



# SWIFT HYDRA: SELF-REINFORCING GENERATIVE FRAMEWORK FOR ANOMALY DETECTION WITH MUL- TIPLE MAMBA MODELS

**Nguyen Do<sup>1</sup>, Truc Nguyen<sup>2</sup>, Malik Hassanaly<sup>2</sup>, Raed Alharbi<sup>3</sup>, Jung Taek Seo<sup>4</sup>, My T. Thai<sup>1\*</sup>**

<sup>1</sup>University of Florida, FL, USA      <sup>2</sup>National Renewable Energy Laboratory, CO, USA

<sup>3</sup>Saudi Electronic University, Saudi Arabia      <sup>4</sup>Gachon University, South Korea

nguyen.do@ufl.edu, Truc.Nguyen@nrel.gov, Malik.Hassanaly@nrel.gov,  
ri.alharbi@seu.edu.sa, seojt@gachon.ac.kr, mythai@cise.ufl.edu



Nguyen Do, PhD student  
University of Florida, USA



Truc Nguyen, PhD  
National Renewable Energy Lab, USA



Malik Hassanaly, PhD  
National Renewable Energy Lab, USA



Raed Alharbi, Professor  
Saudi Electronic University, Saudi Arabia



Jung Taek Seo, Professor  
Gachon University, South Korea

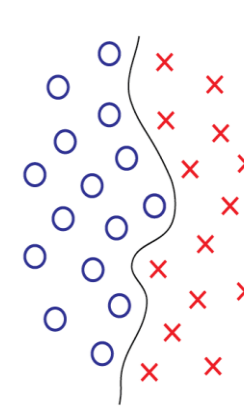
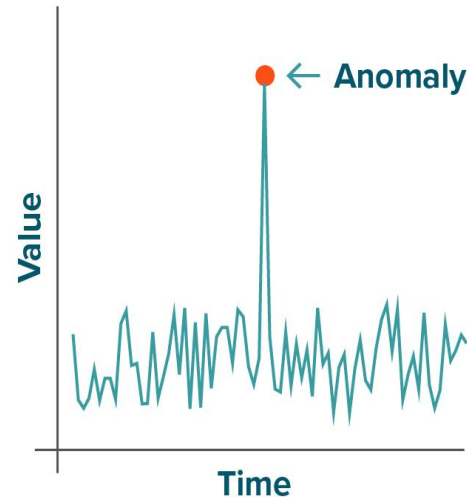
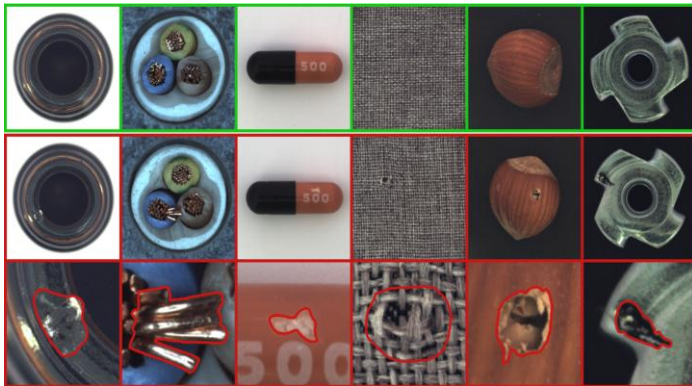


My T Thai, Professor  
University of Florida, USA

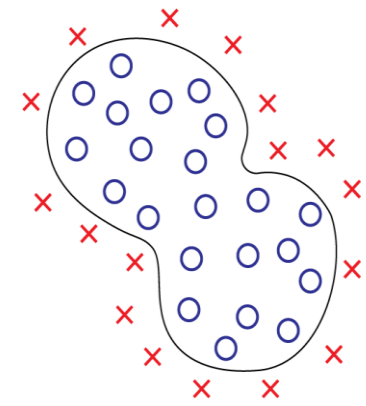
# What is Anomaly Detection ?

**Anomaly Detection** is the process of *identifying* data points, events, or observations that *deviate significantly from the expected pattern or behavior* in a dataset. These anomalies can indicate critical insights such as *fraud, system failures, or unusual trends*. Some applications such as:

1. **Fraud Detection:** Identifying unusual transactions in banking, finance, or online shopping.
2. **System Monitoring & Security:** Detecting intrusions, cyberattacks, or software malfunctions.
3. **Predictive Maintenance:** Identifying early signs of machine failures before breakdowns occur.
4. **Healthcare & Medical Analysis:** Recognizing abnormal patterns in medical data.
5. **User Behavior Analysis:** Detecting unusual user activities in websites, applications, or networks.



Classification



Anomaly Detection

## Drawbacks of the Existing SOTA Methods

- **DTE[1]**: Employs a diffusion model for fast inference, halting when reconstruction errors exceed a threshold. While highly efficient, its reliance on reconstruction leads to memorizing anomalies, reducing reliability. It is ideal for speed-focused scenarios but compromises accuracy in complex datasets. (**Fast but unreliable**)
- **ADGym[2]**: Optimizes key design elements like loss functions and architectures, significantly improving anomaly detection performance. However, it requires high computational resources, struggles with new datasets, and lacks scalability, making it less effective for diverse or dynamic environments. (**High cost and poor scalability**)
- **Rejex[3]**: Implements a fixed rejection threshold for rejecting uncertain predictions, enhancing reliability and theoretical guarantees. Despite these strengths, it lacks adaptability to unlabeled or rapidly changing datasets, limiting its usability in real-world, complex, or evolving scenarios. (**Lacks Generalization**)

















[1] DTE paper (ICLR 2024 Spotlight): <https://openreview.net/forum?id=1R3rk7ysXz>

[2] ADGym paper (NeurIPS 2023): <https://openreview.net/forum?id=9CKx9SsSSc&noteId=jXiE88oWUQ>

[3] Rejext paper (NeurIPS 2023): <https://openreview.net/forum?id=WK8LQzzHwW&noteId=Db98cMHlw7>

*During the work for ICLR 2025, papers in NeurIPS 2024 had not yet been published.*

# Swift Hydra: Objectives and Goals

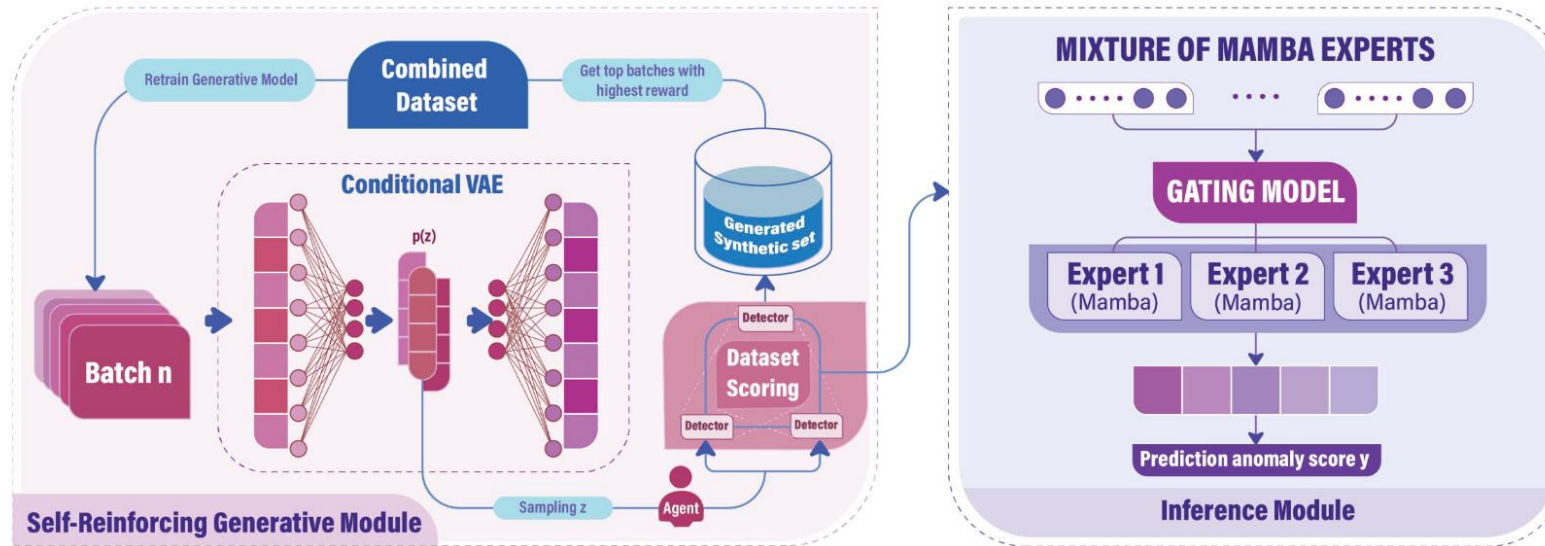
	High Accuracy	Fast Inference	Generalization	Scalability
<u>DTE (ICLR 2024 Spotlight)</u>				
<u>ADGym (NeurIPS 2023)</u>				
<u>Rejext (NeurIPS 2023)</u>				
<u>Swift Hydra (ICLR 2025)</u>				

**We have GenAI + RL**

**We have Mixture of Mamba Expert (MoME)**

During the work for ICLR 2025, papers in NeurIPS 2024 had not yet been published.

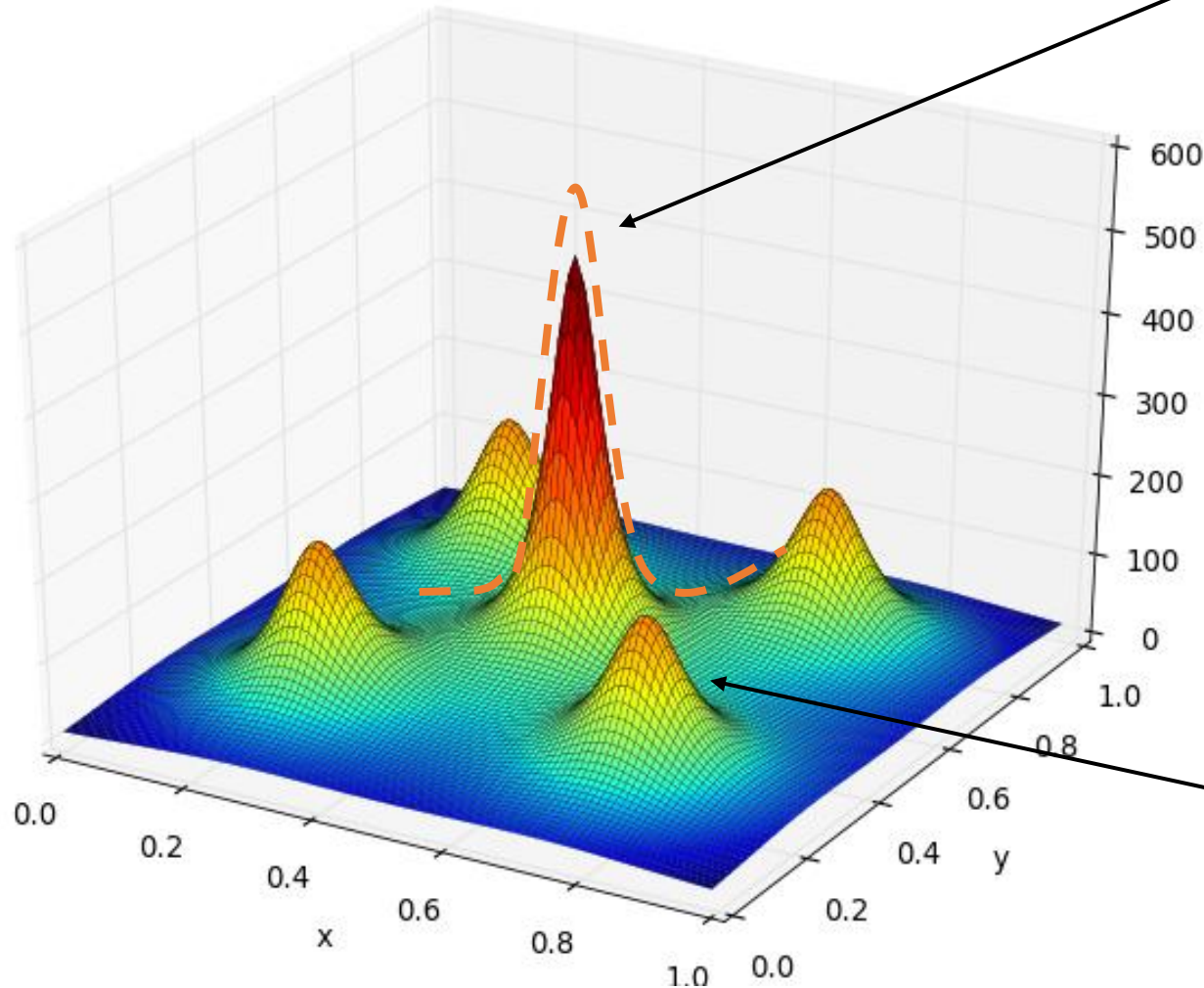
## Swift Hydra: Overall Framework



The Swift Hydra Framework consists of two main modules: the *Self-Reinforcing Generative Module* and the *Inference Module*. The first module includes a **C-VAE**, an **RL agent**, and a **large Mamba-based Detector**. Initially, the C-VAE is trained on the original dataset, referred to as the Combined Dataset in episode 0. In the early stages, the RL agent generates diverse anomalies by refining latent vectors  $z$ , then shifts to producing anomalies that more effectively deceive the detector. The top  $l$  anomalies are added back to the original dataset, creating a new combined dataset to further improve the Generative Model. The second module employs a **Mixture of Mamba Experts (MoME)**, where lightweight models specialize in different parts of the dataset, providing the same performance as the **large detector** but with significantly faster inference.

## How RL and GenAI Collaborate Effectively?

The distribution is captured by the Generative AI model. Note that GenAI models tend to capture features that appear most frequently.



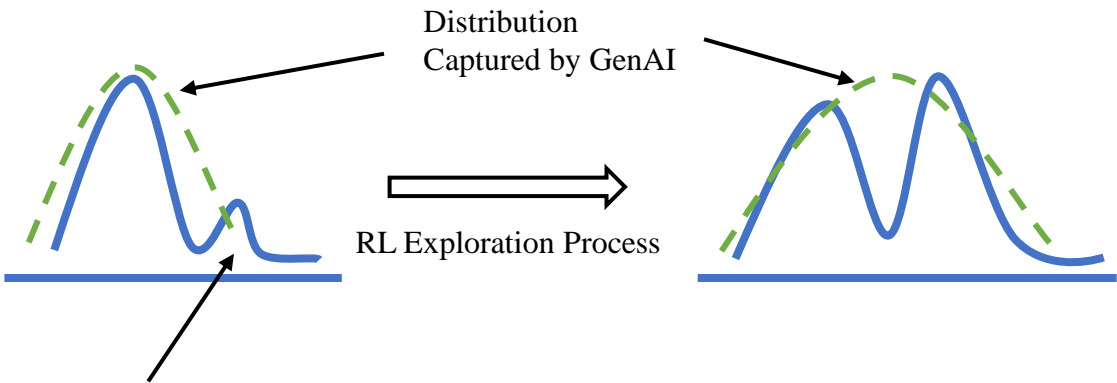
These features are less frequent in the training dataset but may be extremely important, as they could potentially maximize a certain objective function.

In the context of anomaly detection, if we use a GenAI model to generate anomalies, it often produces anomalies with features that appear most frequently in the data, which can make them easily detectable as anomalies by the detector. Instead, we can use an RL agent to modify the feature space of the GenAI model.

$$\mathcal{R}(\mathcal{M}_\phi(\hat{z}_i, y_i = 1), e) = \underbrace{\gamma^e \cdot \mathcal{H}(\mathcal{D}_e^{train} \cup \mathcal{M}_\phi(\hat{z}_i, y_i = 1))}_{\text{Exploration term}} - \underbrace{\log \mathcal{W}_\kappa(\mathcal{M}_\phi(\hat{z}_i, y_i = 1))}_{\text{Exploitation term}},$$

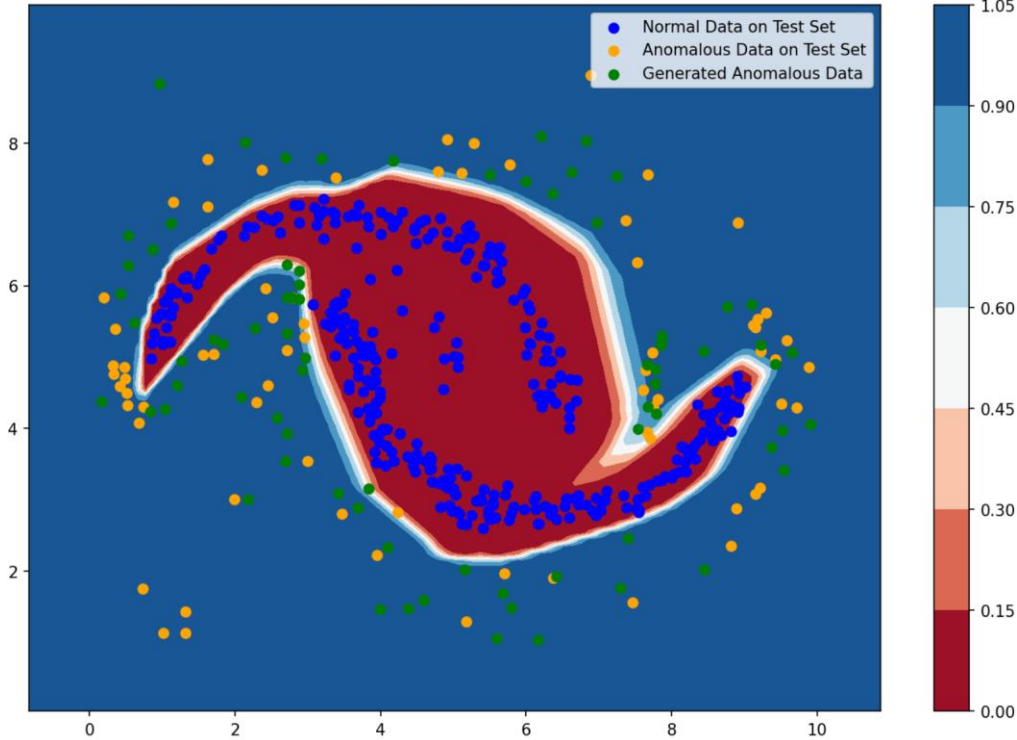
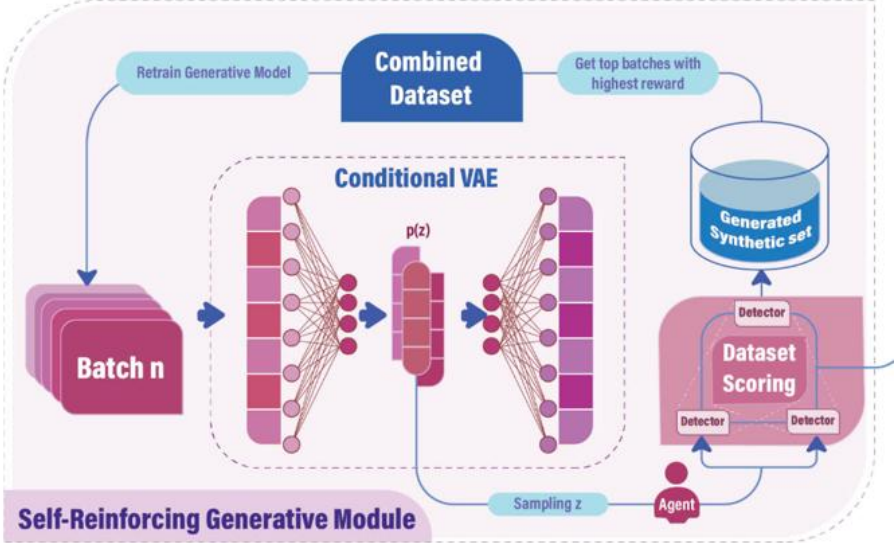
**Early Stage:** Conditional GenAI is trained on the anomaly class to generate the most diverse data possible through Entropy Maximization (increasing the chaotic of the data). This can be considered as the Exploration Process in RL literature.

**Later Stage:** In the later stage, the RL model focuses on identifying samples that can most effectively deceive the detector among the explored samples. This can be considered the Exploitation Process in RL literature.



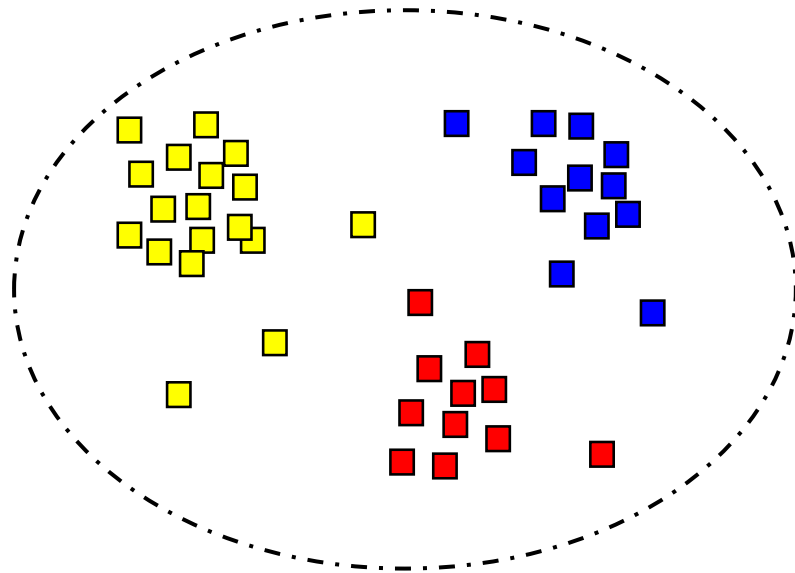
Maybe these feature regions have significant potential to help the detector generalize better. However, these features appear infrequently in the dataset.

Adding potential datapoints back to the dataset changes the distribution, increasing the frequency of rare features. This helps GenAI better capture these features and generate improved data in later episodes.

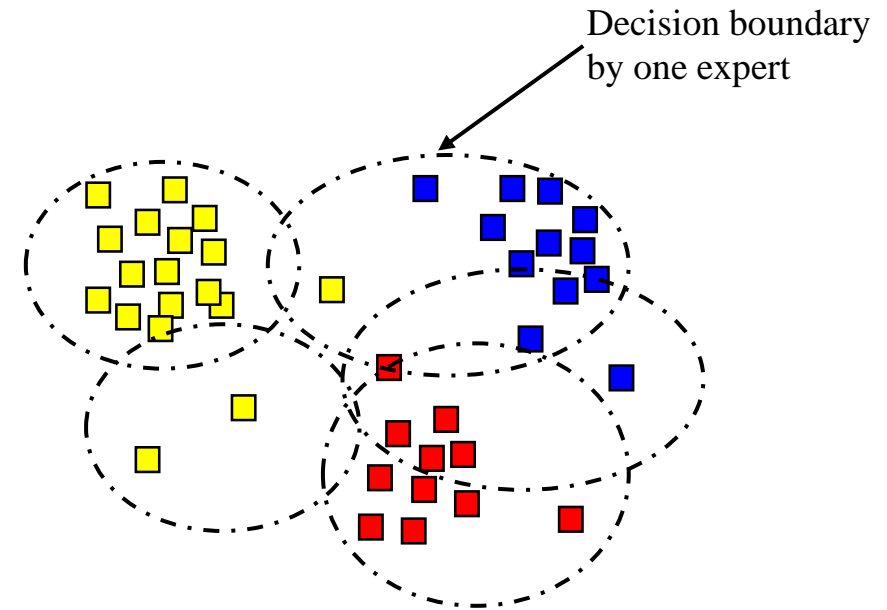


## Mixture of Mamba Expert

After completing phase 1, we observed the following: Building a large detector to capture all data points (including original and generated data) would result in a model with a high number of parameters, leading to high inference time. This approach would not be scalable for challenging anomaly detection tasks that require generating more data during phase 1.



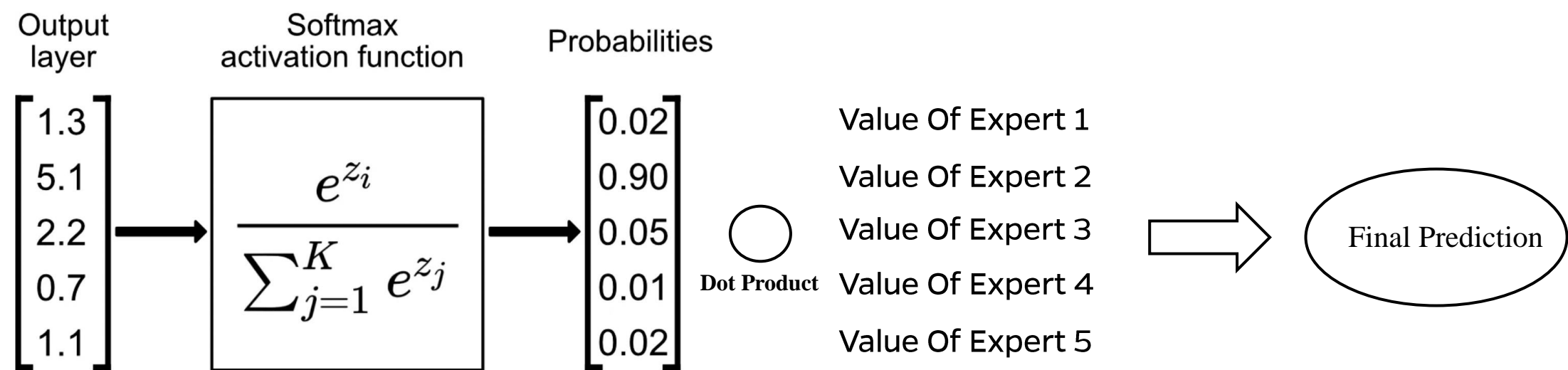
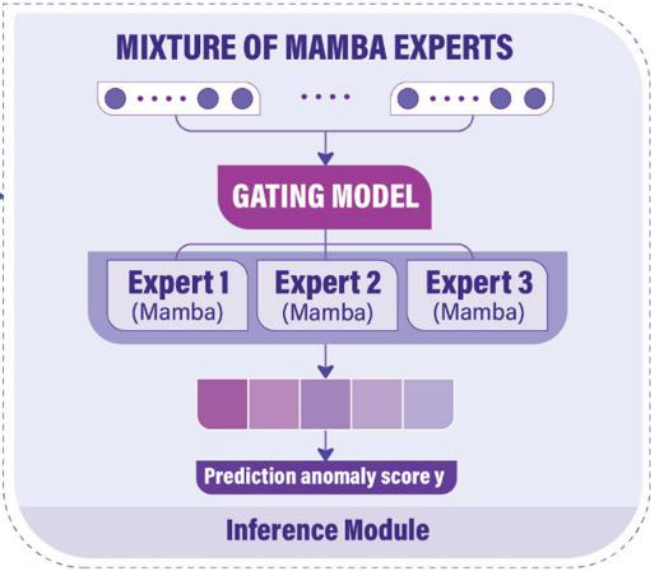
**Single Large Detector**



**Mixture of Mamba Expert**

$$\mathbf{h}(x, \mathfrak{N}_g, \mathfrak{N}_{\text{noise}}) = x \cdot \mathfrak{N}_g + \text{StandardNormal}(\cdot) \cdot \text{Softplus}(x \cdot \mathfrak{N}_{\text{noise}})$$

$$\mathfrak{F}(x, \mathfrak{N}_g, \mathfrak{N}_{\text{noise}}, \mathbf{W}) = \sum_{m \in \mathfrak{T}_x} \lambda_m(x, \mathfrak{N}_g, \mathfrak{N}_{\text{noise}}) f_m(x; \mathbf{W})$$



**Theorem 1.** (*Reward Estimation Consistency*). If the reward function  $\mathcal{R}$  is differentiable,  $\mathcal{F}_\theta$  is well-converged, and  $\hat{z}_i := z_i - \epsilon \cdot \nabla_{z_i}(-\mathcal{R}(\mathcal{M}_\phi(z_i, y_i = 1), e))$  for some small  $\epsilon$ , then  $\mathcal{R}(\hat{x}_i, e) > \mathcal{R}(x_i, e)$ , where  $\hat{x}_i = \mathcal{M}_\phi(\hat{z}_i, y_i = 1)$ . (Proof in Appendix **B.1**)

**Theorem 2.** (*Inefficiency of single detector in handling evolving balance data*). Suppose a feature space  $\mathfrak{X} \subset \mathbb{R}^P$  contains  $U_n$  normal clusters and  $U_a$  anomalous clusters, where each cluster  $u$ -th  $\in [U_n + U_a]$  is modeled as a Gaussian distribution  $\mathcal{N}(\mu_u, \sigma^2 \mathbf{I}_P)$ . Let  $V_{cluster}$  be the cluster's volume and  $\Lambda$  be the total overlapping volume between normal and anomalous clusters, where the number of anomalous data points is equal to the number of normal data points, the training loss  $\mathcal{L}_{train}(\mathcal{W}_\kappa)$  is lower bounded by  $\frac{1}{4} \cdot \frac{\Lambda}{U_a \cdot V_{cluster} - \frac{\Lambda}{2}}$  in a case of linear  $\mathcal{W}_\kappa$ . (Proof in Appendix **B.2**)

**Theorem 3.** (*MoME efficiently handles evolving balance data*). Let  $\mathcal{L}_{test}(\mathfrak{F})$  and  $\mathcal{L}_{test}(\mathcal{W}_\kappa)$  represent the expected error on the test set for the Mixture of Mamba Experts (MoME) model and a single detector, respectively. For any value of  $\Lambda$ , employing MoME with  $\{f_1, f_2, \dots, f_M\}$  guarantees that the minimum expected error on the training set is  $\mathcal{L}_{train}(\mathfrak{F}) = 0$  and the expected error on the test set satisfies  $\mathcal{L}_{test}(\mathfrak{F}) \leq \mathcal{L}_{test}(\mathcal{W}_\kappa)$ . (Proof in Appendix **B.3**)

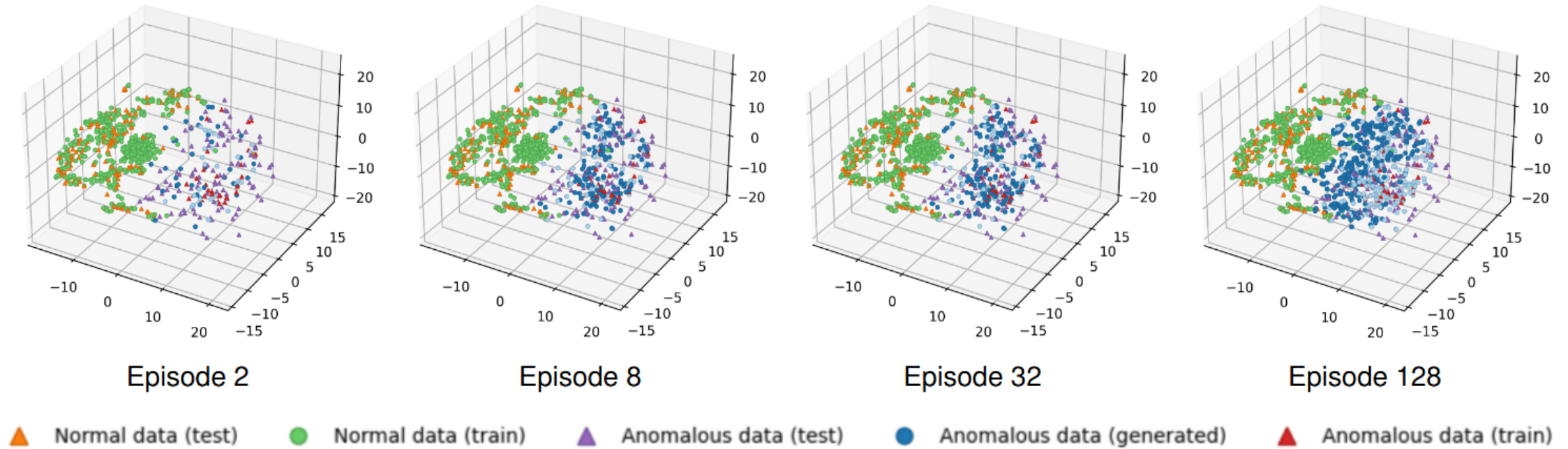


Figure 2: The distribution of the evolving training set  $\mathcal{D}_e^{train}$  and test set  $\mathcal{D}^{test}$  is visualized using the Cardiotocography dataset, one of the 57 datasets from ADBench, generated by Swift Hydra in each episode. The data points are dimensionally reduced using T-SNE (van der Maaten & Hinton, 2008). Note that the light blue points represent the generated datapoints from previous episodes, providing insight into the trend of generating anomalous data across episodes.

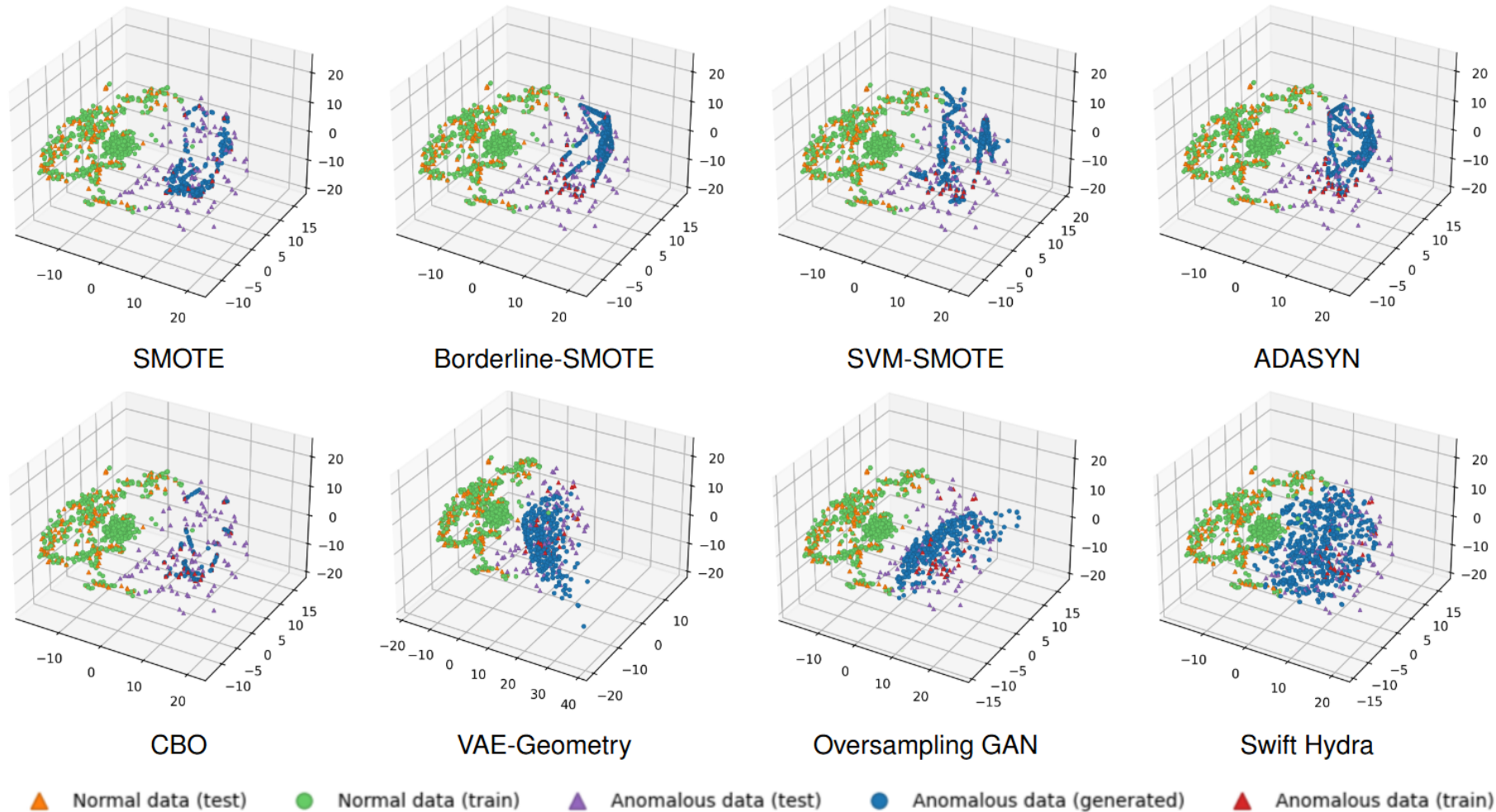


Figure 3: The distribution of the evolving training set  $\mathcal{D}_e^{train}$  and test set  $\mathcal{D}^{test}$  is visualized using the Cardiocography dataset, one of the 57 datasets from ADBench, generated by the oversampling methods in our baselines. The data points are dimensionally reduced using T-SNE (van der Maaten & Hinton, 2008).

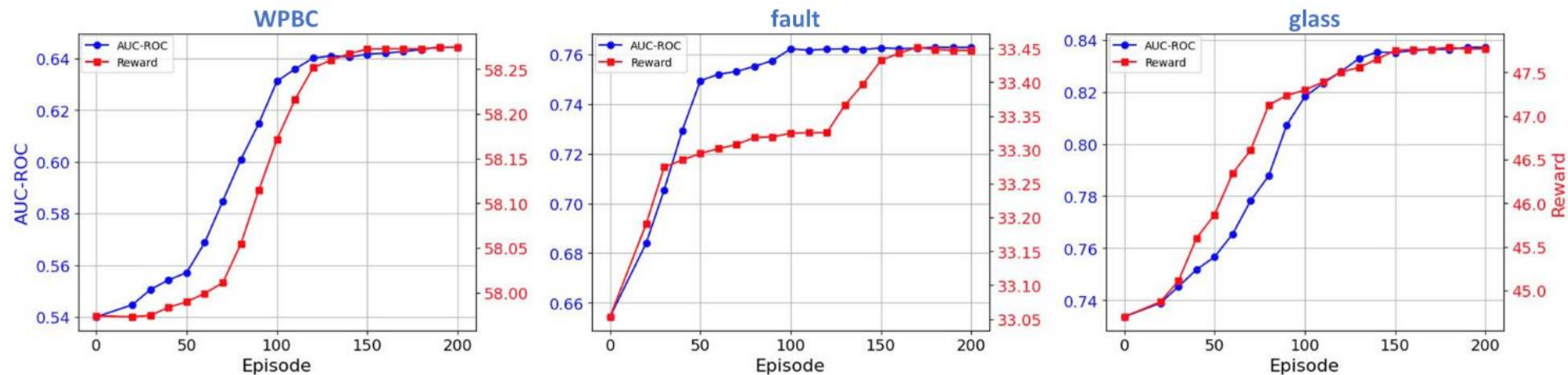


Figure 4: The performance of the RL-Agent is represented by the reward (right y-axis), while the performance of the Mamba-based Detector (single model) is measured by AUC-ROC (left y-axis). Both metrics are plotted against the number of episodes (x-axis) across three challenging datasets from ADBench. Note that both reward and AUC-ROC are averaged over multiple roll-outs.

Methods	DTE		Rejex		ADGym		Swift Hydra (Single)		Swift Hydra (MoME)	
	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF
Train/Test Ratio (40/60%)	0.82	4.02	0.78	3.89	0.86	6.12	0.91	13.11	<b>0.93</b>	4.01
Train/Test Ratio (30/70%)	0.80	4.13	0.77	4.09	0.82	7.03	0.90	14.38	<b>0.91</b>	4.79
Train/Test Ratio (20/80%)	0.79	4.31	0.76	4.22	0.79	8.17	0.87	16.13	<b>0.90</b>	5.22
Train/Test Ratio (10/90%)	0.78	4.42	0.74	4.39	0.77	9.14	0.86	18.52	<b>0.87</b>	5.84
TIF = Total Inference Time (Seconds)										

Table 1: The performance of Swift Hydra and the baselines on the ADBench is evaluated based on two criteria: AUC-ROC and total inference time (TIF). Here, the AUC-ROC is the average calculated across all 57 datasets, while the TIF represents the total time the model takes to predict all data points across all datasets. We vary the train/test ratios to illustrate how the size of the training data impacts the performance. The best AUC-ROC values are highlighted.

Methods	Transformer		Mamba		Swift Hydra(Transformer)		Swift Hydra(Mamba)	
	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF
Train/Test Ratio (40/60%)	0.78	5.18	0.80	3.54	0.90	7.96	<b>0.93</b>	4.01
Train/Test Ratio (30/70%)	0.75	6.07	0.77	3.62	0.88	8.79	<b>0.91</b>	4.79
Train/Test Ratio (20/80%)	0.72	7.10	0.73	4.11	0.86	9.91	<b>0.90</b>	5.22
Train/Test Ratio (10/90%)	0.69	8.22	0.71	4.63	0.82	10.36	<b>0.87</b>	5.84
TIF = Total Inference Time (Seconds)								

Table 5: Comparison of Vanilla Transformer and Vanilla Mamba with Transformer and Mamba enhanced by the Self-Reinforcing Module in Swift Hydra, evaluated based on AUC-ROC and Total Inference Time (TIF).

Methods	Transformer (VAE-Geometry)		Mamba (VAE-Geometry)		Swift Hydra(Transformer)		Swift Hydra(Mamba)	
	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF	AUCROC	TIF
Train/Test Ratio (40/60%)	0.83	5.23	0.85	3.40	0.90	7.96	<b>0.93</b>	4.01
Train/Test Ratio (30/70%)	0.80	6.11	0.83	3.58	0.88	8.79	<b>0.91</b>	4.79
Train/Test Ratio (20/80%)	0.78	7.02	0.81	4.02	0.86	9.91	<b>0.90</b>	5.22
Train/Test Ratio (10/90%)	0.75	8.38	0.78	4.65	0.82	8.36	<b>0.87</b>	5.84
TIF = Total Inference Time (Seconds)								

Table 6: AUC-ROC performance of various backbone models when applying oversampling techniques

The background of the slide is a photograph of a modern, multi-story building with a complex facade of windows and structural elements. The entire image is covered with a semi-transparent blue overlay. The text is positioned in the center-left area of the image.

**UF**

**Herbert Wertheim  
College of Engineering**

*Department of Computer & Information  
Science & Engineering*

**UNIVERSITY of FLORIDA**

---

POWERING THE NEW ENGINEER TO TRANSFORM THE FUTURE