SemantiGuard: AI-Driven Cyber Threat Intelligence for 6G Semantic Communications

Evram Magdy Mikhail*, Syed T. Shah†, Hassan Malik‡, Walid M. Saad§, and Mahmoud A. Shawky†,§
*Faculty of Computer Science & Informatics, German International University, Cairo, Egypt
Email: evram.mikhail@student.giu-uni.de
†School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK
Email: {syed.shah, mahmoud.a.shawky-ahmed}@essex.ac.uk
‡School of Computing Sciences, University of East Anglia, Norwich, UK
Email: Hassan.malik@uea.ac.uk
§The Egyptian Technical Research and Development Center, Cairo, Egypt
Email: walid@sce.carleton.ca

***Abstract*—As 6G networks shift from bit-oriented communication toward Semantic Communication (SemCom), the physical layer becomes vulnerable to a new type of threat called Semantic Adversarial Attacks. In these attacks, adversaries generate precise, low-energy perturbations to alter the meaning of transmitted data while evading detection by traditional energy-based security systems. This paper presents SemantiGuard, a cyber threat intelligence (CTI) framework that uses deep joint source-channel coding (Deep JSCC) with unsupervised geometric anomaly detection mechanism to secure the SemCom systems by monitoring the high-dimensional latent manifold, our framework is able to detect the legitimate signals as well as the malicious gradient-optimized perturbations. Using a developed simulation environment, we demonstrate that SemantiGuard achieves more than 84% detection accuracy even in high noise radio environments. Our findings highlights the importance of spatial-aware security layers to ensure the integrity of the critical control messages in the future of autonomous and energy sustainable 6G network architectures.**

I. Introduction

Telecommunications systems, like fourth and fifth generation (4G/5G) networks, depend on Shannon's separation theorem, where source coding is used for compression, channel coding is used for error correction can be designed independently without losing optimality to minimize bit error rate (BER). In 6G network vision, deep joint source-channel coding (Deep JSCC) is introduced, where deep neural networks (DNNs) are used to map data directly to continuous channel symbols [1]. This approach prioritizes the "meaning" (semantics) of the message rather than bit perfect reconstruction, significantly improving bandwidth efficiency. While Deep JSCC improves efficiency, it introduces the "Black Box" vulnerability. Deep neural networks are inherently susceptible to adversarial examples, inputs with undetectable perturbations designed to mislead the model [2]. In the RF domain, an adversary can transmit a low-power signal δ such that the receiver decodes a critical command (e.g., "SHUTDOWN") instead of the original message ("STATUS_NORMAL"). Because δ is optimized to be subtle, it hides below the thermal noise floor, rendering traditional energy detectors and signal-to-noise ratio (SNR) based defenses ineffective.

In this work, we propose SemantiGuard, a cyber threat intelligence (CTI) framework that leverages the geometric properties of the semantic latent space to identify these stealthy threats. By analyzing the spatial distribution of received signal vectors, SemantiGuard distinguishes optimized adversarial perturbations from natural channel noise, ensuring the integrity of the transmitted meaning without requiring traditional energy-based detection.

1) This paper proposes an autonomous CTI layer designed specifically for 6G semantic communications that operates in parallel with the decoding process to monitor signal integrity.
2) By utilizing the isolation forest algorithm, the framework moves beyond traditional energy-based detection to identify "invisible" adversarial threats based on their unique geometric positioning within the 4D semantic latent space.
3) This work provides a formal methodology and experimental validation for detecting low-power, gradient-based adversarial perturbations that are optimized to deceive Deep JSCC transceivers while remaining hidden below the thermal noise floor.

The remainder of this paper is organized as follows: Section II defines the system and threat models. Section III details the system architecture. Section IV outlines the SemantiGuard methodology. Section V provides the implementation details, followed by the experimental results and performance analysis in Section VI. Finally, Section VII concludes the work.

II. System Model

This section defines the Deep JSCC transceiver architecture and the corresponding adversarial threat model used to evaluate the framework's resilience.

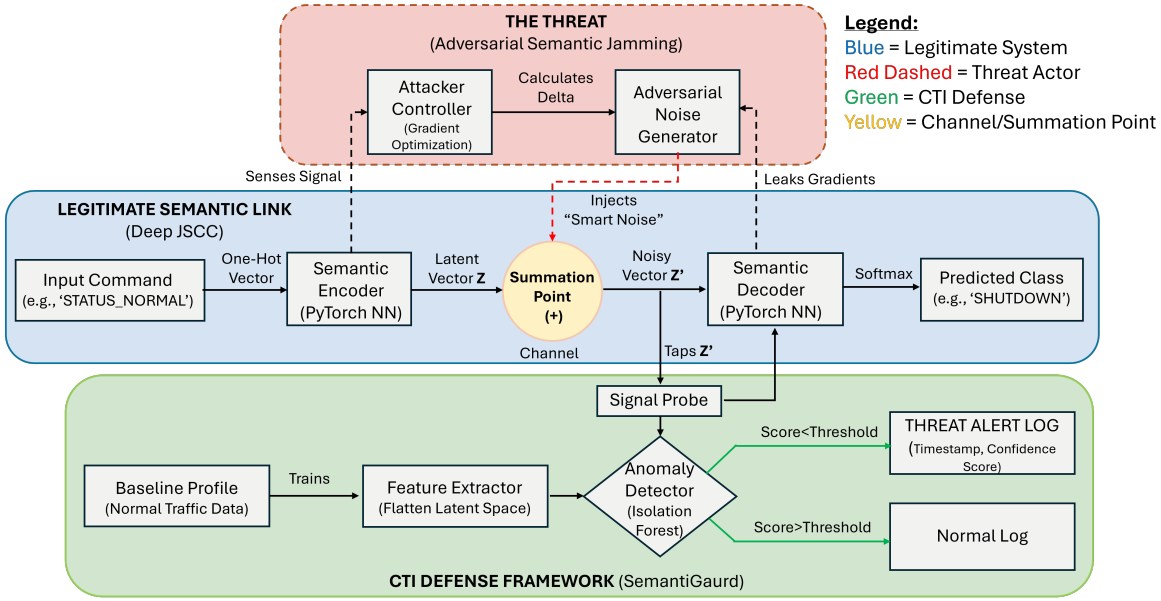

Fig. 1: High-level architecture showing the interaction between the Deep JSCC Link (Blue), the Adversarial Attacker (Red), and the CTI Defense Framework (Green).

A. Deep Joint Source-Channel Coding

We consider a point-to-point semantic communication system modeled as an autoencoder, where the transmitter and receiver are jointly optimized to communicate semantic meaning over a noisy channel. The system architecture is illustrated in Fig. 1. The Deep JSCC system consists of the following three main parts that work together to send and receive semantic data

1) Semantic Encoder: Let $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ denote the finite set of M possible semantic messages (e.g., control commands). The input message $s \in \mathcal{S}$ is represented as a one-hot encoded vector $\mathbf{u} \in \{0,1\}^M$. The semantic encoder $E_\theta(\cdot)$, parameterized by weights θ, maps the input vector $\mathbf{u}$ to a complex-valued channel input vector $\mathbf{x} \in \mathbb{C}^k$, where k is the number of channel uses allocated per message (bandwidth):

$$\mathbf{x} = E_\theta(\mathbf{u}) \tag{1}$$

To satisfy physical hardware constraints, the transmitted signal $\mathbf{x}$ is subject to an average power constraint P. The normalization layer ensures:

$$\frac{1}{k}\mathbb{E}[\|\mathbf{x}\|^2] \leq P \tag{2}$$

2) Wireless Channel: The signal propagates through an additive white Gaussian noise (AWGN) channel. The received signal $\mathbf{y} \in \mathbb{C}^k$ is given by:

$$\mathbf{y} = h\mathbf{x} + \mathbf{n} \tag{3}$$

where $h \in \mathbb{C}$ represents the channel gain (assumed to be $h = 1$ for static AWGN scenarios) and $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2\mathbf{I}_k)$ is the complex Gaussian noise vector with variance σ^2. The SNR is defined as $\text{SNR} = P/\sigma^2$.

3) Semantic Decoder: The receiver employs a semantic decoder $D_\phi(\cdot)$, parameterized by weights ϕ, to reconstruct the probability distribution of the transmitted message from the noisy signal $\mathbf{y}$:

$$\hat{\mathbf{p}} = D_\phi(\mathbf{y}) \tag{4}$$

where $\hat{\mathbf{p}} = [\hat{p}_1, \ldots, \hat{p}_M]^T$ is the output softmax probability vector, with $\hat{p}_i$ representing the probability that the i-th message was sent. The estimated message $\hat{s}$ is selected via maximum likelihood:

$$\hat{s} = \underset{i \in \{1,\ldots,M\}}{\text{argmax}}\ \hat{p}_i \tag{5}$$

4) Optimization Objective: The entire autoencoder is trained end-to-end to minimize the cross-entropy loss function $\mathcal{L}_{CE}$, which measures the divergence between the true distribution $\mathbf{u}$ and the predicted distribution $\hat{\mathbf{p}}$:

$$\mathcal{L}_{CE}(\theta, \phi) = -\sum_{i=1}^{M} u_i \log(\hat{p}_i) \tag{6}$$

B. Threat Model: Semantic Adversarial Attack

We assume a White-Box attack scenario where the adversary has full knowledge of the encoder/decoder parameters (θ, ϕ) but cannot modify them. Attacker's objective is to transmit a perturbation signal $\boldsymbol{\delta}$ such that the receiver misclassifies the message into a target class t (e.g., forcing a "Normal" status to be read as "Shutdown").

The adversarial perturbation $\boldsymbol{\delta}$ is computed by solving the following constrained optimization problem:

$$\min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|^2 \quad \text{s.t.} \quad \text{argmax}(D_\phi(\mathbf{x} + \boldsymbol{\delta})) = t \tag{7}$$

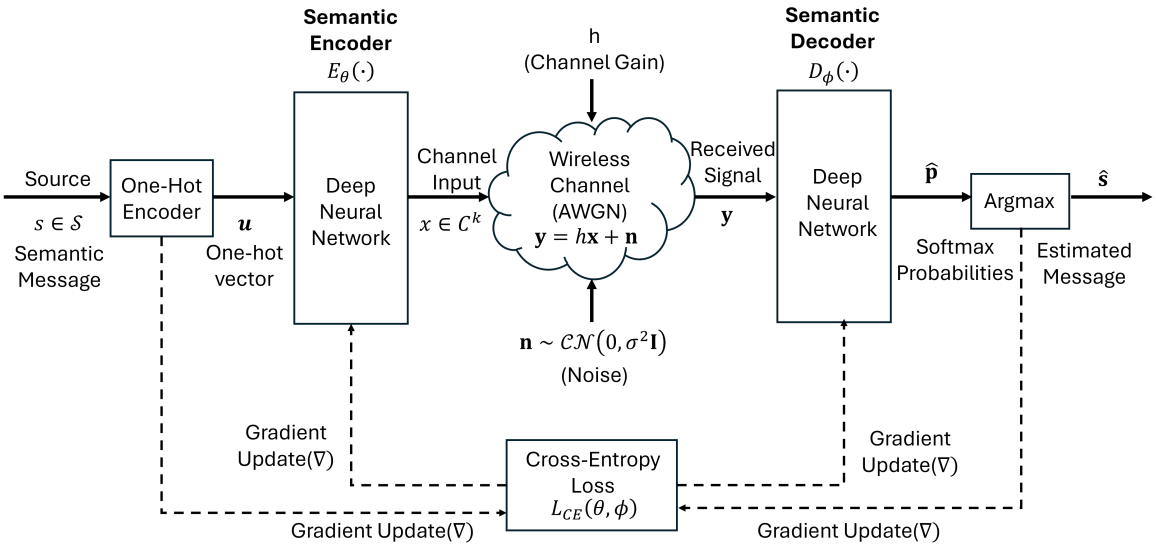

Fig. 2: System Architecture: The SemantiGuard CTI module operates in parallel with the Semantic Decoder.

The constraint $\|\boldsymbol{\delta}\|^2$ ensures the attack power is minimized to remain stealthy. To solve this, we employ the projected gradient descent (PGD) method [3]. At each iteration j, the perturbation is updated by moving in the direction of the loss gradient with respect to the input:

$$\boldsymbol{\delta}_{j+1} = \Pi_\epsilon \left(\boldsymbol{\delta}_j - \alpha \cdot \nabla_{\boldsymbol{\delta}} \mathcal{L}_{CE}(D_\phi(\mathbf{x} + \boldsymbol{\delta}_j), \mathbf{t}_{target}) \right) \quad (8)$$

where α is the step size, $\mathbf{t}_{target}$ is the one-hot vector for the target class, and Π_ϵ projects the perturbation onto the ϵ-ball to satisfy the power constraint $\|\boldsymbol{\delta}\|_\infty \leq \epsilon$.

This formulation generates a semantic jamming signal that is energetically indistinguishable from channel noise $\mathbf{n}$ but semantically potent enough to deceive the decoder D_ϕ.

III. System Architecture and Operational Flow

The proposed 6G semantic communication system integrates a Deep JSCC transceiver with a dedicated CTI module. The system is designed to not only compress and transmit semantic information efficiently but also to autonomously detect and reject adversarial semantic jamming attacks. The complete operational flow is illustrated in Fig. 2.

A. System Overview

The system architecture consists of three main subsystems: the *Legitimate Semantic Link*, consisting of the legitimate input commands, semantic encoding and decoding of transmitted and received commands, channel simulation and the output (predicted class); the *Threat Actor*, which attempts to disrupt communication by injecting gradient-based jamming (smart noise) to the encoded legitimate commands; and the *CTI Defense Framework*, that probe the received commands at decoder side and scan the latent space for potential anomalies.

B. Operational Phases

The system operates in three distinct phases to ensure high semantic accuracy and security:

1) Phase 1: Deep JSCC Training: The Semantic Encoder and Decoder are trained to learn the optimal encoding and decoding parameters before the real-time operation.

- **Dataset:** Training is done using synthetic dataset generated for the study. The dataset has four distinct command classes; each command class is represented by 50 samples ensuring an even distribution across the four classes. To have a dataset of 200 samples used for the training phase.
- **Neural Architecture:** The *Encoder* is designed to receive 10-dimensional one-hot input vector and encode them into a 4-dimensional latent vector. Using dense hidden layer of 8 neurons with ReLU activation function to add nonlinearity. The *Decoder* mirrors this topology to reconstruct the original command.
- **Training Procedure:** Adam optimizer with 400 epochs is used to train the system end-to-end. Different channel noise conditions ($\sigma \in [0.1, 0.8]$) are used during training process to let the model capture varying channel characteristics to minimize cross-entropy loss and ensure robustness.

2) Phase 2: Baseline Profiling: after training the Encoder, its weights are frozen, to build a "Reference Map" for the CTI module.

- **Generation:** The frozen Encoder is used to generate 600 new semantic vectors by passing random commands through the wireless channel.
- **Profiling:** These 600 samples represent the "Normal" semantic distribution, which defines the safe clusters within the latent space.
- **Defense Training:** An *Isolation Forest* algorithm is trained on this baseline data. . It is used to establish a density threshold; any signal falling into the low-density regions between valid clusters is flagged as anomalous.

3) Phase 3: Real-Time Operation & Threat Detection: In the live environment, every incoming transmission follows the below pipeline:

1) **Transmission:** The Transmitter encodes the received command (e.g.`CRITICAL_HEAT`) into a latent vector $\mathbf{z}$.
2) **Channel & Attack:** After encoding, the signal travels through the channel with AWGN noise and adversarial interference. To push the signal toward a false class, the attacker may inject a perturbation vector $\boldsymbol{\delta}$, optimized via gradient descent (Smart Noise).
3) **Signal Probe:** The Receiver "taps" the incoming noisy vector $\mathbf{z}'$ before decoding, for potential threat detection.
4) **CTI Validation:** The vector $\mathbf{z}'$ is analyzed by the Isolation Forest:
   - *Safe:* If the anomaly score exceeds the threshold (indicating high density), the signal is forwarded to the Decoder.
   - *Threat:* If the score falls below the threshold, the CTI module triggers a "Semantic Jamming Alert" and discards the packet.
5) **Decoding:** Validated signals are processed by the Decoder to recover the final command.

IV. Methodology: SemantiGuard Framework

This section describes how the SemantiGuard framework is built, including its anomaly detection logic and the specific neural network structure used for the simulation.

A. Geometric Anomaly Detection

SemantiGuard inspects the geometry of the received signal in the latent space, unlike the traditional security mechanisms that depend on the received signal power (Energy Detection) for threat detection. To make our system robust against different adversarial examples that successfully cross the decision boundary of the Decoder, an isolation forest algorithm is trained on legitimate traffic patterns, to ensure that the adversarial examples are often land in low-density regions (manifolds) of the high-dimensional latent space.

B. Robust Noise Injection Training

The isolation forest model was trained on the baseline dataset generated with injected Gaussian noise ($\sigma = 0.3$) to avoid flagging natural channel noise as an attack (False Positives), This ensures the model is able to distinguish between noisy legitimate signals and optimized adversarial perturbations.

V. Implementation Details

This section provides the technical setup of the experiment, detailing the software tools and the specific neural network layers used to build the semantic communication link.

A. Software Stack

The simulation environment was developed using Python 3.9. The core semantic transceiver was implemented in PyTorch, while the CTI module utilized **Scikit-Learn** for the isolation forest algorithm. Data analysis and visualization were performed using Pandas and Seaborn.

B. Neural Network Architecture

The Deep JSCC transceiver consists of fully connected (dense) layers with the following structure, see Fig. 3.

- **Encoder (E_θ):**
  - Input Layer: 10 neurons (One-hot encoded commands)
  - Hidden Layer: 8 neurons (ReLU Activation)
  - Output Layer: 4 neurons (Latent Semantic Space)
- **Decoder (D_ϕ):**
  - Input Layer: 4 neurons (Received Noisy Symbols)
  - Hidden Layer: 8 neurons (ReLU Activation)
  - Output Layer: 4 neurons (Softmax Probability)

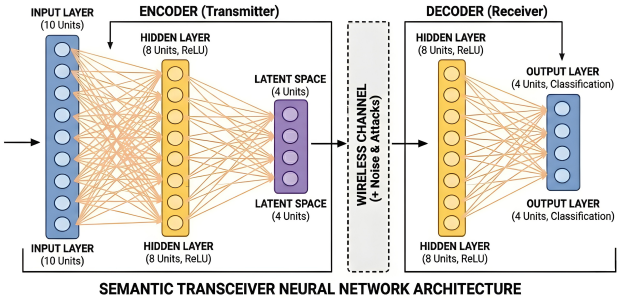

Fig. 3: Deep JSCC Autoencoder Architecture. The Encoder maps 10-bit one-hot vectors to a 4D continuous latent space, which is then monitored by the SemantiGuard CTI module.

VI. Experimental Results

This section presents the simulation results, evaluating how the system performs under different noise levels and visualizing how the semantic data is organized and protected in the latent space.

A. Simulation Setup

The simulation was implemented using PyTorch. The channel noise σ was varied from 0.1 to 0.8 to simulate different Signal-to-Interference-plus-Noise Ratio (SINR) conditions. Table I details the conversion between noise variance and SINR based on the measured average signal power ($P_S \approx 5.22$).

TABLE I: Noise to SINR Conversion

Noise (σ)	Variance (σ^2)	SINR (dB)	Condition
0.1	0.01	27.18	Excellent
0.3	0.09	17.64	Good
0.6	0.36	11.62	Boundary
0.8	0.64	9.12	Noisy

B. Latent Space Visualization

The 2D projection (via principal component analysis (PCA)) of the semantic latent space is presented in Fig. 4. The legitimate commands form four distinct clusters, While the adversarial attacks (marked in red) are observed to deviate from the centroids of these clusters in that representation.

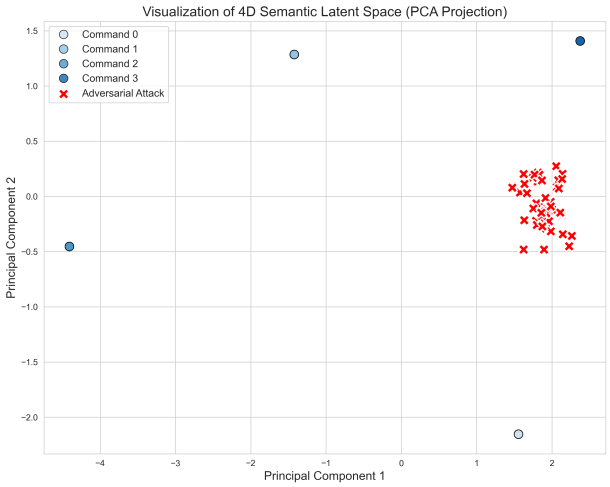

Fig. 4: Visualization of 4D Semantic Latent Space. Adversarial examples (Red) deviate from the dense normal clusters (Blue).

C. Performance Analysis

A balanced synthetic dataset of 500 legitimate semantic transmissions and 500 optimized adversarial jamming attacks per noise level is used to assess the detection performance and robustness of the SemantiGuard framework against different SINR conditions. As shown in Fig. 5 and Fig. 6:

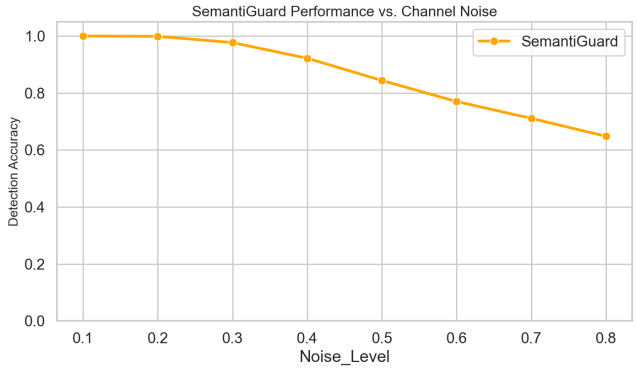

Fig. 5: SemantiGuard System Accuracy vs. Channel Noise. The system maintains perfect reliability ($\approx 100\%$) in good channel conditions ($\sigma \leq 0.3$) and degrades gracefully to $\approx 65\%$ as noise becomes severe.

As shown in Fig. 5 and Fig. 6:

- **Robust Zone** ($\sigma \leq 0.3$)**:** The curves for $\sigma = 0.1$ and 0.3 have perfect rectangularity (AUC=1.00). In standard operating conditions, the geometrical separation between the legitimate signal and adversarial perturbations is perfect, allowing for 100% detection accuracy.
- **Transition Zone** ($\sigma = 0.5$)**:** The AUC still excellent at **0.95**. The curve begins to be slightly rounded, it indicates the safety margins are becoming smaller, but the classifier can maintain high precision.
- **High-Noise Resilience** ($\sigma = 0.8$)**:** Even in the extremely high-noise scenario (SINR $\approx$ 9dB), the AUC achieved **0.84**. Although the absolute detection accuracy decreased to $\approx 65\%$ (due to the decision threshold being fixed), the high AUC confirms that the model's discriminating capability is still maintained and it does not drop to random performance (AUC ≈ 0.5).

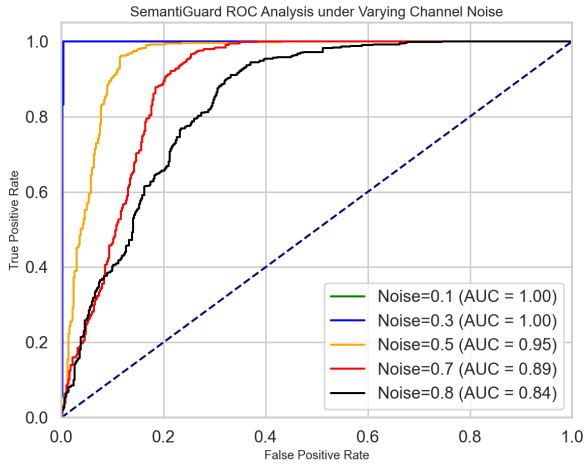

Fig. 6: Receiver Operating Characteristic (ROC) analysis under varying channel noise ($\sigma \in [0.1, 0.8]$). The system exhibits perfect separability (AUC=1.00) in standard conditions and maintains strong predictive power (AUC=0.84) even under extreme noise.

D. Deep JSCC Training Convergence

The semantic transceiver is trained end to end with the Adam optimizer. The learning rate is set to $\eta = 0.01$ and the training process is shown over 400 epochs in Fig. 7 The Cross-Entropy Loss decreases to a constant low level over the initial 100 epochs where the Encoder learned to separate the four commands into distinct spatial regions.

E. Latent Space Structural Analysis

The 4-dimensional latent vector x are visualized to validate the geometry of the learned semantic representation; we used two techniques for this representation:

1) 3D Latent Projection: The four semantic classes (Normal, Warning, Critical, Shutdown) form tight, well-separated clusters as shown from the 3D PCA projection of the sematic symbols in Fig. 8. This spatial separation represents the fundamental property that SemantiGuard leverages for anomaly detection.

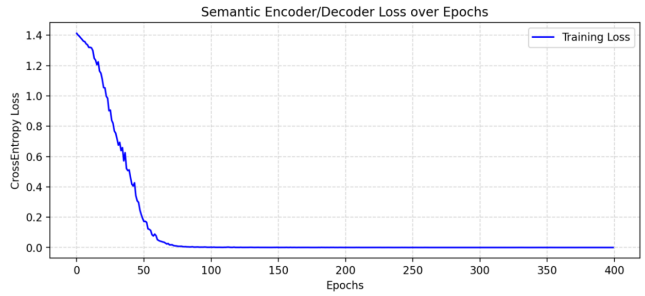

Fig. 7: Training Convergence. The Cross-Entropy Loss minimizes and stabilizes, confirming the model has learned the optimal semantic constellation for the channel conditions.

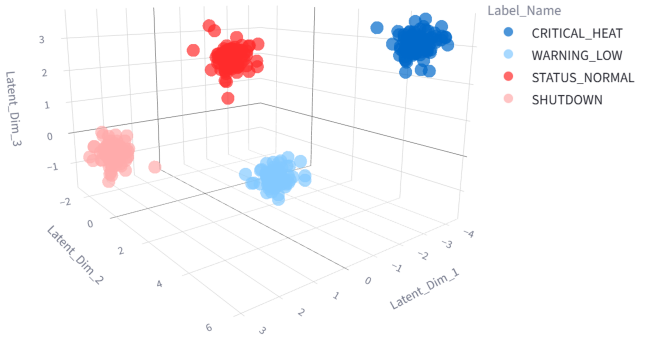

Fig. 8: 3D PCA Projection of the 4D Latent Space. Distinct clusters represent different semantic commands.

2) Parallel Coordinates Analysis: Each command class (encoded by color) follows a unique and tightly clustered trajectory across the four latent dimensions. For example, the "Shutdown" command (Class 3, Red) is represented by a geometric values completely distinct from the "Normal" status (Class 0, Green). This clear separation confirms that the Encoder has successfully learned to map the different semantic meanings to non-overlapping regions of the latent space.

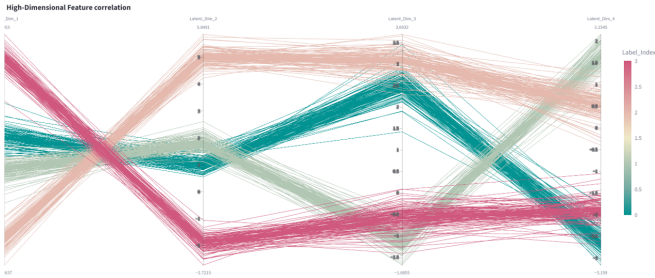

Fig. 9: Parallel Coordinates Plot of 4D Latent Vectors.

F. Semantic Space Monitor (Constellation)

Similarly to QAM constellation diagrams in traditional RF systems, Fig. 10 visualizes the "Semantic Constellation" monitored by the CTI module. The dashed boundaries represent the decision surfaces of the Isolation Forest. Symbols falling inside the clusters are accepted as valid, while points appearing in the low density zones "dead zones" (typically adversarial perturbations) are flagged as threats.

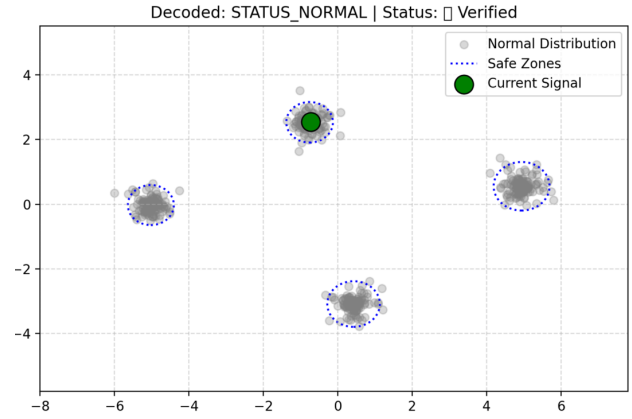

Fig. 10: Semantic Space Monitor. The blue regions indicate the 'Safe Zones' learned by the CTI.

VII. Conclusion

In this work, we have shown that the new geometric analysis approach of anomaly detection is robust against gradient-based perturbations that are tailored to attack the Semantic Communications system, this new approach mitigates vulnerabilities against sophisticated adversarial jamming attacks in future 6G networks. The experimental results showed the effectiveness and the resilience of the SemantiGuard framework under different conditions of SINR as it consistently maintains high accuracy under noisy channel, efficiently isolating the low-power and gradient-based perturbations from normal command messages. This work shows the necessity of incorporating CTI at physical layer for future communication networks that will operate in a Semantic Communication framework. We plan to test the developed architecture on software defined radio (SDR) platforms Additionally, we aim to apply this framework in multi-user with mobility network environment to investigate the framework scalability and develop adaptive training techniques to handle more complex adversarial strategies in diverse 6G application scenarios.

References

[1] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, "Deep joint source-channel coding for wireless image transmission," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, U.K., 2019, pp. 4774-4778, doi: 10.1109/ICASSP.2019.8683463.

[2] M. Sadeghi and E. G. Larsson, "Adversarial attacks on deep-learning-based radio signal classification," *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 213–216, Feb. 2019, doi: 10.1109/LWC.2018.2867459.

[3] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[4] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Pisa, Italy, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17.

[5] L. X. Nguyen, A. D. Raha, P. S. Aung, D. Niyato, Z. Han, and C. S. Hong, "A contemporary survey on semantic communications: Theory of mind, generative AI, and deep joint source-channel coding," *IEEE Communications Surveys & Tutorials*, vol. 28, pp. 2377-2417, 2026, doi: 10.1109/COMST.2025.3616973.

[6] A. D. Raha, A. Adhikary, M. Gain, Y. Park, W. Saad, and C. S. Hong, "DD-JSCC: Dynamic deep joint source-channel coding for semantic communications," in *Proc. IEEE Int. Conf. Communications (ICC)*, Montreal, QC, Canada, 2025, pp. 3754-3759, doi: 10.1109/ICC52391.2025.11162089.

[7] H. Wu, Y. Shao, C. Bian, K. Mikolajczyk, and D. Gündüz, "Deep joint source-channel coding for adaptive image transmission over MIMO channels," *IEEE Transactions on Wireless Communications*, vol. 23, no. 10, pp. 15002–15017, Oct. 2024, doi: 10.1109/TWC.2024.3422794.

[8] J. Xu, T.-Y. Tung, B. Ai, W. Chen, Y. Sun, and D. Gündüz, "Deep joint source-channel coding for semantic communications," *IEEE Communications Magazine*, vol. 61, no. 11, pp. 42-48, Nov. 2023, doi: 10.1109/MCOM.004.2200819.

[9] D. B. Kurka and D. Gündüz, "DeepJSCC-f: Deep joint source-channel coding of images with feedback," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 178-193, May 2020, doi: 10.1109/JSAIT.2020.2987203.