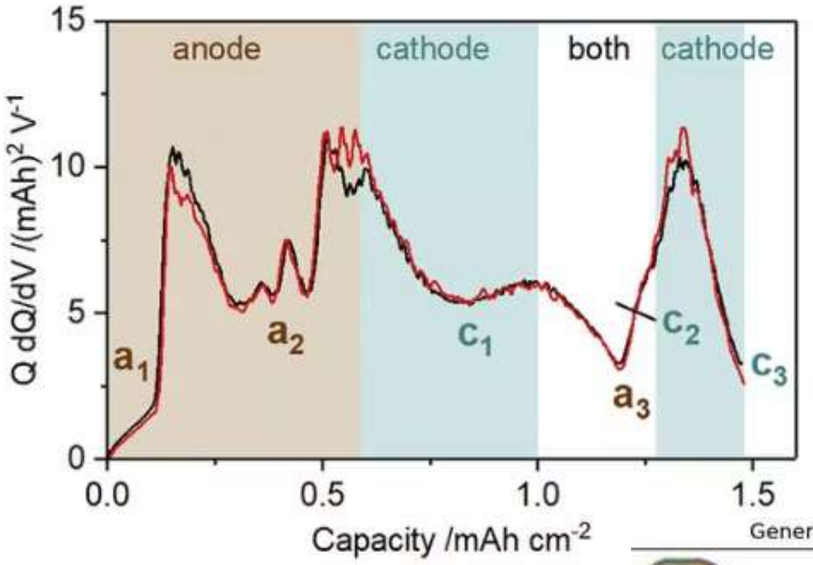
3. Domain-Specific Metrics

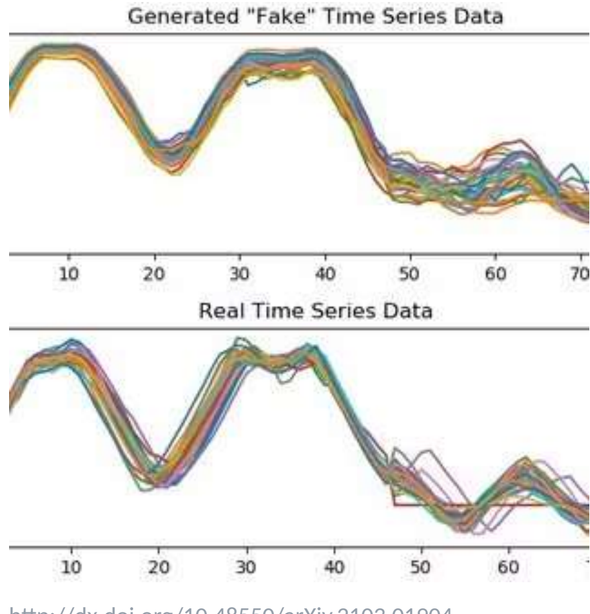
- Cycle life predictions, thermal peak accuracy
- Alignment with electrochemical behavior

4. Model-Based Testing

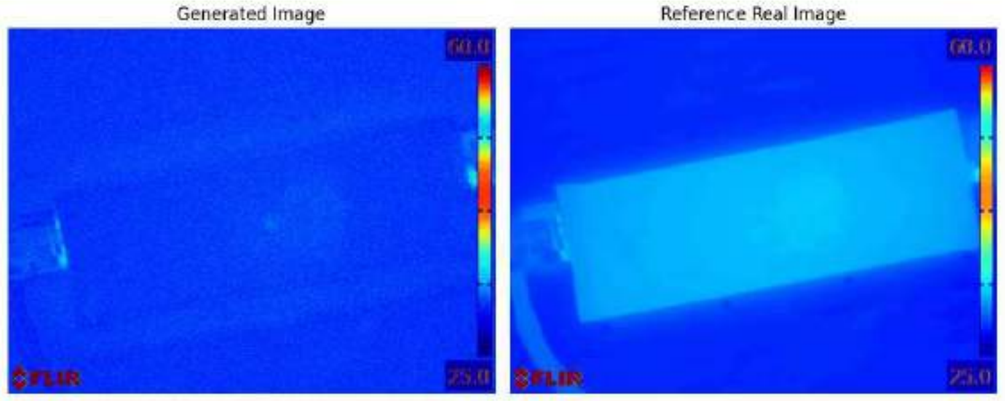
- Train a predictive model on synthetic data → test on real data
- Evaluate if the model generalizes well

http://dx.doi.org/10.1016/j.xcrp.2020.100253



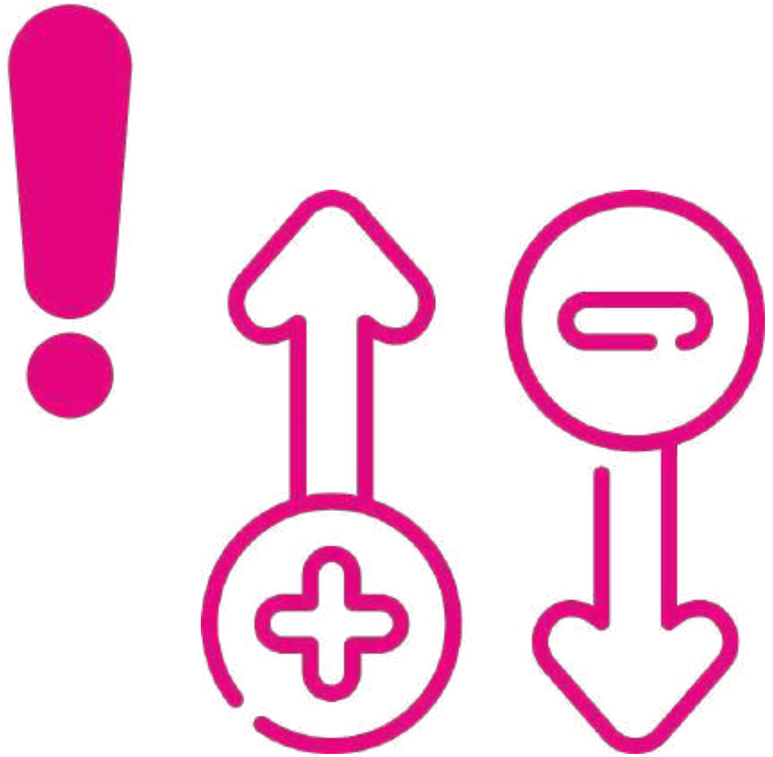


http://dx.doi.org/10.48550/arXiv.2103.01904



DATA EVALUATION OF SYNTHETIC BATTERY DATA

Data Evaluation of Synthetic Battery Data



5. Failure Case Identification

- Detect mode collapse or unrealistic repetitions
- Look for unphysical outputs: negative temperatures, flat curves, etc.

CHALLENGES & LIMITATIONS OF GANS IN BATTERY MODELING

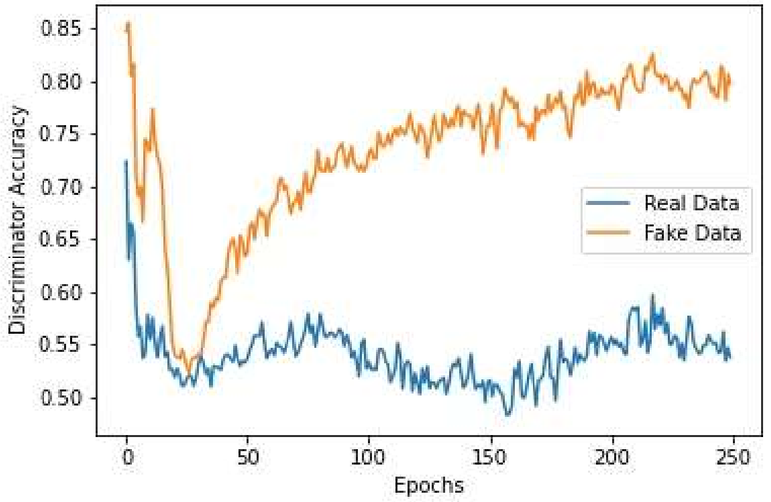
Key Pitfalls When Applying GANs to Battery Data

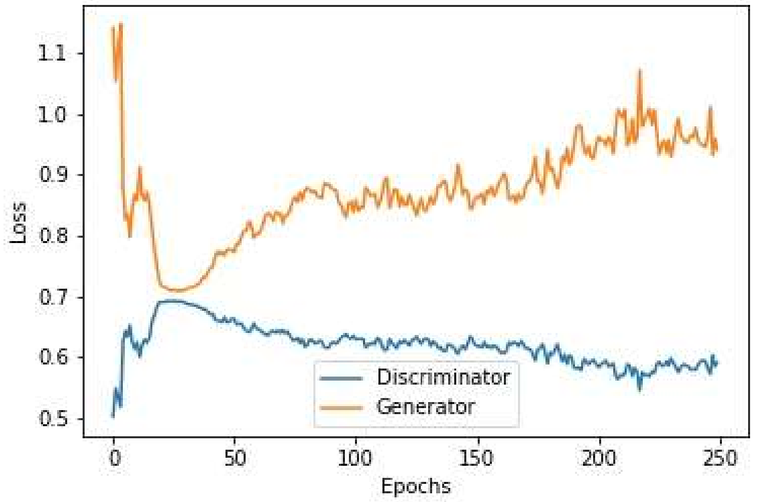
Mode Collapse

- Generator produces limited or identical outputs.
- Misses rare but important battery behavior (e.g., degradation spikes).

Training Instability

- GANs are notoriously hard to train.
- Requires delicate balance between Generator and Discriminator.
- Sensitive to hyperparameters and architecture choice.





FUTURE DIRECTIONS – PHYSICS-INFORMED GANS & BEYOND

What's Next for GANs in Battery Research?

1. Physics-Informed GANs (PI-GANs)

- Incorporate domain knowledge into the training process.
- Enforce physical laws (e.g., conservation of energy, electrochemical constraints).
- Reduces unrealistic outputs and improves trust.

2. Hybrid Models

- Combine GANs with physics-based simulators or differential equations.
- Use simulators for baseline → GANs fill in high-resolution or edge-case behavior.

FUTURE DIRECTIONS – PHYSICS-INFORMED GANS & BEYOND

What's Next for GANs in Battery Research?

3. GANs in Digital Twins

- Synthetic data enables real-time scenario simulation for battery systems.
- Enhance Battery Management Systems (BMS) with generative components.

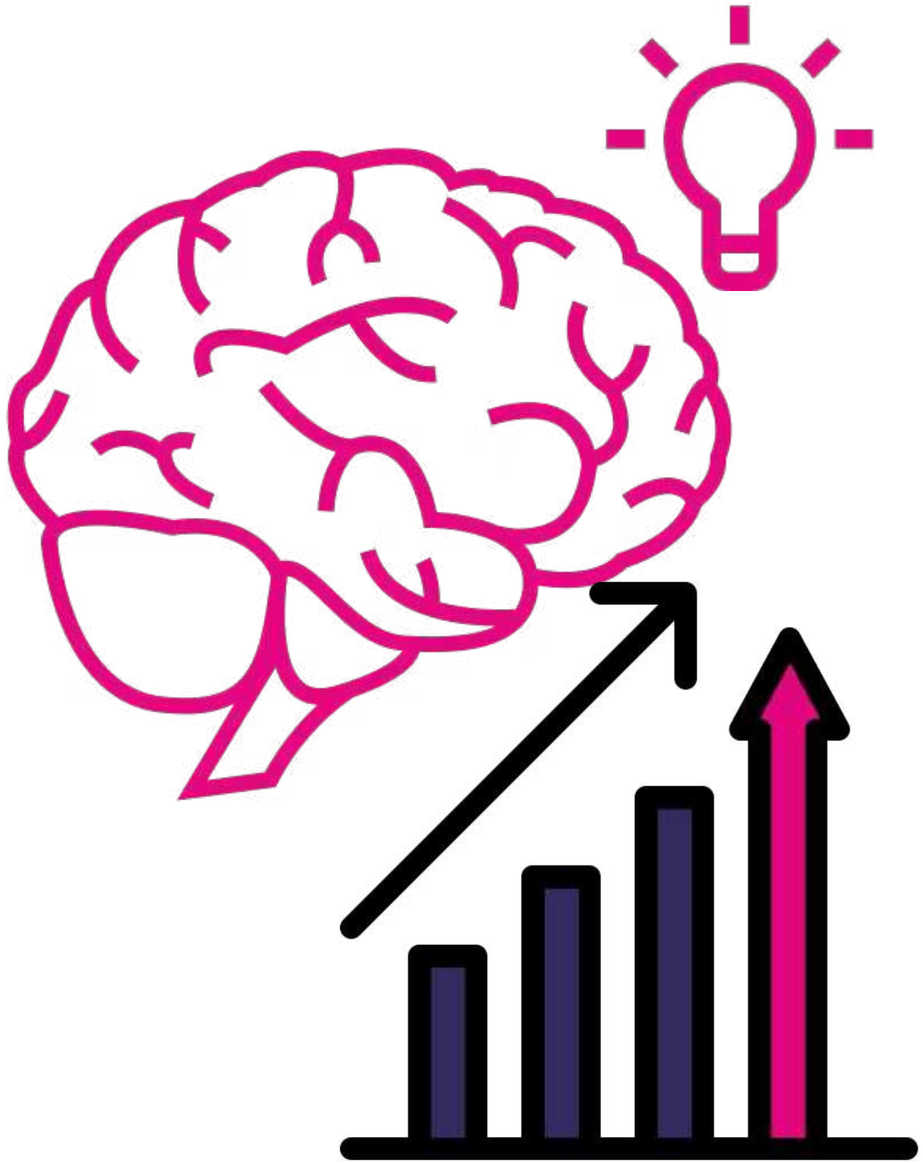
4. Continual Learning & Online Adaptation

- Adapt GAN models to new data as batteries age or environments change.
- Reduce need for full retraining.

Questions?

Ideas?

Your Use Cases?

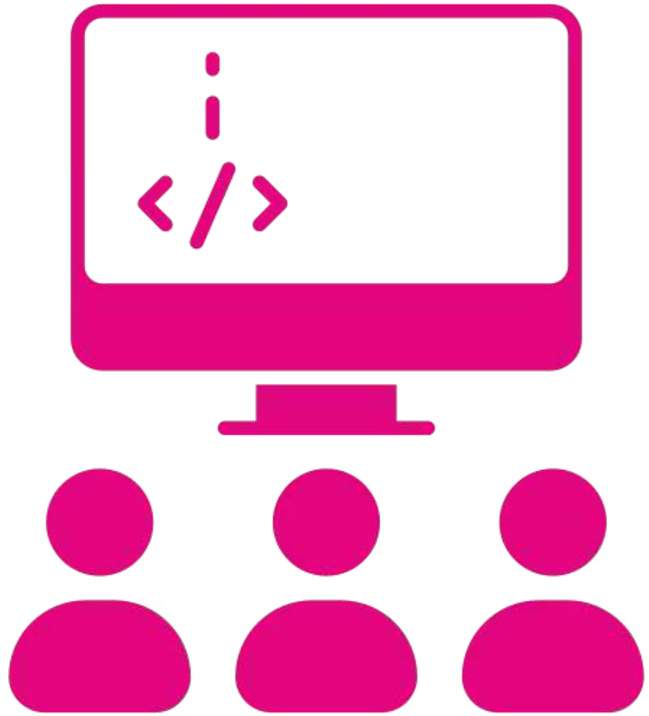


INTERACTIVE PRACTICAL EXAMPLES

Generate Synthetic Data with GANs

Tasks:

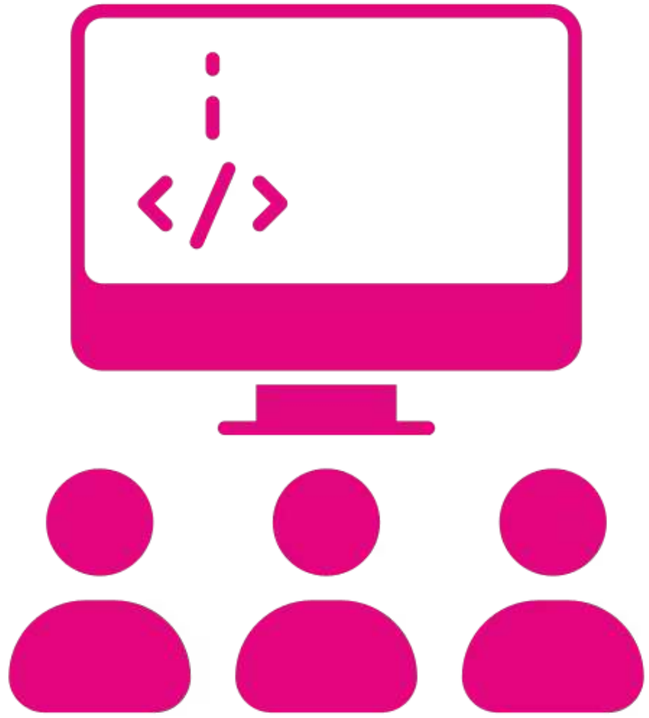
1. Load and visualize the dataset
2. Prepare conditioning vectors (SoC, cycle, ambient T)
3. Build the Generator and Discriminator
4. Train the model & monitor loss curves
5. Generate synthetic data for new SoC/cycle combinations
6. Compare synthetic vs. real outputs (plots + stats)



INTERACTIVE PRACTICAL EXAMPLES

Generate Synthetic Data with GANs

https://github.com/Ounoughi-Chahinez/GAN-Based-Battery-Data-Synthesis



Thank you for your attention!

Contact

E-mail: chahinez.ounoughi@taltech.ee

Phone: +372 53701439