Considering the Impact of Noise on Machine Learning Accuracy

Mahum Naseer, Iram Tariq Bhatti, Osman Hasan, and Muhammad Shafique

1 Introduction

Due to their astounding classification performance and decision-making capability in practical applications such as healthcare, smart cyber-physical systems (CPS), autonomous driving, and Internet of Things (IoTs) [7, 18, 26], there has been a continuous rise in the use of embedded machine learning (ML)-based systems in the past few decades. A major contributing factor to the success of these ML-based systems is the advancements in the underlying artificial neural networks (ANNs). However, the addition of even small noise to the input of ANNs may lead these sophisticated systems to provide erroneous results [28]. This impact of noise is easy to visualize in Fig. 1, where the addition of noise to the input images does not lead to any perceptible change in the input. Nevertheless, the small noise is sufficient to make a trained ANN classify the inputs incorrectly.

Noise is a ubiquitous component of the physical environment. Whether it be due to atmospheric conditions such as fog and pollution, or perturbation at input sensors

M. Naseer (✉)
Technische Universität Wien (TU Wien), Vienna, Austria
e-mail: mahum.naseer@tuwien.ac.at

I. T. Bhatti
SAVe Lab, School of Electrical Engineering & Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan
e-mail: iram.tariq@seecs.edu.pk

O. Hasan
National University of Sciences and Technology (NUST), Islamabad, Pakistan
e-mail: osman.hasan@seecs.nust.edu.pk

M. Shafique
Division of Engineering, New York University Abu Dhabi (NYUAD), Abu Dhabi, UAE
e-mail: muhammad.shafique@nyu.edu

S. Pasricha, M. Shafique (eds.), *Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing*, https://doi.org/10.1007/978-3-031-40677-5_15

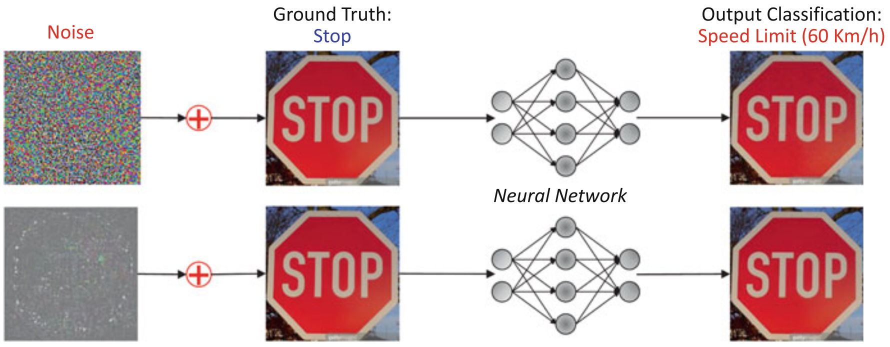


Fig. 1 Impact of noise on the accuracy of machine learning systems: the addition of small noise to input may result in a output misclassification [14]

during data acquisition, it is unlikely to have a system deployed in the real world that is completely immune to noise [27]. Even though the magnitude of noise is often considerably small compared to the magnitude of the input, as shown by the orange and blue bars, respectively, in Fig. 2, it is capable of making the ANN delineate unexpected behavior.

This is a serious concern for ML-based system, particularly for the ones deployed in safety-critical applications. Hence, in order to obtain a robust system, the effects of noise need to be studied and accounted for prior to its deployment in real world. This chapter discusses the possible impact(s) of noise on trained ANNs and explores the techniques to identify the ANN vulnerabilities resulting from noise. The rest of this chapter is organized as follows: Sect. 2 highlights the available approaches from the literature targeted at studying the impact of noise in ANNs, including current limitations in the study of the impact of noise on ANNs. Section 3 elaborates on the various ways in which ANNs are known to be affected by noise. Section 4 describes the different noise models used for modeling noise to study their impact on trained ANNs. Section 5 uses the knowledge of noise analysis, effects, and modeling to experimentally demonstrate the impacts of noise on an actual ANN. Section 6 concludes the chapter while emphasizing upon the key lessons learned in the chapter.

2 Studying the Impact of Noise: A Brief Overview of the Existing Literature

The study of the effects of noise on ANNs has been an active research domain for the past decade. The approaches generally used for the exploration of the impacts of noise range from ANN gradient exploitation techniques to classical formal methods. However, as shown in Fig. 3, these approaches can be broadly categorized under *noise generation* and *formal analysis* techniques. This section provides an overview

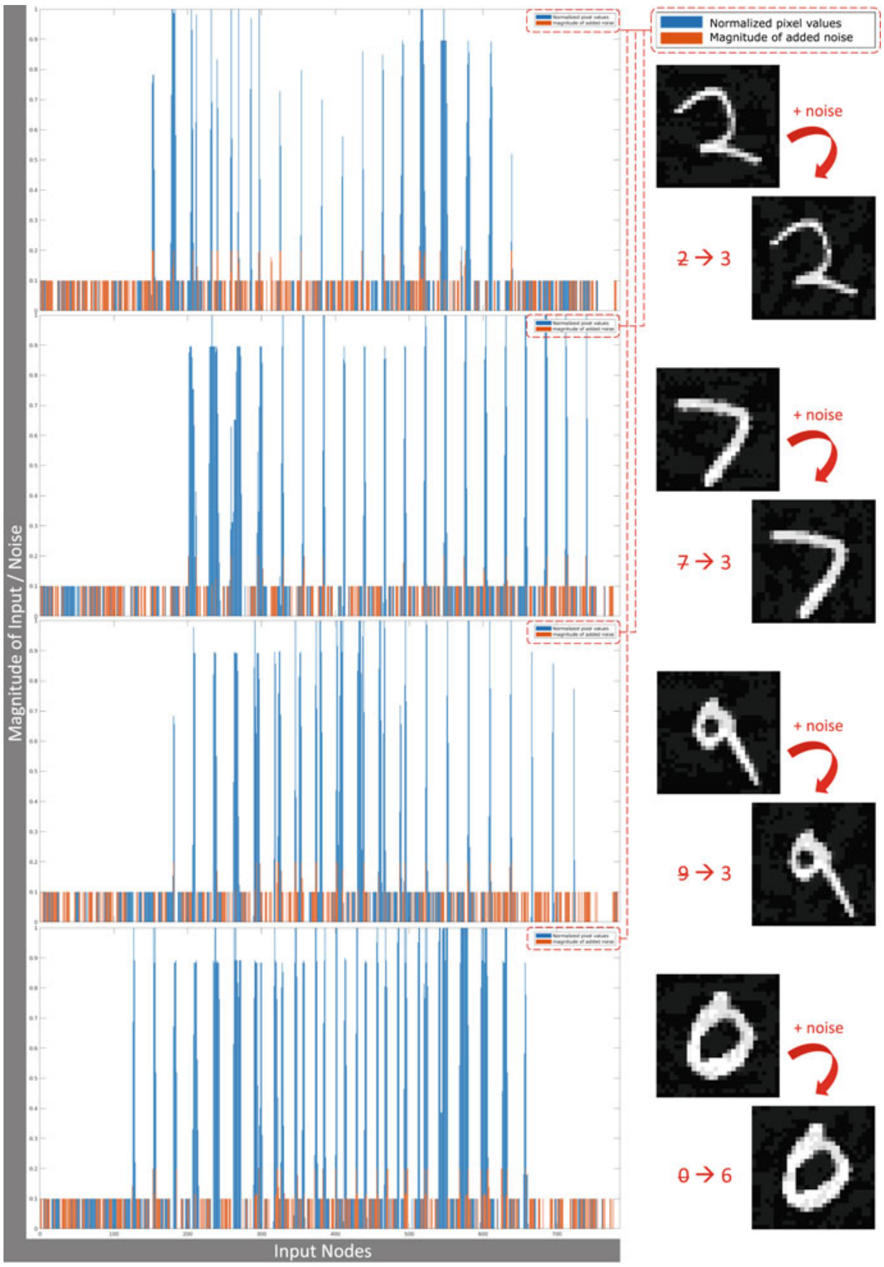


Fig. 2 The magnitude of noise (shown in orange) is often small in comparison to the input magnitude (shown in blue). Hence, the resulting change in input is too minute to be perceptible, while still making the ANN misclassify the (noisy) input

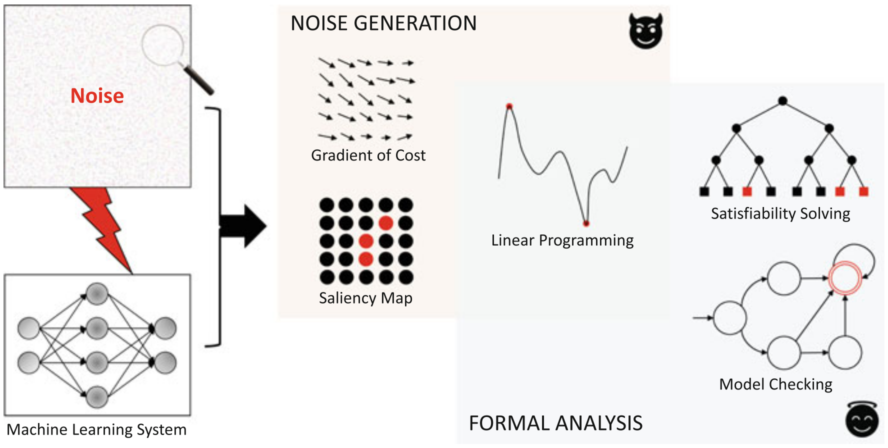


Fig. 3 Categorization of the approaches used for studying the impact of noise on ANNs

of the techniques used for studying the impact of noise, highlighting their underlying assumptions and working principles.

2.1 Noise Generation

The noise generation techniques are generally studied under adversarial attacks literature [15], where an attacker makes use of gradients of trained ANNs, true classification labels, and/or output probability vectors to generate the noise. The underlying assumption of these techniques is that a small noise exists, which when added to the ANN input will cause the ANN to generate an incorrect output. The techniques are formulated as optimization problems with either or both of the following objectives:

1. **Objective** 1: Maximize the probability of the network f classifying the seed input x to an incorrect output class $L(y)$, where $L(x) \neq L(y)$, in the presence of the noise n.

$$max(f(x+n) = L(y)).$$

2. **Objective** 2 : Find the minimal noise n (alternatively, minimize the noise) [3, 9], such that the application of noise to the seed input x of the network f provides an incorrect output classification $L(y)$, i.e., $f(x+n) = L(y)$.

The optimization problem also often contains the *imperceptibility* constraint, i.e., the generated (adversarial) noise must have a smaller magnitude compared to that of the input, hence going unnoticed. This may be achieved by the small iterative increment of the generated noise until an adversarial noise is obtained [13, 17, 19], the addition of noise only to a subset of input nodes [24, 32], or ensuring that the noisy input follows the correlation and structural similarity of the clean input [14].

2.2 Formal Analysis

Formal analysis for studying the impact of noise on ANNs involves the use of either linear programming or classical formal method approaches such as satisfiability (SAT) solving and model checking [27]. Unlike noise generation where adversarial noise is always assumed to exist, here the noise is conjectured to be absent. The task of formal analysis is then to either prove the conjecture or find a counterexample to give evidence of the existence of misclassifying noise.

2.2.1 Linear Programming

Similar to noise generation, linear programming for formal ANN analysis also makes use of optimization. The behavior and architecture of the ANN are expressed as a set of linear constraints. The piece-wise linear activation functions (such as ReLU) may be formalized as linear constraints using techniques such as Big-M approach [1, 6]. However, not all ANN activation functions are piece-wise linear. Hence, they may require approximation techniques to transform the non-linear activations to piece-wise linear functions [2, 20, 29–31]. The effect of noise to be studied is also expressed as a linear constraint. The objective is then either to minimize the input noise while ensuring all constraints are met or to maximize the output bounds for noisy inputs while ensuring the ANN does not delineate faulty behavior.

2.2.2 Satisfiability Solving

SAT solving is a classical formal method approach, where the ANN along with the negation of its desired network property is expressed in the conjunctive normal form (CNF). In the case of studying the impact of noise, the input to the ANN is assumed to be noisy. An automated SAT or SAT modulo theory (SMT) solver then searches for a satisfiable solution to the CNF. The existence of a satisfying solution (a.k.a. counterexample) signifies that the desired property does not hold for the network with noisy input. On the contrary, an unsatisfiable (UNSAT) solution proves that noise has no adverse impact on the desired network property [4, 10–12, 22, 25].

2.2.3 Model Checking

A relatively less explored formal method approach for studying the impact of noise on ANNs is model checking [23]. Here, the ANN, with the noisy input(s), is expressed as a state-transition system. The desired network property is expressed in temporal logic. The task of the model checker is to find a path reachable to a state satisfying the temporal property. If no reachable state satisfies the desired property, the model returns UNSAT/property holds. In the case of a probabilistic model checker, the tool may also be used to obtain the probability of the desired property being satisfied by the network.

2.2.4 Limitations in the Existing Literature

Despite the significant efforts in the domain of impacts of noise on ML-based systems, particularly ANNs, the existing literature has two major limitations:

1. Noise has numerous impacts on the classification, performance, and accuracy of ML-based systems. However, the existing works focus often on only a limited set of ANN properties. This will be discussed further in Sect. 3.
2. Most works explore the impact of noise on ANNs by adding the noise to normalized inputs. However, in practical scenarios, the noise perturbs raw (unnormalized) inputs. This limitation will be elaborated in Sect. 4.

3 Effects of Noise on Machine Learning Accuracy

As highlighted earlier, noise impacts the classification accuracy of ANNs in numerous ways. This section describes and formalizes important noise-dependent ANN properties.

3.1 Decreasing Robustness

Robustness defines the ability of a network to generate correct output classification, despite the presence of noise. It can be further categorized into *global* and *local* robustness.

Definition 1 (Global Robustness) Given a network f and a correctly classified input x with output class $L(x)$ from input domain X, the network is said to be robust against noise n iff the output classification $f(x)$ does not change under the influence of noise, i.e., $\forall x \in X : \forall n \leq N \implies f(x+n) = f(x) = L(x)$.

Definition 2 (Local Robustness) Given a network f and an arbitrary correctly classified input x with output class $L(x)$ from input domain X, the network is said to be robust against noise n iff the output classification $f(x)$ does not change under the influence of noise, i.e., $\exists x \in X : \forall n \leq N \implies f(x+n) = f(x) = L(x)$.

Global robustness requires checking the robustness of the entire input domain. Given the large input domains in real world, with often infinite instances of inputs, checking the global robustness therefore becomes infeasible. Hence, the local robustness of the network is instead checked, i.e., the robustness of ANNs around finite seed inputs. As explored in both noise generation and formal-analysis-based techniques (check references in Sect. 2), the robustness of ANNs is often found inversely correlated to the magnitude of incident noise.

3.2 Noise Tolerance

A stronger notion compared to robustness is noise tolerance. As the name suggests, noise tolerance defines the maximum noise under which the ANN stays robust. Hence, if the network is tolerant to noise N_{max}, it is robust against all noise less than N_{max}:

$$Noise\ tolerance \implies Robustness.$$

Similar to robustness, the noise tolerance can also be further categorized into *global* and *local.*

Definition 3 (Global Noise Tolerance) Given a network f and a correctly classified input x with output class $L(x)$ from input domain X, the network is said to be robust against all noise $n \leq N_{max}$, where N_{max} is the global noise tolerance of the ANN, iff the output classification $f(x)$ does not change under the influence of n, i.e., $\forall x \in X : \forall n \leq N_{max} \implies f(x+n) = f(x) = L(x)$.

Definition 4 (Local Noise Tolerance) Given a network f and an arbitrary correctly classified input x with output class $L(x)$ from input domain X, the network is said to be robust against noise $n \leq n_{max}$, where n_{max} is the local noise tolerance of the ANN for a finite number of input seeds, iff the output classification $f(x)$ does not change under the influence of n, i.e., $\exists x \in X : \forall n \leq n_{max} \implies f(x+n) = f(x) = L(x)$.

Again, given the often infinite scope of input domain for real-world systems, local noise tolerance is checked in practice rather than the global noise tolerance. Since the noise tolerance of a trained ANN is a constant entity, a change in incident noise does not vary it. However, a higher noise tolerance signifies that the ANN provides accurate results even in fairly noisy input settings.

3.3 Aggravating Bias

ANN suffers from numerous biases. Among the most explored include data bias (i.e., the bias resulting from the lack of generalization of the training dataset for the entire input domain) and representation bias (i.e., the bias resulting from acquiring faulty/imprecise training data). Noise, however, is found to aggravate the training (a.k.a. robustness) bias [21, 23], henceforth referred to as simply the *bias*.

Definition 5 (Bias) Let f be a neural network with correctly classified inputs x_A and x_B belonging to input domains X_A and X_B, and having true output classes $L(x_A)$ and $L(x_B)$, respectively. The network is said to be biased toward class $L(x_A)$ if application of noise $n \leq N$ to x_A does not change the output class $f(x_A + n)$, but the application of same noise to input x_B changes its output classification $f(x_B + n)$. In other words, noise n is more likely to change output classification of inputs belonging to input domain X_A than vice versa, i.e., $\forall x_A \in X_A, \forall x_B \in X_B : \forall n \leq N \implies P[f(x_A + n) = L(x_A)] \gg P[f(x_B + n) = L(x_B)]$.

As elaborated in the literature [21], the reason for such bias is the smaller distance between the decision boundaries obtained via training inputs from certain class(es). Hence, this bias aggravates in the presence of noise.

3.4 Varying Sensitivity Across Input Nodes

ML-based systems deploy ANNs that generally comprise of multiple input nodes. The sensitivity of these nodes, in the presence of noise, may vary.

Definition 6 (Input Node Sensitivity) Given a network f with k input nodes. Let x be a correctly classified input from the input domain X with true output classification $L(x)$. The input node i of the network is said to be sensitive to the noise η if the addition of η to x^i triggers an incorrect output classification with large probability, i.e., $\forall x \in X, \exists x^i \subseteq x : \eta \leq N \implies P[f(x \backslash x^i, x^i + \eta) \neq L(x)] > C$, where $C \in \mathbb{R}$ is a large number less than 1.

This is an important impact of noise exploited in the noise generation literature [24, 32] while exploring the ANN's input saliency maps to identify input nodes, the addition of noise to which is more likely to trigger an incorrect network output. The concept has also been studied in a recent model-checking-based formal analysis [23], to identify the type of noise the input node(s) of a trained ANN might be vulnerable to.

4 Modeling Noise

As elaborated in the previous section, noise impacts the accuracy and classification of ML-based systems, using ANN as a component, in numerous ways. Hence, the study of these impacts on trained ANNs is essential prior to their deployment in real-world applications. Whether it be the noise generation works or the formal analysis efforts, a crucial part of studying the impact of noise on ANNs is (realistic) noise modeling. This section explores the most popular noise models used in the literature, along with their strengths and weaknesses.

4.1 L^p Norms

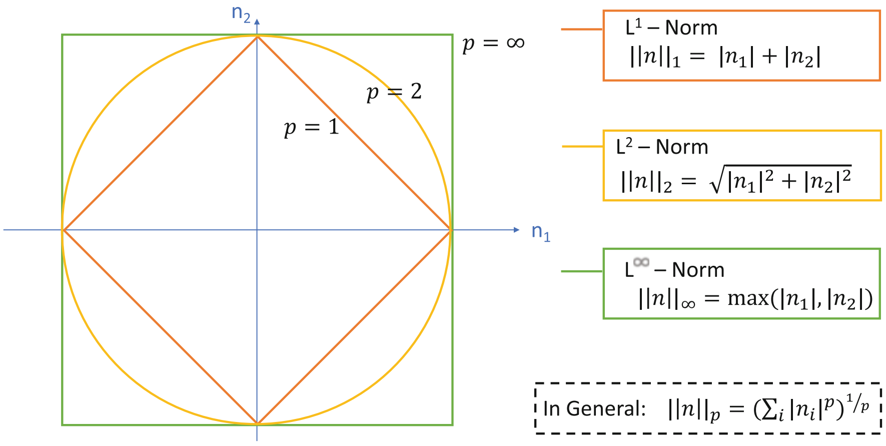
These are the most popular noise models used in the ANN literature. In the context of incidence of noise to the ANN inputs, L^p norms define the magnitude of distance between true and noisy inputs. Mathematically, they determine the p^{th} root of the sum of p^{th} power of absolute distance between the true and noisy inputs:

$$||n||_p = \sqrt[p]{\sum_i |n_i|^p} = \sqrt[p]{\sum_i |x'_i - x_i|^p},$$

where x_i and x'_i denote the ith nodes of true and noisy inputs, respectively. Fig. 4 summarizes the most common L^p norms in the literature, described in detail as follows.

4.1.1 L^1 Norm (Manhattan Distance)

This provides the sum of absolute distances between nodes of true and perturbed inputs (x_i and x'_i, respectively):



n_2
n_1
$p = \infty$
$p = 2$
$p = 1$
L^1 – Norm
$||n||_1 = |n_1| + |n_2|$
L^2 – Norm
$||n||_2 = \sqrt{|n_1|^2 + |n_2|^2}$
L^∞ – Norm
$||n||_\infty = \max(|n_1|, |n_2|)$
In General: $||n||_p = (\sum_i |n_i|^p)^{1/p}$

Fig. 4 The noise bounded by different L^p norms is applied to the neural network input nodes (i.e., $x'_i = x_i + n_i$) during analysis

$$||n||_1 = \sum_i |n_i| = \sum_i |x'_i - x_i|. \tag{1}$$

L^1 norm bounded noise model is fairly straightforward to implement.

4.1.2 L^2 Norm (Euclidean Distance)

This provides a more sophisticated measure of distance between the true and perturbed inputs. Mathematically, it is the square root of the sum of squared distances between nodes of true (x_i) and perturbed (x'_i) inputs:

$$||n||_2 = \sqrt{\sum_i |n_i|^2} = \sqrt{\sum_i |x'_i - x_i|^2}. \tag{2}$$

Compared to L^1 norm, L^2 norm provides a less robust measure of distance between the inputs. This means that even a small magnitude of distance is magnified in L^2 norm due to the squared power.

4.1.3 L^∞ Norm

This gives the maximum magnitude of distance between true and perturbed inputs (i.e., x_i and x'_i):

$$||n||_\infty = max_i(n_i) = max_i(|x'_i - x_i|). \tag{3}$$

As shown in Fig. 4, L^∞ norm encapsulates all other L^p norms. This means that the noise explored under L^p norm, for $p < \infty$, is also explored for L^∞ of the same magnitude. Hence, it is the most widely used noise model used in the literature.

4.2 Relative Noise

In practice, noise bounded by the L^p norm is added to the normalized input. However, in reality, the *noise affects the raw, unnormalized inputs*. The direct application of L^p norm bounded noise to the raw data may not always be a workable solution, particularly for cases where the range of possible input values varies across the different ANN input nodes. As a solution to these problems, recent work [23] proposes the use of the relative noise model. Here, the noise is added to the raw input as a percentage of the actual magnitude of the input. Mathematically:

$$n_i = 0.01 \times \epsilon \times x_i, \tag{4}$$

where n_i refers to the noise applied to the i^{th} input node, i.e., x_i, while ϵ is the percentage of input that contributes to the noisy input.

5 Case Study

To show how noise affects an actual network, this section describes the ANN analysis framework, FANNet. It is then used to analyze the various aforementioned ANN properties impacted by noise on a fully connected neural network trained on real-world dataset. The section later provides and elaborates on the results obtained from the analysis.

5.1 FANNet: Formal Analysis of Neural Networks

The first step for the ANN analysis is the *architectural and behavioral extraction* of a trained network. This implies that the details including the number of ANN layers and neurons in each layer, types of activation functions used at each network layer, and the values of trained parameters (i.e., weights and biases) are determined. The details are used to write the formal ANN model. The preferred choice for formal modeling in this section is model checking, which develops the formal model as a state-transition system. However, any other formal verification tool is also applicable.

The results from the formal model are then checked against labeled inputs to *validate* the correctness of formal modeling. This is to ensure that the formal model fully and correctly encapsulates the behavior of the actual trained ANN. The impact of noise on the accuracy and performance of trained ANN is then analyzed. This involves the application of noise to labeled seed inputs and supplying the noisy inputs to the verified formal ANN model. The desired ANN property (as described in Sect. 3) is then verified using the model checker.

In case the property holds for the ANN, the model checker returns `UNSAT`. In case the property does not hold in the presence of noise, depending on the choice of model checker used, the framework provides either a counterexample (i.e., the evidence of the noise that triggers faulty ANN behavior) or the probability of the ANN delineating faulty behavior for the given input noise. For determining the noise tolerance of the network, the framework takes an iterative approach. Starting with large noise, the noise applied to the ANN inputs is iteratively reduced, while verifying the ANN property at each iteration. Hence, the maximum noise at which the ANN does not delineate misclassification determines the noise tolerance of the ANN. Figure 5 pictorially summarizes the described framework.

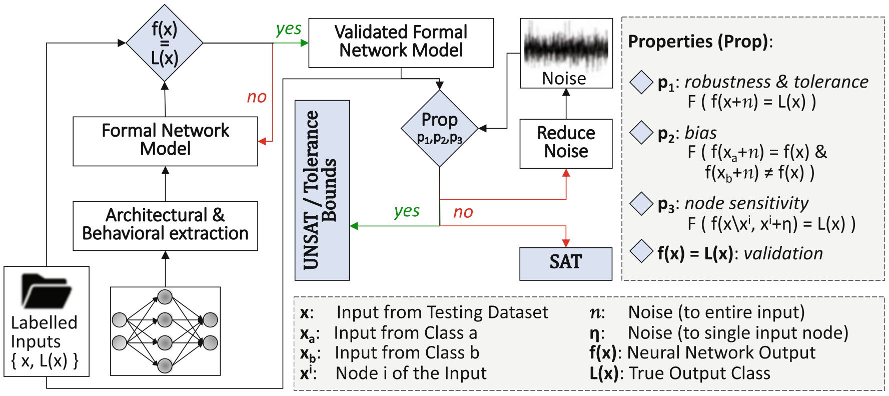

f(x) = L(x)
yes
no
Validated Formal Network Model
Noise
Reduce Noise
Prop p1,p2,p3
Formal Network Model
Architectural & Behavioral extraction
UNSAT / Tolerance Bounds
yes
no
SAT
Labelled Inputs { x, L(x) }
Properties (Prop):
p1: robustness & tolerance F (f(x+n) = L(x))
p2: bias F (f(xa+n) = f(x) & f(xb+n) ≠ f(x))
p3: node sensitivity F (f(x\xi, xi+η) = L(x))
f(x) = L(x): validation
x: Input from Testing Dataset
xa: Input from Class a
xb: Input from Class b
xi: Node i of the Input
n: Noise (to entire input)
η: Noise (to single input node)
f(x): Neural Network Output
L(x): True Output Class

Fig. 5 The framework takes in trained ANN and labelled seed inputs and analyzes the desired ANN properties impacted by noise to the ANN inputs

5.2 *Experimental Setup*

We implemented the framework using the acclaimed and open-source model checker, Storm [5]. For the experiments using the relative noise model, the precision of noise was chosen to be 1%, while for the experiments using the L^p norm model, the precision was 0.01. All experiments were run on AMD Ryzen Threadripper 2990WX processors running Ubuntu 18.04 LTS operating system. The following describes the dataset and network architecture used to show the impact of noise on a trained ANN.

5.2.1 Dataset

We use the Leukemia dataset with top 5 attributes extracted using Minimum Redundancy and Maximum Relevance (mRMR) feature selection [8, 16]. The dataset lists the readings from the genetic attributes of Leukemia patients. The output corresponds to two types of Leukemia: Acute Myeloid Leukemia (AML) and Acute Lymphoblast Leukemia (ALL). Approximately, 70% of the inputs from the training dataset belong to ALL patients, whereas roughly 60% of inputs from the testing dataset belong to ALL patients. Hence, the dataset contains significantly more inputs from ALL patients compared to AML patients, making the ANN trained quite likely to delineate a bias.

5.2.2 Neural Network

We train a fully connected neural network, as shown in Fig. 6a, using the Leukemia dataset. The network comprises a single hidden layer with 20 neurons and uses the

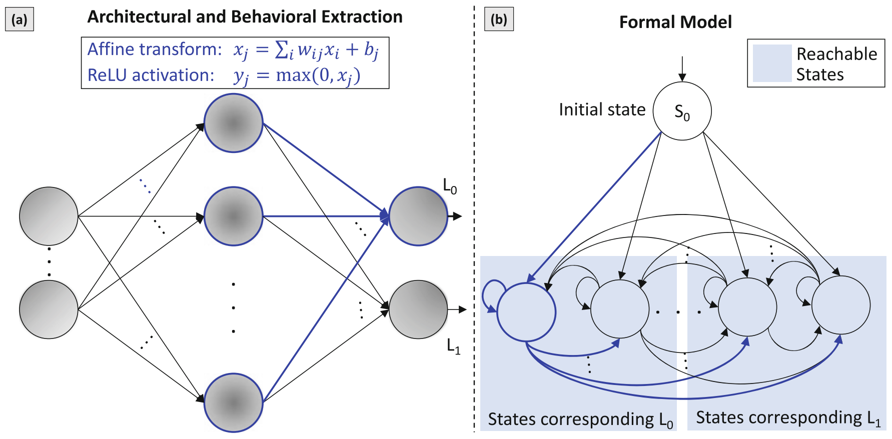

(a)
Architectural and Behavioral Extraction
Affine transform: $x_j = \sum_i w_{ij} x_i + b_j$
ReLU activation: $y_j = \max(0, x_j)$
L0
L1
(b)
Formal Model
Reachable States
Initial state
S0
States corresponding L0
States corresponding L1

Fig. 6 (**a**) The architecture of ANN trained on the Leukemia dataset. (**b**) The state-transition system of the formal ANN model: L_0 and L_1 correspond to the outputs AML and ALL, respectively, while the number of states generated corresponding to each output depends on the noise applied to the model

ReLU activation function. We train the network using 80 epoch, with learning rates of 0.5 and 0.2 for the initial and final half of the epochs, respectively. The network is trained to a training accuracy of 100% and a testing accuracy of 94.12%.

We use the analysis from the original work on FANNet based on nuXmv [23] to identify the most vulnerable inputs in the testing dataset for each label, for a trained ANN with identical parameters as those in the prior work, to perform elaborate Storm model-checker-based experiments.

5.3 *Results and Discussion*

From prior work [23], it was observed qualitatively that the increase in noise reduces the classification accuracy of the ANN while aggravating the bias. This is summarized in Fig. 7. This section provides the quantitative results obtained via aforedescribed experiments and discusses the impact of noise on trained ANN based on the empirical findings.

5.3.1 Robustness and Tolerance

As expected, the probability of correct classification reduces with the increase in the magnitude of noise, as shown in Fig. 8. For all noise less than the noise tolerance of the network (also shown in Fig. 7), the ANN provides correct output classification with a probability of 1.0, even in the presence of noise in the input. For the given

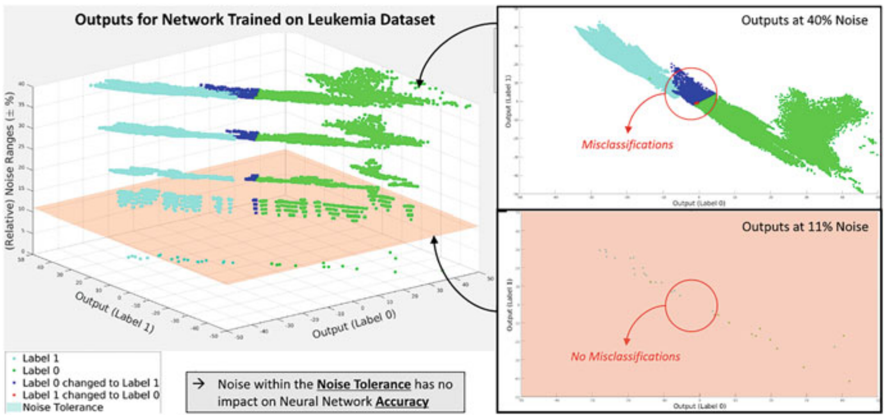


Fig. 7 Impact of increasing (relative) noise on the output classification of the trained network, as observed using nuXmv-based FANNet implementation [23]

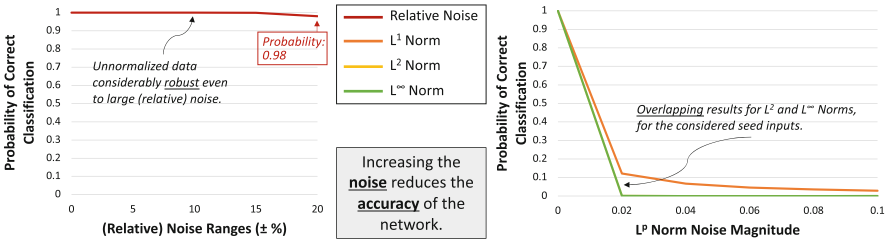


Fig. 8 Increasing noise vs. the probability of correct output classification: the decreasing robustness of ANN beyond noise tolerance is observable in the case of the relative noise model (i.e., the graph on the left)

network, this tolerance is found to be 11% in the case of the relative noise. For the network under L^p norm-based noise, the robustness was significantly low, with the noise tolerance less than the precision of the analysis.

Nevertheless, the decreasing robustness of trained ANN model under the impact of increasing noise is evident for all noise models, as shown in Fig. 8.

5.3.2 Bias

As indicated earlier, the ANN is trained on a dataset with a significantly larger proportion of inputs from patients having ALL (henceforth referred to as Label 1), as compared to those having AML (henceforth referred to as Label 0). This is likely to result in a biased ANN, as observed with the relative noise model (Fig. 9—left). For inputs classified correctly in the absence of noise, i.e., inputs having a correct classification probability of 1.0, the input noise has a more adverse impact on the inputs belonging to Label 0, as compared to vice versa. Observing the qualitative

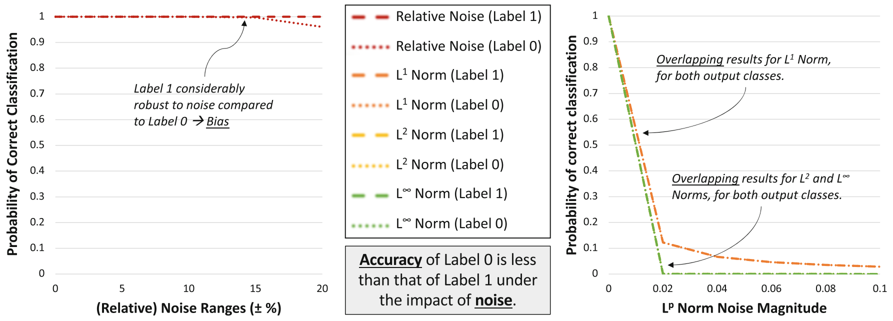


Fig. 9 The bias is visible through the analysis under relative noise model, where probability of correct classification reduces only for single output class. However, the impact is not observable with L^p norm noise model

analysis [23] from Fig. 7 supports the same conclusion. However, the bias is not observable under L^p norm-based noise model, likely due to the low robustness of the ANN under that model.

We believe that, owing to the larger proportion of inputs from Label 1 in the training dataset, the decision boundary learned by the ANN better encapsulates the inputs from Label 1. The inputs from Label 0, on other hand, are presumably closer to the decision boundary and hence more likely to be misclassified under the application of noise.

5.3.3 Node Sensitivity

As discussed in Sect. 3, different input nodes of a trained ANN may have a different sensitivity to the applied noise. Again, this impact of noise is observable only with the relative noise model, for the ANN trained on the Leukemia dataset, as shown in Fig. 10. It can also be observed that certain input nodes may be more sensitive to either positive (for instance, node x_3) or negative (for instance, node x_5) noise.

5.3.4 Discussion

As highlighted earlier in Sect. 4, unlike the relative noise, the L^p norm noise is added to the normalized inputs, i.e., the inputs in the range [0,1]. For the analyzed network, the raw, unnormalized input values range on the scale of hundreds to thousands. Assuming an input node value to be 10,000, the addition of 0.01 units of noise to the input implies the addition of a noise of magnitude 100. Such a large noise may or may not be very realistic for the noise analysis for an ANN to be deployed in a practical setting. This could be a possible reason for the inadequacy of the aforementioned noise model for analyzing the impacts of noise beyond robustness, for the given ANN.

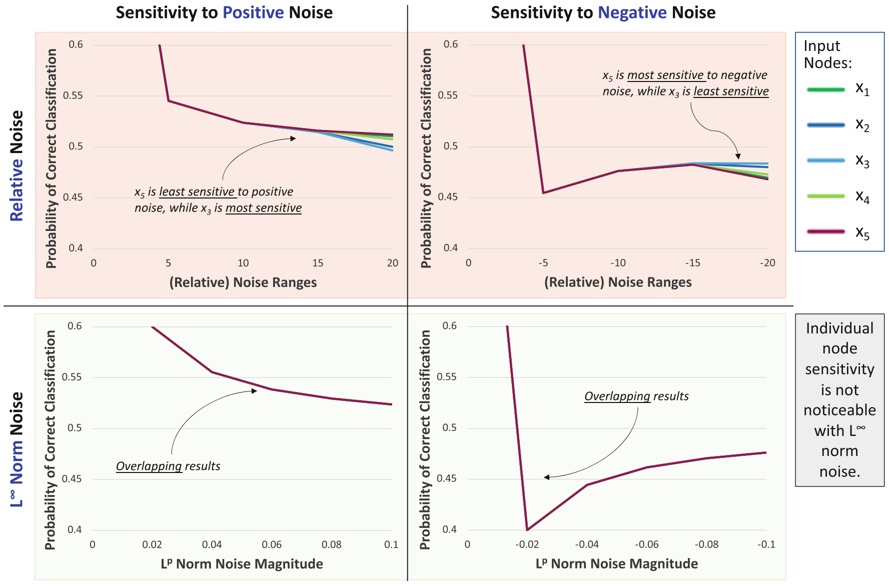


Fig. 10 The sensitivity of individual input nodes, to positive and negative noise, as observed under the relative and L^{∞} norm-based noise models

At the same time, it is possible to have another node with an input value of 100. Here, the application of the same noise (i.e., 0.01) implies a change of only a unit difference in the magnitude of the input node. This is a very likely change in the input of ANN deployed in real world. Hence, the noise 0.01 may result in realistic noise for some input nodes, while unrealistic for others, making the noise model inept for ANNs with inputs having different input ranges.

6 Conclusion

Despite the highly accurate decision-making of the current machine learning (ML)-based system, often due to the high accuracy of their underlying artificial neural networks (ANNs), these systems may fail to provide the expected accuracy in the real-world applications. A major reason for this is the noise in the practical environment, which alters the system input. Though alteration to input by noise may be fairly minimal in comparison to the magnitude of the actual input, the noise may still be able to make the ANN provide incorrect results. Hence, it is essential to analyze the impact of noise on the performance and accuracy of the trained ANN, prior to its deployment in the ML-based system.

This chapter elaborated on the numerous ways in which noise may affect the accuracy and performance, in terms of network robustness, training bias, and sensitivity of individual input nodes, of a trained ANN. The applicable noise models, based on L^p norms and relative noise, were also provided. The knowledge of the possible impacts of noise and noise modeling is then leveraged in a framework for formal analysis of neural networks (FANNet).

The chapter also provided a case study to study the impact of noise on an ANN trained on real-world dataset. As expected, beyond the noise tolerance of the trained ANN, the increase in applied noise reduced the classification accuracy of the network. In addition, this reduction in classification was more drastic for certain output classes, due to the training bias. Moreover, depending on the sensitivity of individual input nodes, the vulnerability of nodes to noise also varied.

While L^p norm-based noise models are often a popular choice for the ANN robustness analysis, the chapter also emphasized its inadequacy for analyzing impacts of noise beyond robustness. Particularly, these models are not ideal for ANNs where the different input nodes have different ranges of values. The choice of the best-suited noise model, along with a more broad-spectrum noise analysis, is an essential tool for ensuring the high accuracy of ML-based systems deploying ANNs in noisy, real-world environments.

References

1. Botoeva, E., Kouvaros, P., Kronqvist, J., Lomuscio, A., Misener, R.: Efficient verification of ReLU-based neural networks via dependency analysis. In: Proc. AAAI (2020)
2. Bunel, R., Lu, J., Turkaslan, I., Kohli, P., Torr, P., Mudigonda, P.: Branch and bound for piecewise linear neural network verification. JMLR **21** (2020)
3. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: Symposium on Security and Privacy (SP), pp. 39–57. IEEE, Piscataway (2017)
4. Cheng, C.H., Nührenberg, G., Huang, C.H., Ruess, H.: Verification of binarized neural networks via inter-neuron factoring. In: Proc. VSTTE, pp. 279–290. Springer, Berlin (2018)
5. Dehnert, C., Junges, S., Katoen, J.P., Volk, M.: A storm is coming: a modern probabilistic model checker. In: International Conference on Computer Aided Verification, pp. 592–600. Springer, Berlin (2017)
6. Dutta, S., Jha, S., Sankaranarayanan, S., Tiwari, A.: Output range analysis for deep feedforward neural networks. In: Proc. NFM, pp. 121–138. Springer, Berlin (2018)
7. Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., Dean, J.: A guide to deep learning in healthcare. Nat. Med. **25**(1), 24 (2019)
8. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science **286**(5439), 531–537 (1999)
9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: Proc. ICLR (2015)
10. Huang, X., Kwiatkowska, M., Wang, S., Wu, M.: Safety verification of deep neural networks. In: Proc. CAV, pp. 3–29. Springer, Berlin (2017)
11. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: an efficient SMT solver for verifying deep neural networks. In: Proc. CAV, pp. 97–117. Springer, Berlin (2017)

12. Katz, G., Huang, D.A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., et al.: The Marabou framework for verification and analysis of deep neural networks. In: International Conference on Computer Aided Verification, pp. 443–452. Springer, Berlin (2019)
13. Khalid, F., Ali, H., Hanif, M.A., Rehman, S., Ahmed, R., Shafique, M.: FaDec: a fast decision-based attack for adversarial machine learning. In: Proc. IJCNN, pp. 1–8. IEEE, Piscataway (2020)
14. Khalid, F., Hanif, M.A., Rehman, S., Ahmed, R., Shafique, M.: TrISec: training data-unaware imperceptible security attacks on deep neural networks. In: Proc. IOLTS. IEEE/ACM (2019)
15. Khalid, F., Hanif, M.A., Shafique, M.: Exploiting vulnerabilities in deep neural networks: adversarial and fault-injection attacks (2021). arXiv preprint arXiv:2105.03251
16. Khan, S., Ahmad, J., Naseem, I., Moinuddin, M.: A novel fractional gradient-based learning algorithm for recurrent neural networks. CSSP **37**(2), 593–612 (2018)
17. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial examples in the physical world. In: International Conference on Learning Representations, ICLR, pp. 1–14 (2017)
18. Li, G., Yang, Y., Qu, X., Cao, D., Li, K.: A deep learning based image enhancement approach for autonomous driving at night. Knowl.-Based Syst. **213**, 106617 (2021)
19. Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., Frossard, P.: Universal adversarial perturbations. In: Proc. CVPR, pp. 1765–1773 (2017)
20. Müller, M.N., Makarchuk, G., Singh, G., Püschel, M., Vechev, M.: PRIMA: general and precise neural network certification via scalable convex hull approximations. Proc. POPL **6**(POPL), 1–33 (2022)
21. Nanda, V., Dooley, S., Singla, S., Feizi, S., Dickerson, J.P.: Fairness through robustness: investigating robustness disparity in deep learning. In: Proc. FAccT, pp. 466–477 (2021)
22. Narodytska, N., Kasiviswanathan, S.P., Ryzhyk, L., Sagiv, M., Walsh, T.: Verifying properties of binarized deep neural networks. In: Proc. AAAI, pp. 6615–6624 (2018)
23. Naseer, M., Minhas, M.F., Khalid, F., Hanif, M.A., Hasan, O., Shafique, M.: FANNet: Formal analysis of noise tolerance, training bias and input sensitivity in neural networks. In: Proc. DATE, pp. 666–669. IEEE, Piscataway (2020)
24. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: Symposium on Security and Privacy (SP), pp. 372–387. IEEE, Piscataway (2016)
25. Pulina, L., Tacchella, A.: Challenging SMT solvers to verify neural networks. AI Commun. **25**(2), 117–135 (2012)
26. Ratasich, D., Khalid, F., Geissler, F., Grosu, R., Shafique, M., Bartocci, E.: A roadmap toward the resilient Internet of Things for cyber-physical systems. IEEE Access **7**, 13260–13283 (2019)
27. Shafique, M., Naseer, M., Theocharides, T., Kyrkou, C., Mutlu, O., Orosa, L., Choi, J.: Robust machine learning systems: Challenges, current trends, perspectives, and the road ahead. Design Test **37**(2), 30–57 (2020)
28. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks (2013). arXiv preprint arXiv:1312.6199
29. Tjeng, V., Xiao, K.Y., Tedrake, R.: Evaluating robustness of neural networks with mixed integer programming. In: Proc. ICLR (2019)
30. Tran, H.D., Pal, N., Musau, P., Lopez, D.M., Hamilton, N., Yang, X., Bak, S., Johnson, T.T.: Robustness verification of semantic segmentation neural networks using relaxed reachability. In: Proc. CAV, pp. 263–286. Springer, Berlin (2021)
31. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Efficient formal safety analysis of neural networks. In: Proc. NeurIPS, pp. 6367–6377 (2018)
32. Wiyatno, R., Xu, A.: Maximal Jacobian-based saliency map attack (2018). arXiv preprint arXiv:1808.07945