Intrinsic Explanation Stability under Adversarial Perturbations in Self-Explaining Neural Networks for IoT Firmware Malware Detection

Michael Georgiades*, Iacovos Ioannou†, Hasan Mahmood Aminul Islam‡
Kin-Hon Ho§, Yun Hou¶, Andreas Gregoriades‖
*Department of Computer Science, Neapolis University & Infostrada Communications, Cyprus
†Department of Computer Science, Philips University; CYENS & UCY NETRL, Cyprus
‡Department of Computer Science and Engineering, East West University, Bangladesh
§Department of Business Administration, Hong Kong Shue Yan University, Hong Kong SAR
¶Department of Computer Science, Hang Seng University of Hong Kong, Hong Kong SAR
‖Department of Communication and Marketing, Cyprus University of Technology, Cyprus
m.georgiades@nup.ac.cy, iacovos.ioannou@gmail.com, hasan.mahmood@ewubd.edu
khho@hksyu.edu, aileenhou@hsu.edu.hk, andreas.gregoriades@cut.ac.cy

***Abstract*—Self-Explaining Neural Networks (SENNs) provide intrinsic explanations by decomposing predictions into latent concepts and input-dependent relevance weights. This paper studies the stability of SENN explanations for byte-level IoT firmware malware detection under adversarial perturbations. Firmware binaries are represented as 32×32 grayscale images derived from raw byte values, and robustness is evaluated using FGSM, PGD, and MI-FGSM attacks. Results reveal a separation between representation stability and decision vulnerability: classification accuracy degrades sharply, while latent concepts, relevance weights, and concept-relevance explanations remain highly aligned. Concept deletion/insertion tests further show that the learned concepts carry predictive signal, while a matched CNN baseline indicates that high feature stability is not unique to SENNs. A fair input-space comparison with SHAP, Integrated Gradients, and Grad-CAM shows that projected SENN explanations provide complementary diagnostic information rather than dominating post-hoc attribution methods. These findings suggest that explanation stability can support firmware-malware analysis when combined with conventional robustness testing and analyst review.**

***Index Terms*—Explainable Artificial Intelligence, Self-Explaining Neural Networks, IoT Security, Firmware Malware Detection, Adversarial Robustness**

I. Introduction

The rapid proliferation of the Internet of Things (IoT) and Internet of Medical Things (IoMT) devices has significantly expanded the attack surface of modern cyber-physical ecosystems. Firmware-level compromise, supply-chain manipulation, and malicious over-the-air (OTA) updates represent particularly severe threats, as tampered firmware can persist across device lifecycles and evade traditional network monitoring mechanisms. Consequently, intrusion detection and malware analysis systems for IoT environments must not only achieve high predictive performance but also provide transparency and robustness against adversarial manipulation [1], [2].

ISBN 978-3-903176-82-9

Deep learning models have recently demonstrated strong performance for IoT malware detection by learning structural patterns directly from firmware binaries and network traffic [3], [4]. In particular, convolutional neural networks can capture spatial byte-pattern structures when firmware binaries are represented as grayscale images. However, most existing approaches operate as black-box classifiers, providing limited insight into the internal reasoning behind model predictions.

To address this limitation, Explainable Artificial Intelligence (XAI) techniques have been increasingly incorporated into intrusion detection systems. Several works have explored explainable frameworks for IDS analysis and anomaly detection [5]–[7]. Post-hoc explanation techniques such as SHAP and LIME have also been applied to neural intrusion detection models to highlight influential features and improve analyst trust [8]. More recent studies evaluating the reliability of explainable methods in IDS pipelines highlight challenges associated with inconsistent feature-relevance estimation across explanation techniques [9], [10].

The growing interest in explainable intrusion detection has also produced several explainable IDS architectures for IoT environments [11]–[14]. In parallel, surveys such as [15], [16] provide comprehensive overviews of explainable IDS techniques and identify interpretability reliability as a key research challenge. While these approaches improve transparency, they typically rely on post-hoc explanations generated after the prediction process.

Concept-based explainable models offer an alternative paradigm by embedding interpretability directly into the model architecture. Concept-level reasoning has recently been advocated as a promising direction for interpretable machine learning [17]. Self-Explaining Neural Networks (SENNs) operationalize this idea by decomposing predictions into latent concepts and their input-dependent relevance weights, thereby producing explanations as an intrinsic component of the infer-

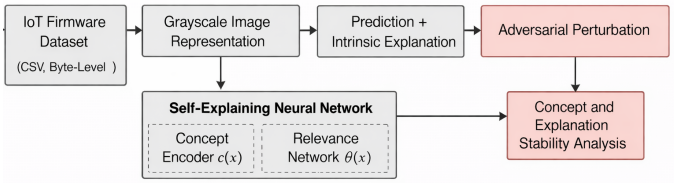


Fig. 1: Overview of the proposed framework. Firmware binaries are transformed into grayscale image representations and processed by a Self-Explaining Neural Network that jointly learns latent concepts and their relevance. Adversarial perturbations are applied to evaluate the stability of predictions and intrinsic explanations.

ence process [18].

Prior work on adversarial robustness has shown that predictive accuracy and internal representations may degrade differently under perturbations, while XAI studies in intrusion detection have mainly evaluated post-hoc explanations after prediction. However, fewer studies examine whether intrinsic concept-based explanations remain stable when the classifier itself becomes unreliable. This gap motivates our focus on the relationship between latent concept stability, decision vulnerability, and input-space explanation stability in IoT firmware malware detection.

In this work, we study SENN explanation stability for IoT firmware malware detection using grayscale byte-image representations and FGSM, PGD, and MI-FGSM perturbations. The results show that accuracy can collapse while latent concepts and concept-relevance explanations remain highly aligned, revealing a separation between representation stability and decision vulnerability. We also project SENN explanations back to the byte-image input space for a fair comparison with SHAP, Integrated Gradients, and Grad-CAM.

A. Contributions

The main contributions are:

- Evaluation of SENN explanation robustness under FGSM, PGD, and MI-FGSM attacks.
- Evidence of a separation between stable latent representations and vulnerable decisions.
- A fair input-space comparison between projected SENN explanations and SHAP, Integrated Gradients, and Grad-CAM.
- Compact controls using concept deletion/insertion faithfulness and a matched-capacity CNN baseline.

II. Problem Formulation

This section formalizes the SENN-based firmware detection framework, consisting of firmware representation, intrinsic concept-based inference, adversarial perturbation generation, and stability analysis, as illustrated in Fig. 1.

A. IoT Firmware Representation

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote an IoT firmware dataset containing N labeled samples, where $x_i \in \mathbb{R}^{1024}$ represents a byte-level vector extracted from a firmware binary and $y_i \in \{1, \ldots, C\}$ denotes the class label. In our setting, the dataset contains three firmware categories: benignware, malware, and hackware.

To enable structural learning using convolutional neural networks, each firmware byte vector is converted into a grayscale image representation. Each byte value is mapped to a pixel intensity in the range $[0, 255]$, producing an image

$$x \to I(x) \in \mathbb{R}^{H \times W}.$$

This transformation preserves statistical characteristics of the binary such as byte distribution and structural patterns while enabling spatial feature learning.

B. Self-Explaining Neural Network Model

The grayscale firmware representation is processed by a SENN, which jointly produces predictions and intrinsic explanations. A SENN decomposes inference into three components: a feature extractor $\phi(\cdot)$, a concept encoder $g(\cdot)$, and a relevance network $r(\cdot)$.

Given an input x, the feature extractor produces a latent representation

$$h(x) = \phi(x), \quad h(x) \in \mathbb{R}^m. \tag{1}$$

The concept encoder maps this representation into a set of K latent concepts

$$c(x) = g(h(x)), \quad c(x) \in \mathbb{R}^K, \tag{2}$$

while the relevance network estimates input-dependent concept importance weights

$$\theta(x) = r(h(x)), \quad \theta(x) \in \mathbb{R}^{C \times K}. \tag{3}$$

The final prediction is obtained through a linear aggregation of concepts

$$f(x) = \theta(x)c(x), \tag{4}$$

where $f(x) \in \mathbb{R}^C$ represents the class logits. The predicted label is

$$\hat{y} = \arg\max_c f_c(x).$$

This formulation decomposes predictions into interpretable concept activations and their associated relevance weights.

C. Intrinsic Explanation Representation

Because concept activations and relevance weights directly determine the prediction, explanations are intrinsically embedded within the inference process. The explanation vector for input x is defined as

$$e(x) = \theta_y(x) \odot c(x), \tag{5}$$

where $\odot$ denotes element-wise multiplication. The vector $e(x)$ therefore captures the contribution of each learned concept to the final decision.

D. Adversarial Perturbation

To evaluate robustness, adversarial perturbations are applied during evaluation. Given an input sample x, an adversarial example x' is generated as

$$x' = x + \delta,$$

where δ denotes a bounded perturbation applied to the byte-image representation in order to induce misclassification. Since this perturbation is applied in representation space, it should be interpreted as a controlled robustness probe rather than as a semantics-preserving executable firmware modification.

E. Concept and Explanation Stability

To quantify robustness of the learned representations and explanations, we measure their stability between clean inputs x and adversarial samples x'. Concept stability is measured using cosine similarity

$$S_c = \frac{c(x) \cdot c(x')}{\|c(x)\|\|c(x')\|}. \tag{6}$$

Similarly, relevance stability is defined as

$$S_\theta = \frac{\theta(x) \cdot \theta(x')}{\|\theta(x)\|\|\theta(x')\|}. \tag{7}$$

Finally, explanation stability is computed from the composed explanation vectors

$$S_e = \frac{(\theta(x) \odot c(x)) \cdot (\theta(x') \odot c(x'))}{\|\theta(x) \odot c(x)\|\|\theta(x') \odot c(x')\|}. \tag{8}$$

Higher similarity values indicate stronger robustness of concept representations and intrinsic explanations under adversarial perturbations.

F. Evaluation Protocol

The framework is evaluated using predictive, representation-level, and explanation-level metrics, including accuracy, logit margin, concept stability, and explanation similarity between clean and adversarial inputs.

III. Experimental Setup

This section describes the dataset, model architecture, training configuration, adversarial attack model, and evaluation metrics used to assess the robustness and explanation stability of SENNs.

A. Dataset and Preprocessing

Experiments were conducted using the *IoT Firmware Image Classification* dataset from Kaggle [19]. The dataset contains 38,887 ELF firmware samples distributed across three classes: 38,073 benignware samples, 711 malware samples, and 103 hackware samples, corresponding to approximately 97.91%, 1.83%, and 0.26% of the dataset, respectively. Each sample is constructed by extracting the first 1024 bytes of an ELF binary and mapping the byte values into grayscale pixel intensities, resulting in a fixed-length 1024-dimensional representation.

In this study, *hackware* is treated as a dataset-specific third class rather than being merged with either benignware or malware. We use the term to denote dual-use or security-oriented executable tools that may not be inherently malicious in the same sense as malware, but can be flagged as suspicious or potentially unwanted by antivirus systems. Therefore, the task is formulated as a three-class classification problem involving benignware, malware, and hackware, rather than a binary benign-versus-malicious classification task.

Following the representation described in Section II, each firmware vector is reshaped into a 32×32 grayscale image by mapping byte values to pixel intensities in the range $[0, 255]$. This byte-image representation is a compact abstraction of the firmware header/early binary content and enables the convolutional feature extractor to learn local byte-pattern regularities across benignware, malware, and hackware samples. A stratified 75/25 train–test split was applied to maintain class balance across partitions, and pixel intensities were retained in byte-intensity scale $[0, 255]$ during training and adversarial evaluation.

B. Model Architecture

We implement a convolutional SENN following the formulation introduced in Section II. The model consists of a convolutional feature extractor, a concept encoder producing latent concept activations, and a relevance network generating input-dependent concept importance scores.

The feature extractor contains two convolutional blocks followed by a fully connected projection layer. In the reported implementation, the concept encoder outputs $K = 8$ latent concepts representing internal features learned from firmware structure. The relevance network produces class-dependent relevance weights of dimension $C \times K$, where C denotes the number of classes. Predictions are computed using the linear aggregation

$$f(x) = \theta(x)c(x), \tag{9}$$

where $c(x)$ represents concept activations and $\theta(x)$ denotes the corresponding relevance weights.

C. Training Configuration

The model was trained using the Adam optimizer with a learning rate of 10^{-3} and cross-entropy loss. No explicit SENN Lipschitz or robustness regularizer was used in the reported experiments; the corresponding regularization weight was set to $\lambda_{\text{rob}} = 0$, so the model was optimized using only the supervised cross-entropy objective. Training was performed for 8 epochs using only clean samples. Adversarial samples were intentionally excluded so that the evaluation measures the natural robustness and explanation stability of the clean-trained SENN, rather than the effect of adversarial training.

D. Adversarial Attack Model

Adversarial robustness was evaluated using three white-box gradient-based attacks: FGSM, PGD, and MI-FGSM. FGSM is used as the single-step baseline attack. Given an input sample x with label y, the FGSM adversarial example x' is generated as

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(x, y)), \tag{10}$$

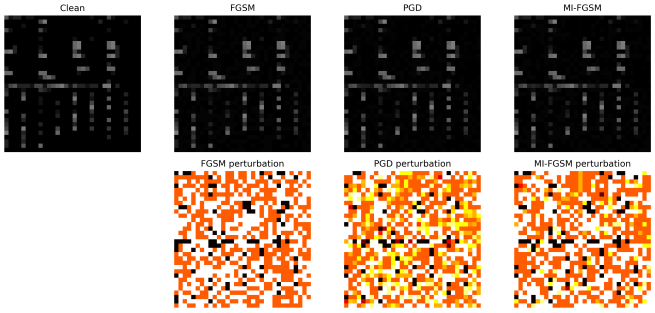


Fig. 2: Example clean and adversarial byte-image firmware representations under FGSM, PGD, and MI-FGSM at $\epsilon = 4.0$. The top row shows the clean and perturbed firmware images, while the bottom row shows the corresponding perturbation maps. The visualized perturbations illustrate that the attacks are applied in byte-image representation space rather than as executable-preserving firmware transformations.

where ϵ controls the perturbation magnitude and $\mathcal{L}$ denotes the classification loss.

In addition to the single-step FGSM formulation in Eq. (10), we also evaluate stronger iterative attacks. For PGD, adversarial examples are generated iteratively as

$$x^{t+1} = \Pi_{B_\epsilon(x)} \left(x^t + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}(x^t, y)) \right), \qquad (11)$$

where α is the step size and $\Pi_{B_\epsilon(x)}(\cdot)$ projects the perturbed sample back into the ℓ_∞ ball of radius ϵ around the original input. For MI-FGSM, the iterative update incorporates gradient momentum as

$$g^{t+1} = \mu g^t + \frac{\nabla_x \mathcal{L}(x^t, y)}{\|\nabla_x \mathcal{L}(x^t, y)\|_1}, \qquad (12)$$

$$x^{t+1} = \Pi_{B_\epsilon(x)} \left(x^t + \alpha \cdot \text{sign}(g^{t+1}) \right), \qquad (13)$$

where μ is the momentum decay factor. We evaluate $\epsilon \in \{0, 0.5, 1.0, 2.0, 4.0\}$ for all attacks. PGD and MI-FGSM use 10 iterations with $\alpha = \epsilon/4$, $\mu = 1.0$ for MI-FGSM, and clipping to $[0, 255]$. Although the perturbations are small in byte-intensity magnitude, they significantly affect the model's predictions by introducing adversarially optimized modifications to the firmware representation, as illustrated in Fig. 2.

E. Evaluation Metrics

Model behavior is evaluated using classification accuracy, concept stability (S_c), relevance stability (S_θ), latent explanation stability (S_e), concept ℓ_2 drift, and logit margin. We further include concept deletion/insertion faithfulness based on $|\theta_{y,k}(x)c_k(x)|$ and a matched CNN baseline with the same convolutional backbone to assess whether high cosine stability is SENN-specific or also appears in standard deep features. All stability and drift metrics were averaged across the test set for each perturbation level.

F. Implementation Details

All experiments were implemented in PyTorch using a unified evaluation pipeline for training, adversarial generation, and metric computation. A fixed random seed was used for reproducibility. Reported stability values correspond to test-sample means with sample-level standard deviations. We do not report multi-seed confidence intervals in this workshop version.

IV. Results and Analysis

This section evaluates predictive robustness, representation stability, decision confidence degradation, and explanation robustness under adversarial perturbations.

A. Classification Robustness

We first evaluate the predictive robustness of the model together with the stability of its latent SENN representations. Figure 3 summarizes classification accuracy, concept stability, relevance stability, and latent SENN explanation stability as functions of the perturbation magnitude ϵ.

On clean samples ($\epsilon = 0$), the model achieves an accuracy of 99.38% using the fixed experimental seed. Stability values are reported as test-sample means with sample-level standard deviations, rather than as multi-seed confidence intervals.

However, classification accuracy deteriorates rapidly as perturbation strength increases for all attacks. As shown in Fig. 3, FGSM reduces accuracy from near-perfect clean performance to approximately 10% at $\epsilon = 4.0$. The stronger iterative attacks are more damaging: both PGD and MI-FGSM reduce accuracy to approximately 3% at the same perturbation level. These results confirm that IoT firmware classifiers trained only on clean data remain highly vulnerable to both single-step and iterative white-box adversarial perturbations.

Per-class recall analysis further showed that adversarial perturbations disproportionately affect the minority hackware and malware classes, consistent with the severe class imbalance reported in the dataset.

B. Representation Stability

Despite the strong degradation in predictive performance, internal representations remain remarkably stable. Figure 3 presents cosine similarity between clean and adversarial representations for concepts (S_c), relevance weights (S_θ), and latent SENN explanations (S_e) under FGSM, PGD, and MI-FGSM. Even under $\epsilon = 4.0$, representation similarity remains high. Under FGSM, concept, relevance, and explanation stability remain approximately 0.986, 0.993, and 0.973, respectively. Under PGD, the corresponding values are approximately 0.978, 0.992, and 0.957, while MI-FGSM gives approximately 0.976, 0.992, and 0.953.

These results indicate a structural separation between representation robustness and predictive reliability: adversarial perturbations primarily affect the decision aggregation mechanism rather than corrupting the learned concept space.

To verify that concepts carry predictive signal, we performed deletion/insertion faithfulness using $|\theta_{y,k}(x)c_k(x)|$ as

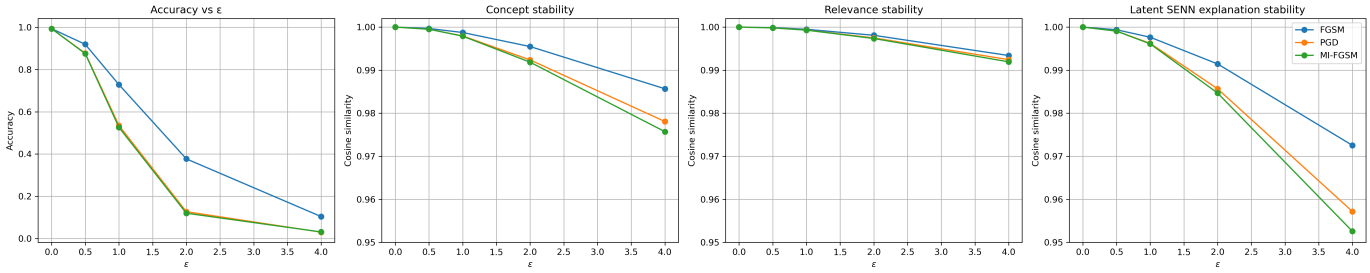


Fig. 3: Predictive robustness and latent SENN stability under FGSM, PGD, and MI-FGSM perturbations. Panel (a) shows that classification accuracy decreases sharply as ϵ increases, especially under PGD and MI-FGSM. Panels (b)–(d) show that concept stability, relevance stability, and latent SENN explanation stability remain high across perturbation levels, indicating a separation between decision vulnerability and internal representation stability.

TABLE I: Concept faithfulness and matched CNN control.

Control	Setting	Accuracy	Stability
Original SENN	clean	0.9938	–
Top-1 concept deleted	clean	0.0792	–
Bottom-1 concept deleted	clean	0.9829	–
Top-2 concepts deleted	clean	0.0229	–
Top-1 concept inserted	clean	0.9820	–
SENN	FGSM, $\epsilon = 4.0$	0.1037	0.9857
Matched CNN	FGSM, $\epsilon = 4.0$	0.6479	0.9897
SENN	PGD, $\epsilon = 4.0$	0.0308	0.9781
Matched CNN	PGD, $\epsilon = 4.0$	0.4599	0.9901
SENN	MI-FGSM, $\epsilon = 4.0$	0.0309	0.9757
Matched CNN	MI-FGSM, $\epsilon = 4.0$	0.4436	0.9896

the concept ranking score. Removing the most important concept reduced accuracy from 99.38% to 7.92%, while removing the least important concept preserved accuracy at 98.29%. Because raw byte-image patterns have limited visual interpretability, this quantitative faithfulness test serves as a compact proxy for concept usefulness.

We also trained a matched-capacity CNN baseline with the same convolutional backbone and a comparable number of parameters: 70,368 for SENN and 70,467 for CNN. The CNN retained higher adversarial accuracy, showing that high cosine stability is not exclusively a SENN property. SENN's value is therefore its explicit concept-relevance decomposition, which exposes how stable internal representations can coexist with vulnerable decisions.

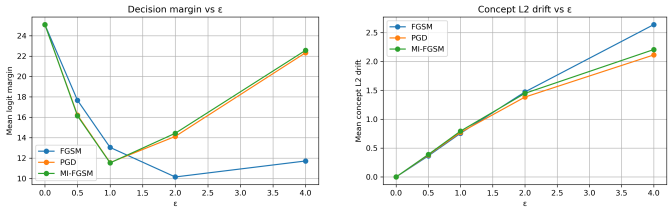
This control also prevents over-interpreting high cosine stability as a property unique to SENNs: the key contribution is not that SENN is more robust than a CNN, but that it makes the stable internal decision structure observable through concepts and relevance weights.

For SENN rows, stability denotes concept cosine similarity; for matched CNN rows, it denotes backbone feature cosine similarity.

C. Decision Vulnerability

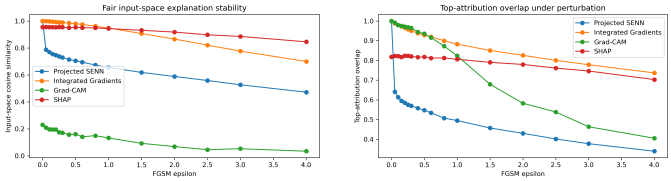
We further analyze decision confidence using logit margin and concept ℓ_2 drift.

Fig. 4 shows that concept drift increases gradually with perturbation strength, while concept directions remain highly aligned as shown in Fig. 3. The logit-margin behavior is attack-dependent: FGSM keeps margins relatively low at high ϵ,



(a) Logit margin (b) Concept ℓ_2 drift

Fig. 4: Decision confidence and concept drift under FGSM, PGD, and MI-FGSM perturbations.



(a) Cosine stability (b) Top-attribution overlap

Fig. 5: Fair input-space explanation comparison under FGSM perturbations.

whereas PGD and MI-FGSM push samples toward confident but incorrect regions. This supports the interpretation that the concept encoder remains stable while the aggregation mechanism $\theta(x)c(x)$ is decision-sensitive.

D. Explanation Robustness

We analyze explanation robustness in two stages: latent SENN explanation stability under FGSM, PGD, and MI-FGSM, and fair input-space comparison with post-hoc attribution methods under FGSM. As shown in Fig. 3, latent explanation stability remains high even at $\epsilon = 4.0$: approximately 0.973 for FGSM, 0.957 for PGD, and 0.953 for MI-FGSM.

We then compare SENN explanations with post-hoc attribution methods under FGSM using a fair input-space protocol. Rather than directly comparing the latent SENN explanation vector with input-space attribution maps, the SENN explanation is first projected back to the 32×32 byte-image space.

For class y, the projected SENN attribution map is computed as

$$A_{\mathrm{SENN}}(x, y) = \sum_{k=1}^{K} \left(\theta_{y,k}(x) c_k(x)\right) \frac{\partial c_k(x)}{\partial x}. \tag{14}$$

This maps concept contributions to the same 1024-dimensional input space used by SHAP, Integrated Gradients, and Grad-CAM. Fig. 5(a) reports cosine similarity between clean and FGSM-perturbed attribution maps, while Fig. 5(b) reports the overlap of the highest-attribution input locations as ϵ increases. SHAP and Integrated Gradients remain more stable in input space, whereas projected SENN explanations show moderate stability and Grad-CAM is consistently unstable. These results indicate that SENN does not dominate post-hoc methods under input-space attribution metrics. Its advantage is diagnostic: unlike post-hoc maps, SENN exposes the internal concept-relevance structure used for prediction, allowing concept stability, relevance stability, and decision aggregation to be analyzed separately.

E. Limitations and Practical Implications

This study has several limitations. First, the firmware representation uses only the first 1024 bytes of each ELF file, providing a compact byte-image abstraction rather than a complete executable representation. Second, the evaluated perturbations are applied in byte-image space and are not guaranteed to preserve executable ELF semantics; therefore, FGSM, PGD, and MI-FGSM should be interpreted as controlled robustness probes rather than deployable firmware attacks. Third, the evaluation is limited to one dataset, one fixed seed, and $K = 8$ concepts. Broader validation across datasets, multi-seed runs, transfer-based and decision-based attacks, firmware-aware executable-preserving attacks, and concept-number sensitivity remains future work. Finally, explanation stability may be manipulated by adaptive attackers and should therefore be treated as a diagnostic signal rather than a complete defense. In deployment, such monitoring can support firmware triage by flagging cases where predictions change while latent concepts remain stable, but it should complement standard malware analysis, executable validation, and analyst review.

V. Conclusion

This paper examined intrinsic SENN explanation stability for IoT firmware malware detection under FGSM, PGD, and MI-FGSM perturbations. We analyzed predictive accuracy, concept and relevance stability, latent explanation stability, decision confidence, and input-space attribution stability. Results show that attacks can severely degrade accuracy while the learned concept-relevance structure remains largely aligned. Concept deletion/insertion confirms that learned concepts carry predictive signal, while the matched CNN baseline shows that high feature stability is not unique to SENNs. Comparisons with SHAP, Integrated Gradients, and Grad-CAM indicate that SENN explanations provide complementary diagnostic information rather than universally stronger attribution stability. Future work will consider additional datasets, concept visualization, concept-number sensitivity, multi-seed validation, transfer/decision-based attacks, executable-preserving attacks, and other white-box explainable models.

References

[1] A. Aljuhani, A. Alamri, P. Kumar, and A. Jolfaei, "An intelligent and explainable SaaS-based intrusion detection system for resource-constrained IoMT," *IEEE Internet of Things Journal*, vol. 11, no. 15, pp. 25 454–25 463, 2023.

[2] A. Abu-Mahfouz, S. Alrabaee, M. Khasawneh, M. Gergely, and K.-K. R. Choo, "A deep learning approach to discover router firmware vulnerabilities," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 1, pp. 691–702, 2023.

[3] M. Asam, S. H. Khan, A. Akbar, S. Bibi, T. Jamal, A. Khan, U. Ghafoor, and M. R. Bhutta, "IoT malware detection architecture using a novel channel boosted and squeezed CNN," *Scientific Reports*, vol. 12, no. 1, p. 15498, 2022.

[4] E. Larsen, K. MacVittie, and J. Lilly, "A survey of machine learning algorithms for detecting malware in iot firmware," *arXiv preprint arXiv:2111.02388*, 2021.

[5] M. Wang, K. Zheng, Y. Yang, and X. Wang, "An explainable machine learning framework for intrusion detection systems," *IEEE access*, vol. 8, pp. 73 127–73 141, 2020.

[6] K. Amarasinghe, K. Kenney, and M. Manic, "Toward explainable deep neural network based anomaly detection," in *2018 11th international conference on human system interaction (HSI)*. IEEE, 2018, pp. 311–317.

[7] S. Mane and D. Rao, "Explaining network intrusion detection system using explainable AI framework," *arXiv preprint arXiv:2103.07110*, 2021.

[8] D. Gaspar, P. Silva, and C. Silva, "Explainable AI for intrusion detection systems: Lime and Shap applicability on multi-layer perceptron," *IEEE Access*, vol. 12, pp. 30 164–30 175, 2024.

[9] J. Tritscher, M. Wolf, A. Hotho, and D. Schlör, "Evaluating feature relevance XAI in network intrusion detection," in *World Conference on Explainable Artificial Intelligence*. Springer, 2023, pp. 483–497.

[10] O. Arreche, T. R. Guntur, J. W. Roberts, and M. Abdallah, "E-XAI: Evaluating black-box explainable AI frameworks for network intrusion detection," *IEEE Access*, vol. 12, pp. 23 954–23 988, 2024.

[11] M. Keshk, N. Koroniotis, N. Pham, N. Moustafa, B. Turnbull, and A. Y. Zomaya, "An explainable deep learning-enabled intrusion detection framework in IoT networks," *Information Sciences*, vol. 639, p. 119000, 2023.

[12] F. Ebrahimi, R. Javidan, R. Akbari, and Y. Hosseini, "Intrusion detection in the internet of things using convolutional neural networks: an explainable ai approach," *Cybersecurity*, vol. 8, no. 1, p. 66, 2025.

[13] A. Alabdulatif, "A novel ensemble of deep learning approach for cybersecurity intrusion detection with explainable artificial intelligence," *Applied Sciences*, vol. 15, no. 14, p. 7984, 2025.

[14] S. Ghosh, R. K. Goyal, and K. Chowdhury, "Explainable AI-Driven Intrusion Detection System for DoS Attack Classification Using Deep Learning and Optimization Techniques," *IEEE Access*, vol. 14, pp. 5618–5642, 2026.

[15] S. Neupane, J. Ables, W. Anderson, S. Mittal, S. Rahimi, I. Banicescu, and M. Seale, "Explainable intrusion detection systems (X-IDS): A survey of current methods, challenges, and opportunities," *IEEE Access*, vol. 10, pp. 112 392–112 415, 2022.

[16] S. Patil, V. Varadarajan, S. M. Mazhar, A. Sahibzada, N. Ahmed, O. Sinha, S. Kumar, K. Shaw, and K. Kotecha, "Explainable artificial intelligence for intrusion detection system," *Electronics*, vol. 11, no. 19, p. 3079, 2022.

[17] E. Poeta, G. Ciravegna, E. Pastor, T. Cerquitelli, and E. Baralis, "Concept-based explainable artificial intelligence: A survey," *ACM Computing Surveys*, 2023.

[18] D. Alvarez Melis and T. Jaakkola, "Towards robust interpretability with self-explaining neural networks," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[19] D. A. Noever, "IoT Firmware Image Classification: Rendered ELF Binaries by Class as Malware as Imagery," Kaggle dataset, 2021, accessed: 28 Apr. 2026. [Online]. Available: https://www.kaggle.com/datasets/datamunge/iot-firmware-image-classification