ISSN 1870-4069

Invertible Neural Networks for Inference Integrity Verification

Malgorzata Schwab, Ashis Biswas

[1] University of Colorado, Denver,
USA

{malgorzata.schwab,ashis.biswas}@ucdenver.edu

Abstract. In this study we explore the topic of Trustworthy AI and how reversibility in neural networks can play a role in protecting machine learning applications. We propose a framework to enhance machine learning systems robustness through the integrity verification across the inference pipeline of a deployed model and apply a concept of a Trusted Neural Network, which provides a system engineering abstraction to implement it. We leverage the Invertible Neural Network architecture with its remarkable data reconstruction and anomaly detection capabilities to validate that the inference flow pipeline is intact and thus the network prediction can be trusted, as trained. The result of that assessment is measured as an Inference Integrity Score and can be reported in real time to safeguard system integrity and suppress suspicious results. We propose an AI firewall in the form of test nodes implementing the Trusted Neural Network interface comprising an input verification layer in front of the running models, participating in the workflow. This easy to implement verification-based paradigm offers a pragmatic approach to achieve machine learning robustness and takes a step towards Trustworthy AI.

Keywords: Integrity, invertible, reversibility, robustness, trustworthiness.

1 Introduction

With the explosion of AI-augmented systems that are impacting millions around the world each day, we need to ensure that those systems are trustworthy, robust, and protected to stand up against adversarial attacks. This concept paper is inspired by the increasing role of machine learning in the decision-making process across a wide spectrum of domains, which brings to the forefront the importance of verifying the integrity of end-to-end inference flow, so the outcome of the system can be trusted.

We propose that the general solution architecture paradigm for any mission critical decision support system that leverages machine learning components incorporates a layer of integrity verification around a running model to ensure trustworthiness of the pipeline. Our technique is applicable to machine learning inference flows significant enough to be protected by an extra security layer.

We build upon the concept of a Trusted Neural Network (TNN) [1], which leverages Invertible Neural Network (INN) architecture based on a revolutionary approach to achieve reversibility in neural networks introduced by Dinh [2] and subsequently incorporated into a framework by Ardizone [3]. An INN, which is invertible by

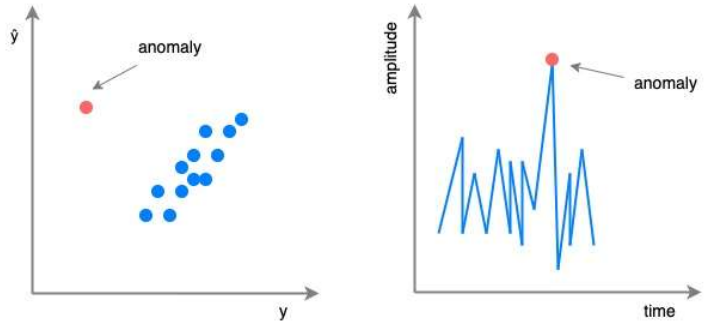


Fig. 1. Anomalies visualized [5].

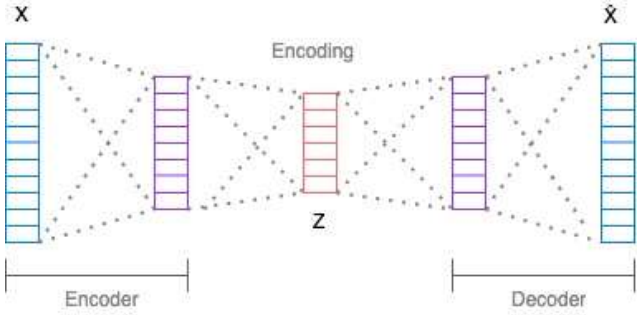


Fig. 2. Classic autoencoder [8].

construction, offers a remarkable data reconstruction capability that can be leveraged to validate that the inference flow pipeline is intact and that the output of it can be trusted.

The result of that assessment, which we call the Inference Integrity Score, can be reported in real time and acted upon to safeguard system integrity by suppressing suspicious outcomes. The implementation of our Trustworthy AI paradigm employs the TNN-based test nodes comprising an AI-firewall layer offers a pragmatic approach to protecting machine learning pipelines and does not require any intricate intervention into the models themselves to handle adversarial inputs.

The remainder of this paper is organized as follows: Section 2 briefly reviews related work pertaining to safeguarding machine learning inference pipelines. It then elaborates on anomaly detection techniques [8] and Invertible Neural Networks touching upon normalizing flows [2] - the theory underlying the reversibility of deep neural networks. We also introduce the Framework for Easily Invertible Architectures (FrEIA) previously established by Ardizone [3], which provides an SDK to construct custom INN configurations to make it quick and approachable.

We then discuss the remarkable ability of an Invertible Neural Network to reconstruct data from its compressed latent representation, outperforming traditional autoencoder architecture. In Section 3 we look at the Trusted Neural Network API and

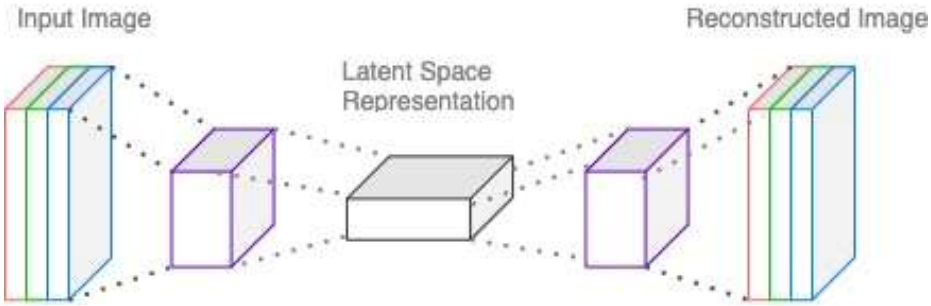


Fig. 3. Convolutional Autoencoder [8].

learn how a TNN node can be incorporated into a verification-based inference protection layer. Section 4 summaries the study and offers conclusion.

2 Related Works

Machine learning system robustness, which encompasses building reliable, resilient, and fault-tolerant machine learning systems, is an active area of research. Much attention is given to strengthen adversarial resistance of the deep learning models themselves, but a test-based verification-driven approach to validate the inference pipeline provides an effective scheme to improve system robustness, while narrowing the gap between machine learning research and practice. The risks related to the inference pipeline's state of integrity can be effectively mitigated by verifying the reasonability of the prediction outcome, which is discussed by Apruzzese in the methodology survey "Real Attackers Don't Compute Gradients" [4].

2.1 Anomaly Detection

We invoke the topic of anomaly detection as relevant to the verification of the inference pipeline integrity. Anomaly detection is a process of identifying data that does not fit into a pattern of what is expected.

As described in [5] and depicted in Figure 1, abnormal patterns in the phenomena characterized by low dimensionality can be easily discovered with an algorithmic approach based on acceptable value ranges, with simple clustering techniques, or even assessed visually.

Giannoni [5] and subsequently Yin [6] put anomaly detection methods in several categories, such as statistical-based methods, probability-based methods, similarity-based methods, and most recent prediction-based methods. The high dimensional scenarios surrounding systems with machine learning components highly dependent on integrity of the data, however, require more sophisticated multivariate statistics methods based on probability distributions and deep learning techniques.

They are exemplified by generative neural networks, such as several classes of autoencoders, including novel INN-based autoencoders described [8] based on Invertible Neural Networks trained for anomaly detection.

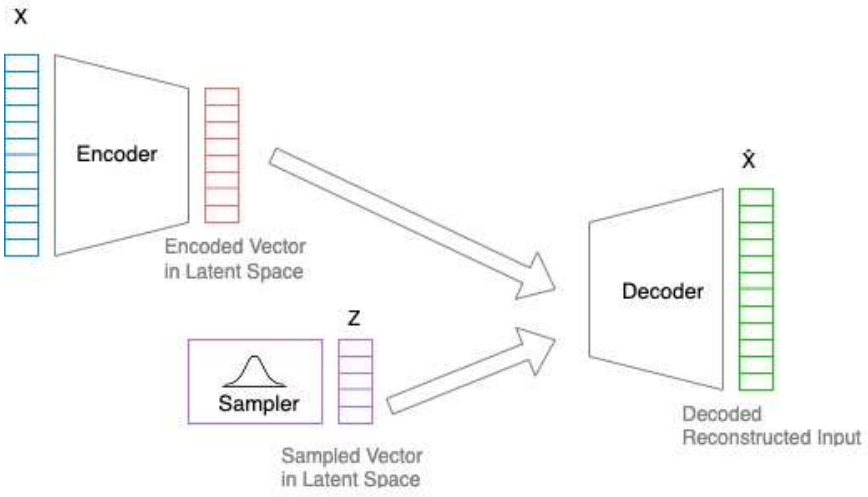


Fig. 4. Variational Autoencoder [8].

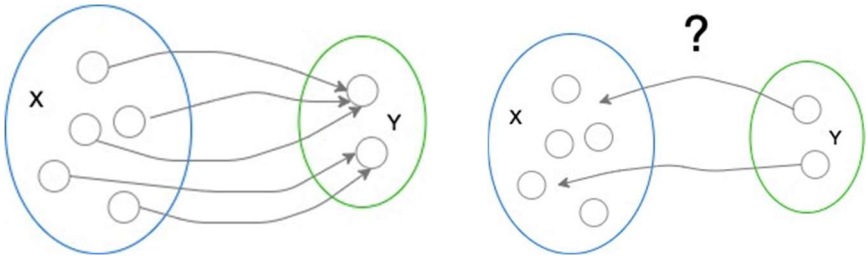


Fig. 5. Forward mapping of x → y (left) and Inverse ambiguity (right).

2.2 Autoencoders

Autoencoders belong to the family of unsupervised deep learning neural network models well suited for dimensionality reduction and have been described extensively in numerous works, such as [5] and [6], then referenced in [8]. The general idea around this type of neural network is to extract the most relevant features from input data and then learn how to reconstruct the original data from its compressed representation.

For unexpected inputs, which the model has not seen during training, the reconstruction error should be higher, and crossing a configurable threshold, dependent on a problem domain, constitutes an anomaly. As described in [8] and shown in Fig. 2, a classic autoencoder consists of an encoder and a decoder, implemented as fully connected neural networks.

The encoder compresses the network input x into a lower dimensional latent representation z defined by the bottleneck. The decoder takes the output of the encoder

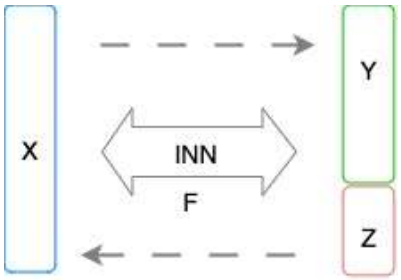


Fig. 6. Invertible Neural Network Conceptual Diagram.

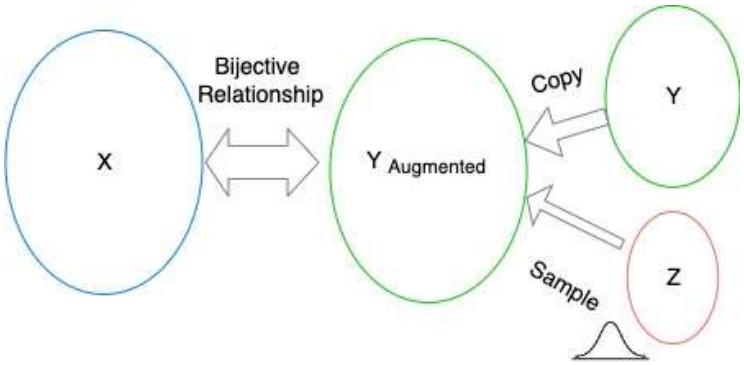


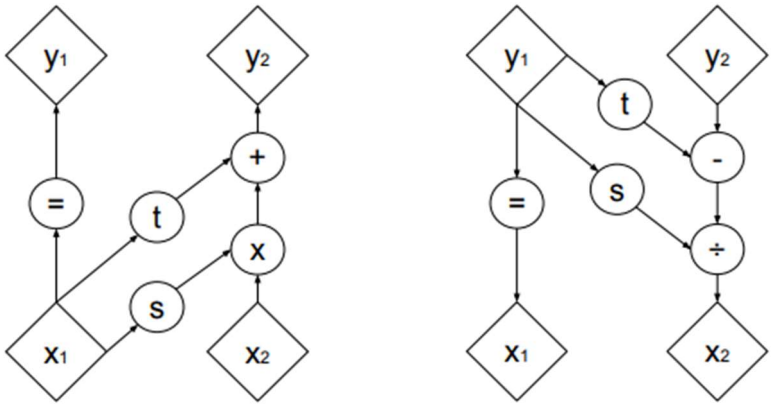
Fig. 7. Reconstructing phenomenon X from observation Y.

and decodes the latent representation back to the original input $\hat{x}$. The information preserved in hidden neurons is considered as the encoded features. The learning process is based on minimizing the reconstruction error, which is assessed by comparing the reconstructed input with the original one. The learned representation corresponds to the final hidden state of the encoder network and acts like a summary of the input sequence.

There are several variations of autoencoder architecture [8], such as a convolutional autoencoder, depicted in Fig. 3, which uses convolutional layers to create a compressed representation [6], or a variational autoencoder depicted in Fig. 4, capable not only of reconstructing the original input, but also enhancing it by generating new content based on the sampling from the learned probability density distribution of the input domain.

A compress-reconstruct type of a challenge reflected in the autoencoder encoder-decoder architecture belongs to the class of "ill-posed" inverse problems, which are characterized by inherent ambiguity due to the existence of an information bottleneck. Such problems have been successfully addressed by the reversible neural network architecture applied in Invertible Neural Networks, which makes them an interesting option to help with our integrity verification undertaking.

In this work we leverage previous findings and principles regarding several types of autoencoders together with reversible neural networks and apply the INN-based architecture for anomaly detection as a core of the TNN network integrity verification nodes.



(a) Forward propagation (b) Inverse propagation

Fig. 8. Real NVP Affine Coupling Block [2].

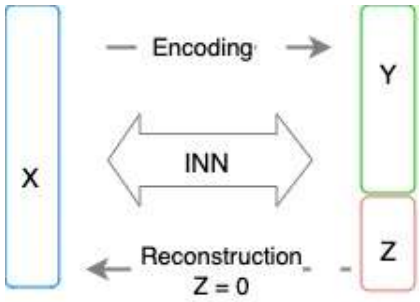


Fig. 9. INN as Autoencoder [8].

2.3 Invertible Neural Network

As explored in [8] and referenced here for context, an Invertible Neural Network is a class of networks suited to solve ambiguity that characterizes inverse problems, where multiple parameter sets can produce the same observed outcome, as depicted in Fig. 5. To express this ambiguity, the posterior probability of the parameters' distribution, given an outcome y, must be learned so the most appropriate set can be selected.

Such a model can perform log-density estimation of data points, leading to efficient inference and precise reconstruction of the inputs from the hierarchical features extracted by the model. This extraordinary capability to reconstruct the inputs corresponding to the encoder-decoder functionality makes INN a natural candidate to help solve the problem of anomaly detection.

An INN is trained simultaneously in the forward and reverse directions, Fig. 6. The forward learning process uses additional latent output variables to capture information otherwise lost, making the learning of the inverse process explicit. To solve the general inverse problem, we augment the observation space Y with a latent variable Z which follows a normal distribution and look for a bijective function F that can map Z back to $\hat{X}$.

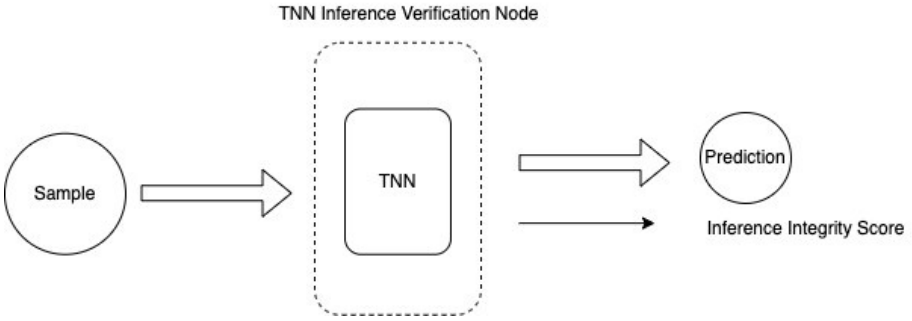


Fig. 10. TNN Context Diagram [1].

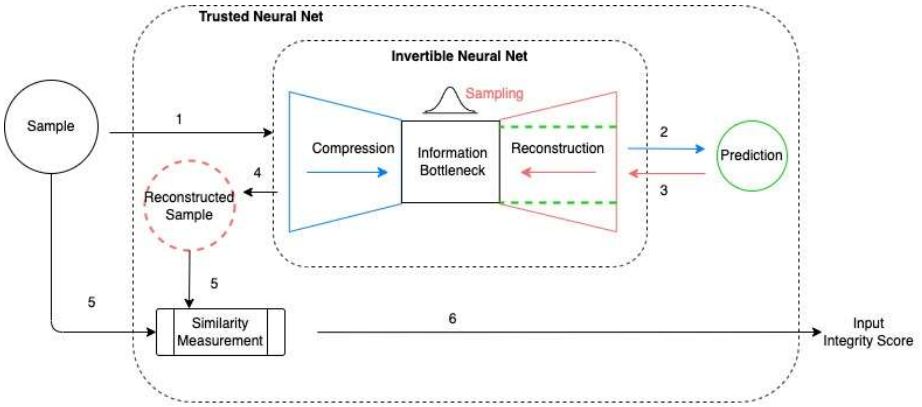


Fig. 11. TNN Architecture.

An INN learns an invertible, stable, mapping between a data distribution PX and a latent distribution PZ, typically Gaussian, as shown in Fig. 7. Invertibility of neural networks was spearheaded by Dinh [2] as "real-valued non-volume preserving transformations" (Real NVP) architecture, who introduced a stack of invertible affine coupling blocks (Fig. 8), arranged in hidden layers. Given a D-dimensional input x and d < D, the output y of an affine coupling layer follows the following equations [2]:

$$y_{1:d} = x_{1:d}, \tag{1}$$

$$y_{d+1:D} = x_{d+1:D} \odot \exp[s(x_{1:d}) + t(x_{1:d})], \tag{2}$$

where s and t are functions from $R^d \rightarrow R^{D-d}$, and $\odot$ is the Hadamard product or element-wise product. Each block splits its input and output into two parts and applies transformations s (scale) and t (translation), which themselves do not have to be invertible – they can be quite complex and are often implemented as artificial neural networks, such as a CNNs.

It has been proven [3] that a stack of such invertible blocks makes the end-to-end layout also invertible. Based on this architecture, the Invertible Neural Network guarantees reversibility by its construction and solves the ambiguous inverse relationships directly.

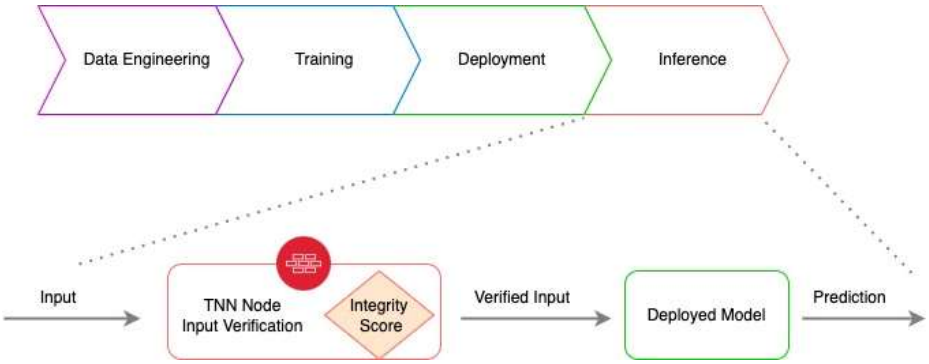


Fig. 12. A TNN node for input integrity verification.

2.4 INN Trained as an Autoencoder

As demonstrated by Nguyen [7] on MNIST, CIFAR and CelebA, and recently by Schwab [8] for time series data, an INN has superb capability for anomaly detection. It compared an INN-based implementation to conventional autoencoders for different bottleneck sizes, which demonstrated that INN autoencoders can achieve similar or better reconstruction results.

It showed that the architecture restrictions on INN autoencoders to ensure invertibility do not negatively affect their performance, while the advantages of INNs are still preserved. This entails a tractable Jacobian for both forward and inverse mapping as well as explicit computation of posterior probabilities.

It also provided an explanation for the saturation in reconstruction loss for large bottleneck sizes in classical autoencoders and concluded that an INN might not have any intrinsic information loss and thereby are not constrained by a maximal depth after which only suboptimal results can be achieved.

The concept of an INN entails bijective input-output mapping, so the dimensions of input x and output y augmented with z must be equal. As depicted in Fig. 9 below, an artificial bottleneck must be constructed to achieve autoencoder-like behavior. It is accomplished by zeroing the latent **z** to make sure that no extra information is retained by the network in the inverse process of representation learning.

As demonstrated in [8], the reconstruction loss on the anomalous samples across a variety of datasets was an order of magnitude greater as compared to the reconstruction error on the healthy validation data. The INN-autoencoder architecture also shows excellent performance, which renders it as an effective tool for the inference integrity verification task.

2.5 Trusted Neural Network

The diagram in Fig. 10 below depicts a conceptual template of a system comprising a Trusted Neural Network conceptualized in [1], where the output, in addition to the predicted result, includes an Inference Integrity Score to help assess trustworthiness of the outcome.

```
Request:

url = 'http://api.tnn.com/'
params = {'query': 'node_1'}
response = requests.get (url, params)
response.json()
```

```
Response:

Output:
{'confidence': 0.777,
 'prediction': 'compromized',
 'Inference Integrity Score': 0.987}
```

Fig. 13. TNN API Request and Response.

It leverages the capability of an Invertible Neural Network deal with inverse problems and to reconstruct an input from an output, in their respective domains.TNN is a general solution architecture paradigm and the concrete implementations reflecting the needs of specific problem domains can be derived from there. Current methodologies employed to verify the integrity of Artificial Neural Networks leverage sampling strategies, which operate in the outer perimeter of the network.

The TNN concept, however, incorporates the integrity measure as an integral part of the system. We propose that the inference flow is augmented with the inverse output-to-input verification steps, and that the INN-based Trusted Neural Network stackable nodes assume this responsibility – trained on the respective datasets, they are tasked with detecting and suppressing suspicious out-of-distribution data anomalies along the pipeline.

3 Proposed Framework for Inference Integrity Verification

3.1 Trusted Neural Network Architecture

A TNN (Fig. 11) used as the module integrity verification node is composed of several high-level building blocks, each of which is independently defined, can be independently improved, and empirically tuned to fit the needs of any individual application use case.

The integrity measure is computed by comparing an original input sample with the sample reconstructed by the Invertible Neural Network component embedded inside the TNN, and if too low, the overall prediction shall be discarded.

3.2 Information Bottleneck Principle

The INN is optimized along the principles of the Information Bottleneck Theory [10, 11] (alluded to in Fig. 11), capable of balancing the purposeful information loss against the desired accuracy of the model. The Information Bottleneck method measures how well Y can be predicted from a compressed representation Z, compared to its direct prediction from X. The algorithm minimizes the loss function L with respect to conditional distribution $p(z|x)$:

$$L_{IB} = I(X,Z) - \beta\, I\,(Y,Z), \tag{3}$$

where $I(X;Z)$ and $I(Z;Y)$ are the mutual information of X and Z, and Y and Z respectively, and β is Lagrange multiplier.

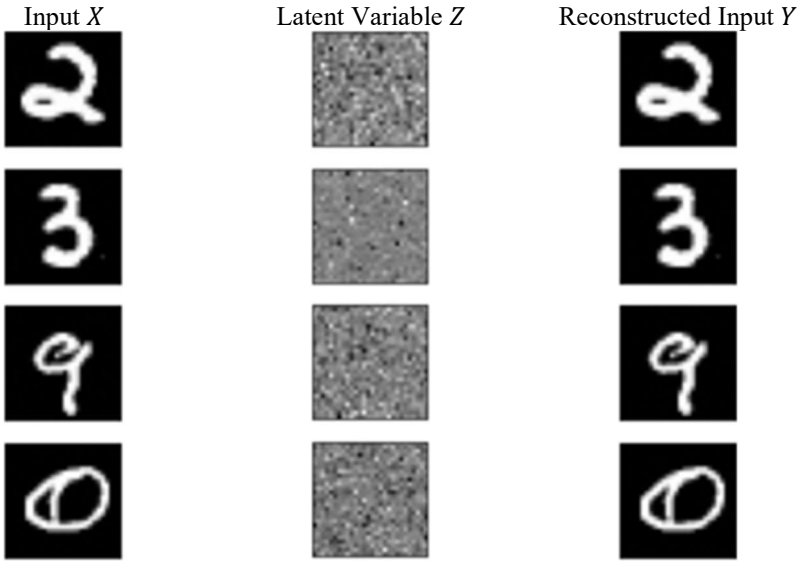


Fig. 14. MNIST Experiment.

3.3 Trustworthy AI Solution Architecture

We propose a novel type of test-driven approach to ensure ML integrity, depicted in Fig. 12, which leverages the TNN nodes to protect against adversarial data at any given step of the inference pipeline, and thus guarding its integrity. The solution employs one or more Trusted Neural Network node(s) with INN at its heart configured for data reconstruction, so that the inputs of the modules comprising a pipeline can be subjected to a test, as indicated in Fig. 12 steps 1-6.

Input and outputs of a module may or may not be in the data domain, which is the strength of Invertible Neural Networks, as compared to the classic autoencoder architecture. The similarity measure and the thresholds would vary per use case, and thus they must be designed specifically for any given domain:

$$\left\| X_{\text{inverted}} - X_{\text{original}} \right\| < \text{Reconstrucion Error Margin.} \tag{4}$$

The Trusted Neural Network design pattern comes with REST API [12], depicted in Fig. 13, which in addition to the prediction outcome also returns the Inference Integrity Score. The proposed standard would add the Integrity Score parameter to the ML API response payload as an integrated workflow security measure.

3.4 Input Reconstruction

Several experiments were conducted to verify various INN configurations with respect to reconstructing the most probable input given an output. Described in [1], they followed the implementation examples provided in [3] using synthetic points data sets.

Another experimental INN, configured to process the MNIST data set, tested successfully as well (Fig. 14). The forward pass through the invertible network gives us a latent image Z, which fed to the network in the reversed flow outputs a regenerated X, noted as X_{inverted}:

$$Z = INN_{\text{forward}}(X_{\text{original}}), \tag{5}$$

$$X_{\text{inverted}} = INN_{\text{reverse}}(Z). \tag{6}$$

The difference between the original input X entering the TNN and its counterpart X_{inverted} regenerated by the network in the reverse flow is negligible:

$$\left\| X_{\text{inverted}} - X_{\text{original}} \right\| < 1e-5. \tag{7}$$

A result like that which would be reflected in a high value of Inference Integrity Score and provide a successful test for a TNN node at a given step of the inference flow.

4 Summary and Conclusion

This work proposes an easy to implement pragmatic scheme to enhance robustness of machine learning systems through a test-driven inference flow verification layer based on the Trusted Neural Network nodes and their API abstraction. It leverages the Invertible Neural Network architecture and an open-source framework to construct the INN-based state-of-the-art anomaly detector.

The paradigm is generalizable across problem domains and aspires to become a useful practice in drafting robust high-level solution architectures for systems which incorporate machine learning capabilities and can benefit from additional measures of trustworthiness.

References

1. Schwab, M., Kumer-Biswas, A.: Trusted neural network (TNN); Reversibility in neural networks for inference integrity verification. In: Proceedings of the International Conference on Machine Learning and Applications (2023)
2. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real NVP. In: Proceedings of the International Conference on Learning Representations (2016) doi: 10.48550/ARXIV.1605.08803
3. Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., Maier-Hein, L., Rother, C., Köthe, U.: Analyzing inverse problems with invertible neural networks. In: Proceedings of the International Conference on Learning Representations (2019) doi: 10.48550/ARXIV.1808.04730
4. Apruzzese, G., Anderson, H. S., Dambra, S., Freeman, D., Pierazzi, F., Roundy, K. A.: Real attackers don't compute gradients: Bridging the gap between adversarial ML research and practice. In: IEEE Conference on Secure and Trustworthy Machine Learning (2022) doi: 10.48550/ARXIV.2212.14315
5. Giannoni, F., Mancini, M., Marinelli, F.: Anomaly detection models for IoT time series data (2018) doi: 10.48550/ARXIV.1812.00890

6. Yin, C., Zhang, S., Wang, J., Xiong, N. N.: Anomaly detection based on convolutional recurrent autoencoder for IoT time series. IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 1, pp. 112–122 (2022) doi: 10.1109/tsmc.2020.2968516
7. Nguyen, T. L., Ardizzone, L., Köthe, U.: Training invertible neural networks as autoencoders. In: German Conference on Pattern Recognition, vol. 11824, pp. 442–455 (2019) doi: 10.1007/978-3-030-33676-9_31
8. Schwab, M., Biswas, A.: Invertible neural network for time series anomaly detection. In: 8th International Conference on Software Engineering, pp. 227–239 (2023) doi: 10.5121/csit.2023.131220
9. Sutskever, I., Vinyals, O., Le, Q. V.: Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, vol. 2, pp. 3104--3112 (2014)
10. Tishby, N., Zaslavsky, N.: Deep learning and the information bottleneck principle. In: IEEE Information Theory Workshop (2015) doi: 10.48550/ARXIV.1503.0240
11. Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., Cox, D. D.: On the information bottleneck theory of deep learning. In: International Conference on Learning Representations (2018)
12. Fielding, T.: Architectural styles and the design of network-based software architectures. Dissertation submitted in partial satisfaction of the requirements for the degree of doctor of philosophy, University of California (2000) www.ics.uci.edu/~fielding/pubs/dissertation/fieldi ng_dissertation.pdf