BARS: Local Robustness Certification for Deep Learning based Traffic Analysis Systems

Kai Wang*†, Zhiliang Wang*†‡, Dongqi Han*†, Wenqi Chen*†, Jiahai Yang*†‡, Xingang Shi*†, Xia Yin†§
*Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing, China
†Zhongguancun Laboratory, Beijing, China
‡Quan Cheng Laboratory, Jinan, Shandong, China
§Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China
{k-wang20, handq19, chenwq19}@mails.tsinghua.edu.cn, {wzl, yang, shixg}@cernet.edu.cn, yxia@tsinghua.edu.cn

***Abstract*—Deep learning (DL) performs well in many traffic analysis tasks. Nevertheless, the vulnerability of deep learning weakens the real-world performance of these traffic analyzers (e.g., suffering from evasion attack). Many studies in recent years focused on robustness certification for DL-based models. But existing methods perform far from perfectly in the traffic analysis domain. In this paper, we try to match three attributes of DL-based traffic analysis systems at the same time: (1) highly heterogeneous features, (2) varied model designs, (3) adversarial operating environments. Therefore, we propose `BARS`, a general robustness certification framework for DL-based traffic analysis systems based on boundary-adaptive randomized smoothing. To obtain tighter robustness guarantee, `BARS` uses optimized smoothing noise converging on the classification boundary. We firstly propose the *Distribution Transformer* for generating optimized smoothing noise. Then to optimize the smoothing noise, we propose *some special distribution functions and two gradient based searching algorithms for noise shape and noise scale*. We implement and evaluate `BARS` in three practical DL-based traffic analysis systems. Experiment results show that `BARS` can achieve tighter robustness guarantee than baseline methods. Furthermore, we illustrate the practicability of `BARS` through five application cases (e.g., quantitatively evaluating robustness).**

I. Introduction

Network traffic is one of the most important data sources for analyzing network activities and detecting cyberspace attack. In recent years, deep learning (DL) has been widely applied for traffic analysis systems, such as network intrusion detection systems (NIDS) [51], [49], concept drift traffic detection systems [74], traffic multi-classification systems [16], [65], [60], [56], etc. However, many studies indicate deep learning is vulnerable to data perturbation [24], [63], [39], [48], [14], which may cause misclassification of DL-based traffic analysis systems [61], [5], [54], [28], [6], [2]. It is thus especially necessary to focus on robustness analysis and improvement of DL-based traffic analysis systems, due to the high cost of misclassification in this domain.

Many studies in recent years focused on robustness certification [45], [72], [66], [34], [64], [77], [71], [15], [73], [44], [52], [42], [11], [43] to analyze and improve the robustness of DL-based models. They theoretically prove whether a DL-based model satisfies special robustness properties. Existing robustness certification studies have achieved excellent performance in computer vision (CV) [72], [15] and natural language processing (NLP) [58], [19]. However, little attention has been paid to robustness certification for DL-based traffic analyzers.

TABLE I: Comparison of robustness certification methods.

Method	Global	Complete	Relaxed	Probabilistic	
	`G.R.P.`♠ [11]	α-`CROWN` [72] β-`CROWN` [66]	`CROWN-IBP` [77]	`V.R.S.`♣ [15]	`BARS`
Heterogeneity Adaptability♡	○	○	○	○	●
Universality	●	○	○	●	●
Real-time Capability	○	◐	●	●	●

* ♡ The adaptability for heterogeneous features. ♠ `G`lobal `R`obustness `P`roperty. ♣ `V`anilla `R`andomized `S`moothing.
* ● = excellent performance. ◐ = normal performance. ○ = poor performance.

It is challenging to propose a suitable robustness certification method for DL-based traffic analysis systems due to their three attributes:

I. **Different from image and natural language, traffic features are highly heterogeneous [55], [57], [41].** Different from pixels and words, traffic features have different physical meanings (e.g., flow duration and total packets) [57].
II. **Due to the difficulties in detecting and classifying malicious activities, model designs of traffic analyzers are varied [5], [51], [16].** For example, `Kitsune` [51] uses ensemble autoencoders under zero-positive learning to detect zero-day attack. `ACID` [16] uses a supervised adaptive clustering network to classify malicious traffic.
III. **Traffic analyzers always run in adversarial environments [61], [5], [54], [28], [6], [2].** Due to its security sensitivity, adversarial attack against traffic analysis will cause more damage (e.g., privacy disclosure, system destroy) [7]. A real-time detection technology is needed to be aware of these adversarial attacks.

Existing robustness certification studies can be divided into global certification and local certification. *Global certification* focuses on certifying the global robustness properties which are independent of data distributions [11], [43]. However, it mainly has two drawbacks. Firstly, global certification cannot give the local robustness region around a specific sample, whose size reflects the relative distance between the sample and the classification boundary. Secondly, global certification

Network and Distributed System Security (NDSS) Symposium 2023
27 February - 3 March 2023, San Diego, CA, USA
ISBN 1-891562-83-5
https://dx.doi.org/10.14722/ndss.2023.24508
www.ndss-symposium.org

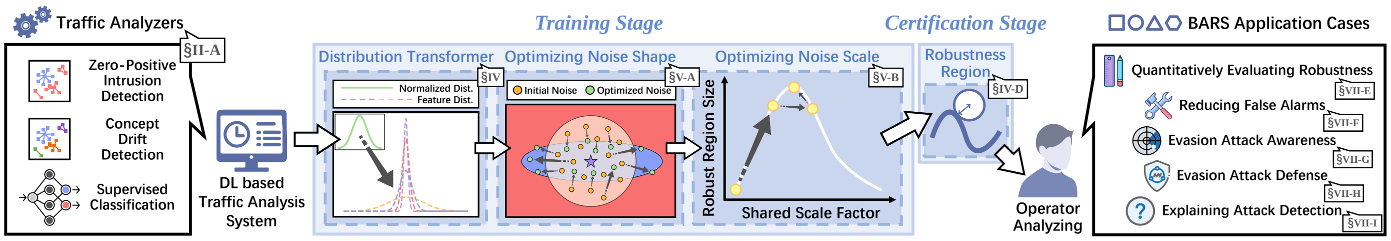


Fig. 1: The workflow of this paper.

cannot certify robustness with real-time traffic after traffic analysis systems are deployed, which conflicts with the attribute III of traffic analyzers. Therefore, it is necessary to certify the local robustness of DL-based traffic analyzers.

Unfortunately, existing local robustness certification methods [45] perform far from perfectly in DL-based traffic analyzers. They mainly include three categories of methods: *Complete certification* (e.g., α-CROWN, β-CROWN) [72], [66] always suffers from high computational complexity and has low efficiency for real-time analysis. It conflicts with the attribute III of traffic analyzers. *Linear relaxation certification* (e.g., CROWN-IBP) [77] needs to be specifically designed according to model architectures and has poor universality and flexibility, which also exists in most complete certification methods. It conflicts with the attribute II of traffic analyzers. *Probabilistic certification* (e.g., Randomized Smoothing) [15], [73] traditionally uses isotropic classical noise as smoothing noise. Thus it only provides ℓ_p robustness guarantee, which provides the same robustness region size for all dimensions. Furthermore, to our knowledge, most existing local certification methods mainly focus on the ℓ_p robustness. Due to the heterogeneity of traffic features, they cannot provide tight robustness guarantee. It conflicts with the attribute I of traffic analyzers. To our knowledge, there is no special local robustness certification method for DL-based traffic analyzers.

Our Solution. In this paper, we propose BARS (**B**oundary-**A**daptive **R**andomized **S**moothing), a general robustness certification framework for DL-based traffic analysis systems. It should be noted that BARS can also be used for other DL-based heterogeneous tabular data analysis systems. The comparison between BARS and other methods is shown in Table I. BARS overcomes the drawbacks of other methods and matches DL-based traffic analysis systems better. Furthermore, BARS can apply local robustness certification for specific problems in the traffic analysis domain.

A brief workflow of this paper is shown in Figure 1. As shown at the middle of Figure 1, the core idea of BARS is to generate optimized smoothing noise which converges on the classification boundary under heterogeneous features and provides tighter robustness guarantee. Therefore, we propose *Distribution Transformer*, a parameterized model which can convert isotropic classical noise distributions into anisotropic optimized noise distributions and provide the dimension-heterogeneous robustness guarantee. To optimize smoothing noise, we use *the superposition of special distribution functions* for the Distribution Transformer, and use gradient based algorithms to search the parameters of the Distribution Transformer for *optimizing noise shape* (the relative noise distributions between different dimensions) and *optimizing noise scale* (the overall scale of noise distributions in all dimensions).

With optimized smoothing noise, BARS provides tighter robustness guarantee and matches the attribute I of traffic analyzers. Because the above algorithm assumes nothing about the model designs, BARS matches the attribute II of traffic analyzers. Because BARS certifies the local robustness of specific samples, and the classifications of noised samples can be efficiently implemented in parallel. It matches the attribute III of traffic analyzers.

Implementation and Evaluations. As shown in the left part of Figure 1, we implement and evaluate BARS on three practical DL-based traffic analysis systems, including Kitsune [51] (*zero-positive NIDS*), CADE [74] (*concept drift detection system*), ACID [16] (*supervised multi-classification system*). Experiment results show that BARS can provide tighter robustness guarantee than baseline methods in all three traffic analyzers, which indicates the significant heterogeneous feature adaptability and universality of BARS. Experiment results also show that in real-time certification, BARS spends much shorter delay time compared with complete certification (reduced by 99% on average), which indicates the significant scalability and real-time capability of BARS. Furthermore, as shown in the right part of Figure 1, we provide five practical cases to show how security operators use BARS to quantitatively evaluate robustness, reduce false alarms, be aware of evasion attack, defend against evasion attack, explain attack detection for DL-based traffic analysis systems.

Contributions. This paper makes three contributions:

- We propose BARS, a general robustness certification framework for DL-based traffic analyzers. The core of BARS is to generate anisotropic optimized smoothing noise for obtaining tighter robustness guarantee. To optimize smoothing noise, we use the superposition of special distribution functions and gradient based parameter searching algorithms.
- We implement BARS on three practical DL-based traffic analyzers[1], and evaluate robustness guarantee tightness and certification delay of BARS. Results show BARS significantly outperforms baseline methods in three traffic analyzers.
- We apply BARS to five domain-specific problems of DL-based traffic analysis, such as quantitatively evaluating robustness, reducing false alarms.

II. Background

In this section, we firstly introduce three categories and three attributes of DL-based traffic analyzers (§II-A). Then we introduce existing studies for robustness certification (§II-B).

A. DL-based Traffic Analysis Systems

Deep learning has been widely applied to traffic analysis tasks [51], [74], [16], [60], [53]. In this paper, we discuss three

[1]Code of BARS is released at: https://github.com/KaiWangGitHub/BARS

typical categories of DL-based traffic analysis systems.

DL-based zero-positive NIDS. `Kitsune` is a state-of-the-art zero-positive NIDS [51], which is also called as "zero-positive" learning [17]. The ensemble autoencoders (AEs) of `Kitsune` are trained with benign traffic by minimizing the reconstruction error, which is the *Root Mean Square Error* (**RMSE**) of input data and reconstructed data. Then, `Kitsune` monitors malicious traffic according to RMSE.

DL-based concept drift detection systems (new class traffic detection). A typical concept drift detector is `CADE` [74]. The autoencoder of `CADE` is trained with history traffic from the known classes through contrastive learning. Then, `CADE` detects unforeseen classes according to the detection score.

DL-based supervised multi-classification systems. `ACID` is a state-of-the-art malicious traffic classifier [16]. The adaptive clustering network of `ACID` is trained with history traffic from the known classes. Then, `ACID` classifies real-time traffic into the known classes according to softmax probabilities.

DL-based traffic analyzers mainly have three attributes:

I. Highly heterogeneous features [55], [57], [41]. In CV [30], [59] and NLP [36], [40], each data element has the same physical meaning (i.e., pixels in images, words in sentences). However, traffic features are highly heterogeneous. Different features have different physical meanings and follow different distributions. For example, inter-packet delays follow exponential distribution, while packet sizes do not follow that [55].

II. Varied model designs [5], [51], [16]. The challenges of traffic analysis tasks are various. It is difficult to design a universal model to solve all problems. The state-of-the-art methods in different tasks need different model designs. For example, `Kitsune` uses ensemble autoencoders. `CADE` uses a single autoencoder, `ACID` uses an adaptive clustering network.

III. Adversarial operating environments [61], [5]. Based on the vulnerability of deep learning, evasion attack (typical adversarial attack) can slightly manipulate traffic features to defeat traffic analyzers in real time [54], [28], [6], [2]. Compared with CV and NLP, adversarial attack against traffic analysis will cause more damage, such as privacy disclosure, system destroy, property damage [7]. To be aware of evasion attack, we should develop a real-time detection technology instead of offline technologies.

B. Robustness Certification

Many studies in recent years focused on robustness certification to analyze and improve the robustness of DL-based models. Robustness certification can be divided into global certification and local certification. Furthermore, local certification mainly includes complete certification, linear relaxation certification, probabilistic certification.

Global certification. Global certification mainly includes `Global Robustness Property` based methods [11], etc. They propose some heuristic robustness properties (e.g., small neighborhood), and verify whether a model satisfy those properties. However, they have two drawbacks. Firstly, global robustness properties mainly focus on the robustness of a model in the whole decision region. It cannot give the local robustness region around a specific sample, which can reflect the relative distance between the sample and the classification boundary [52]. Secondly, global robustness property verification has been completed before model deployment. They cannot be used for certifying robustness with real-time data, which conflicts with the attribute III of traffic analyzers. Therefore, it is necessary to explore local robustness certification for DL-based traffic analysis systems.

Complete certification. Complete certification mainly includes Branch and Bound based algorithms (e.g., α-`CROWN` [72], β-`CROWN` [66]), etc. Those methods attempt to certify all possible input values to provide exact robustness guarantee with high computational complexity. It will significantly decrease certification efficiency, which conflicts with the attribute III of traffic analyzers.

Linear relaxation certification. Linear relaxation certification includes linear inequality propagation (e.g., `CROWN-IBP` [77]), etc. Those methods relax nolinear activation functions of DL-based models to linear functions, and will efficiently estimate the ranges of model outputs for certifying robustness. However, they need to design a special relaxation method for each specific activation function (e.g., ReLU) and hidden layer (e.g., fully-connected layer). As more novel models are proposed, it might be difficult to apply them for the models with novel designs, which also exists in most complete certification methods. Therefore, it conflicts with the attribute II of traffic analyzers.

Probabilistic certification. Probabilistic certification mainly includes `Randomized Smoothing` based on Neyman-Pearson lemma [15], [73], etc. These methods certify robustness based on classification results of noised samples, and assume nothing about model architectures. However, their performance is sensitive to smoothing noise distributions [73]. Existing methods use isotropic classical noise distributions. It only provides ℓ_p robustness guarantee, which provides the same robustness region size for all dimensions. Furthermore, to our knowledge, most existing local certification methods mainly focus on the ℓ_p robustness. They cannot provide tight robustness guarantee under the heterogeneous features of traffic analyzers. Therefore, they all conflict with the attribute I of traffic analyzers.

III. Problem Scope

In this section, we firstly introduce threat models (§III-A). Then we formulate the research problem by defining the certification goal (§III-B). Next we present the `BARS` overview (§III-C). Some important notations are shown in Table II.

A. Threat Model

In this section, we introduce two typical threat models for DL-based traffic analysis systems.

Evasion attack. According to the attribute III in Section II-A, suffering from evasion attack is a main challenge to DL-based traffic analysis systems [54], [28], [6], [2]. Based on the vulnerability of deep learning, attackers can slightly manipulate malicious traffic to defeat traffic analysis systems. Manipulated malicious traffic will be misclassified into the benign class with original attack attributes.

False alarms. Suffering from false alarms is another main challenge to DL-based traffic analysis systems [61], [5]. For

TABLE II: Notations.

Notation	Description
$\boldsymbol{x}, \boldsymbol{\delta}, y$	Traffic sample: original, perturbation, label
f, g, s	Traffic analyzer: base, smoothed, score
c_A, p_A	From smoothed traffic analyzer: the predicted class, the lower confidence bound of c_A probability
$\Omega, \boldsymbol{r}, R$	For robustness: region, dimension-wise radius vector, dimension-heterogenous radius
$\mathcal{X}, \mathcal{C}_A, \mathcal{V}, \mathcal{R}$	The set in a batch of: $\boldsymbol{x}$, c_A, $\boldsymbol{r}$, R
$\mathbb{P}, \mathbb{E}, \mathbb{I}$	Probability, expectation, indicator function
$\boldsymbol{\varepsilon}, \mathcal{D}$	Noise, noise distribution
F, H	Cumulative distribution function for: noise distribution, special distribution.
Ψ	Distribution Transformer
$(\cdot)_n, (\cdot)_f$	For special noise: normalized, feature
$(\cdot)_i, (\cdot)^{(i)}$	The i^{th} dimension, the i^{th} sample or function
n_0, n, n_t	Noise number for: c_A, p_A, optimizing noise shape
d, N	Number of: dimensions, dataset samples
$\mathcal{L}, \mathcal{L}_w, \mathcal{L}_c$	Loss function for: noise shape, wrongly-classified and correctly-classified noised samples
Λ, λ	Regularizer, regularizer weight
Mean, Std, sgn	Statistical function: mean, standard deviation, sign
`Lower-Confidence-Bound`	Interface function for lower bound of binomial proportion confidence interval based on Beta distribution
`Certify`	Interface function for certifying robustness

example, it is assumed that concept drift detection systems can theoretically learn all patterns of the traffic from the known classes, so that previously unseen classes can be detected [74]. However, in practice, some known activities may be classified into previously unseen attack. It can be attributed to that the coverage of training traffic for the known classes is insufficient.

B. Certification Goal

Smoothed traffic analyzer. Consider a traffic analyzer $f : \mathbb{R}^d \to \mathcal{Y}$. A smoothed traffic analyzer can be defined as:

$$g(\boldsymbol{x}) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon} \sim \mathcal{D}} \left(f\left(\boldsymbol{x} + \boldsymbol{\varepsilon}\right) = c \right), \tag{1}$$

where $\mathcal{Y}$ is the class set. $\boldsymbol{\varepsilon}$ is the noise perturbation and follows the noise distribution $\mathcal{D}$. Inspired by [15], [73], we can identify c_A and estimate p_A based on $g(\boldsymbol{x})$.

c_A denotes the predicted class of $g(\boldsymbol{x})$. We use a small number of noised samples to identify c_A:

$$c_A = \arg\max_{c \in \mathcal{Y}} \frac{1}{n_0} \sum_{i=1}^{n_0} \mathbb{I}\left\{ f\left(\boldsymbol{x} + \boldsymbol{\varepsilon}^{(i)}\right) = c \right\}, \tag{2}$$

where n_0 is the number of the noised samples. $\boldsymbol{\varepsilon}^{(i)}$ $i = 1 \dots n_0$ is the noise perturbations sampled from the noise distribution $\mathcal{D}$. The indicator function $\mathbb{I} : X \to \{0, 1\}$ satisfies that: (1) When X is true, $\mathbb{I}\{X\}$ is 1. (2) When X is false, $\mathbb{I}\{X\}$ is 0.

Based on that identified c_A, p_A denotes the lower confidence bound of $\mathbb{P}_{\boldsymbol{\varepsilon}\sim\mathcal{D}}(f(\boldsymbol{x} + \boldsymbol{\varepsilon}) = c_A)$. We use a large number of noised samples to estimate p_A:

$$\begin{aligned} p_A &= \texttt{LowerConfidenceBound}\left(n_A, n, \alpha\right), \\ n_A &= \sum_{i=1}^{n} \mathbb{I}\left\{ f\left(\boldsymbol{x} + \boldsymbol{\varepsilon}^{(i)}\right) = c_A \right\}, \end{aligned} \tag{3}$$

where n_A is the number of the noised samples classified into class c_A. n is the number of noised samples. $\boldsymbol{\varepsilon}^{(i)}$ $i = 1 \dots n$ is the noise perturbations sampled from the noise distribution $\mathcal{D}$. `LowerConfidenceBound` is the lower bound of the binomial proportion confidence interval based on Beta distribution [33], [9], whose parameter α is the significance level.

Certification goal. Given a perturbation $\boldsymbol{\delta} \in \mathbb{R}^d$ for the input sample $\boldsymbol{x}$, the local robustness certification of the traffic analyzer is to search the largest local region Ω which satisfies $\forall \boldsymbol{x} + \boldsymbol{\delta} \in \Omega,\ g(\boldsymbol{x}) = g(\boldsymbol{x} + \boldsymbol{\delta})$, where Ω is called as a robustness region. Existing local robustness certification methods mainly focus on the ℓ_p robustness guarantee whose robustness region can be formulated as $\Omega = \left\{ \boldsymbol{x} + \boldsymbol{\delta} \,\middle|\, \|\boldsymbol{\delta}\|_p \le R_{\ell_p} \right\}$ under ℓ_p robustness radius R_{ℓ_p}. It only provides the same robustness region size for all dimensions. Different from CV and NLP, the features of DL-based traffic analysis systems are highly heterogeneous, so that the ℓ_p robustness guarantee is not tight enough. Therefore, we extend the ℓ_p robustness guarantee to the dimension-heterogeneous robustness guarantee in DL-based traffic analysis systems. To quantitatively describe dimension-heterogeneous Ω, we define the *dimension-wise robustness radius vector* $\boldsymbol{r}$ of Ω as follows.

$$r_i = \max\left(\left\{ |\delta_i| \,\middle|\, \boldsymbol{x} + \boldsymbol{\delta} \in \Omega \right\} \right) \quad i = 1, 2 \dots d. \tag{4}$$

To describe the size of Ω, we extend the ℓ_p robustness radius to the *dimension-heterogeneous robustness radius* R of Ω as follows:

$$R = \frac{1}{d} \sum_{i=1}^{d} r_i, \tag{5}$$

where R is the mean robustness radius of Ω in all dimensions. To be noticed, when Ω is under the ℓ_p robustness guarantee, R in Equation 5 is equal to the ℓ_p robustness radius.

C. `BARS` Overview

A brief overview of `BARS` is shown at the middle of Figure 1. `BARS` is a general robustness certification framework for DL-based traffic analysis systems based on boundary-adaptive randomized smoothing. `BARS` consists of two stages: the training stage and the certification stage.

At the training stage, `BARS` consists of two parts: *Distribution Transformer for generating optimized smoothing noise* (§IV) and *two gradient based searching algorithms for optimizing smoothing noise* (§V). (1) We firstly build a Distribution Transformer which generates anisotropic optimized noise shared by all samples. We theoretically prove the robustness guarantee under the generated noise. (2) Then based on the training dataset, we use the two gradient based searching algorithms to optimize noise shape and noise scale of the Distribution Transformer respectively. The above two parts work together for providing tighter robustness guarantee under highly heterogeneous features in the traffic analysis domain.

At the certification stage, based on the certification dataset, `BARS` uses the trained Distribution Transformer to certify the robustness of the DL-based traffic analyzers (§IV-D). Due to the i.i.d. assumptions [22] and the same optimization objectives, the optimized smoothing noise for the training dataset is still suitable for the certification dataset. The certification dataset can be offline testing dataset or online real-time dataset. This robustness certification algorithm assumes nothing about model designs and has high efficiency which guarantee universality and real-time capability respectively.

IV. Certifying Robustness with Distribution Transformer

In this Section, we firstly introduce the motivation of the Distribution Transformer (§IV-A). Then we introduce the designs and the robustness guarantees of two Distribution Transformers (§IV-B, §IV-C). Then we give the robustness certification procedure with the Distribution Transformer (§IV-D).

A. *Motivation behind Distribution Transformer*

As stated in Section II-A, traffic features are highly heterogeneous (attribute I). The isotropic classical smoothing noise and ℓ_p robustness guarantee used in existing methods cannot provide tight robustness guarantee in the traffic analysis domain [15], [73]. Therefore, we propose a model, called as *Distribution Transformer*, for converting isotropic classical smoothing noise into anisotropic optimized smoothing noise and providing dimension-heterogeneous robustness guarantee. For conciseness, we call isotropic classical smoothing noise as normalized noise and call anisotropic optimized smoothing noise as feature noise.

The framework of the Distribution Transformer is shown in Figure 2. Consider normalized noise $\boldsymbol{\varepsilon}_n \in \mathbb{R}^d$ ($\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n$) and feature noise $\boldsymbol{\varepsilon}_f \in \mathbb{R}^d$ ($\boldsymbol{\varepsilon}_f \sim \mathcal{D}_f$). The Distribution Transformer is formally defined as a map from normalized noise to feature noise, $\Psi : \mathbb{R}^d \to \mathbb{R}^d$. Then we have $\boldsymbol{\varepsilon}_f = \Psi(\boldsymbol{\varepsilon}_n)$. In this paper, we mainly consider $\mathcal{D}_n$ satisfying a condition: a. Symmetric distribution, such as isotropic Gaussian $\mathcal{N}(0, \sigma^2 I)$. More details about the condition and the available distribution are provided in Appendix A-A. $\mathcal{D}_f$ will vary according to Ψ types. With the Distribution Transformer, the smoothed traffic analyzer can be reformulated as:

$$g(\boldsymbol{x}) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon}_f \sim \mathcal{D}_f} \left(f(\boldsymbol{x} + \boldsymbol{\varepsilon}_f) = c \right). \tag{6}$$

According to [73], [52], the tightness of the robustness guarantee depends on the noise distribution scales and the noise distribution functions. Therefore, for tight robustness guarantee, we propose two Distribution Transformers. Firstly, *Linear Distribution Transformer* applies anisotropic distributions to feature noise to adapt to different scales and variances of different traffic features. But its distribution functions in different dimensions are the same. We call `BARS` with the Linear Distribution Transformer as `BARS-L`.

Secondly, on the basis of the Linear Distribution Transformer, *General Distribution Transformer* applies special distribution functions to feature noise for more representation capability. It means that we have more chances to select suitable distribution functions for tighter robustness guarantee. Furthermore, inspired by BLUE [35], feature noise can use the superposition of multiple special distributions for more representation capability. BLUE is one of the most practical estimators in the statistical signal processing domain. Similar to BLUE, we define the superposition of multiple special distributions as their weighted linear sum. We call `BARS` with the General Distribution Transformer as `BARS-G`.

B. *Linear Distribution Transformer*

Model design. Consider normalized noise $\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n$. The Linear Distribution Transformer Ψ^L can be formulated as:

$$\begin{aligned} &\Psi^L(\boldsymbol{\varepsilon}_n) = \boldsymbol{w} \odot \boldsymbol{\varepsilon}_n, \\ &w_i = t \cdot \left(w^S + w_i^I \right) \ i = 1, 2 \ldots d, \end{aligned} \tag{7}$$

where $\odot$ is the Hadamard product, and $\boldsymbol{w}$ is the parameter vector of Ψ. $\boldsymbol{w}$ consists of t, w^S and w_i^I which are trainable parameters, and satisfy $t \geq 0$, $w^S \geq 0$, $w_i^I \geq 0$. t is the scale factor shared in all dimensions. The value of t is determined by the searching algorithm in Section V-B. w^S is the weight parameters Shared in all dimensions. w_i^I is the weight parameters only used in the i^{th} dimension Independently. The values of w^S and w_i^I are determined by the searching algorithm in Section V-A. Then we have feature noise $\boldsymbol{\varepsilon}_f = \boldsymbol{w} \odot \boldsymbol{\varepsilon}_n$.

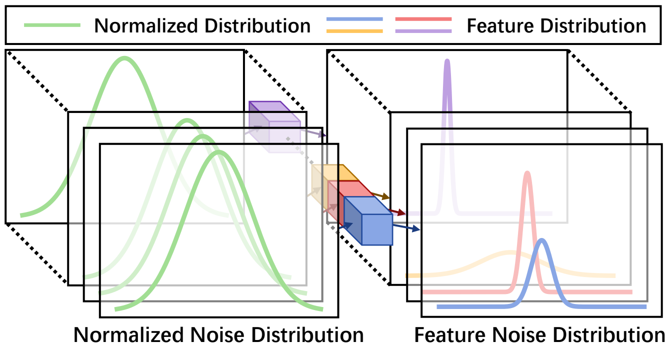


Fig. 2: Distribution Transformer. The model in the middle is the Distribution Transformer. Modules with different colors work in different feature dimensions. Each module converts the normalized noise distribution on the left side into the feature noise distribution on the right side in a feature dimension.

Robustness guarantee. Suppose that the smoothed traffic analyzer has returned c_A in Equation 2 and $\underline{p_A}$ in Equation 3, which denote the predicted class of $g(\boldsymbol{x})$ and the lower confidence bound of $\mathbb{P}_{\boldsymbol{\varepsilon}_f \sim \mathcal{D}_f}(f(\boldsymbol{x} + \boldsymbol{\varepsilon}_f) = c_A)$ respectively. The robustness guarantee with the Linear Distribution Transformer is introduced in Theorem 1 as follows, whose detailed proof is provided in Appendix B-A.

Theorem 1. *Given a smoothed traffic analyzer g and the Linear Distribution Transformer Ψ^L, suppose that the robustness region Ω satisfies:*

$$\begin{aligned} &\Omega = \left\{ \boldsymbol{x} + \boldsymbol{\delta} \, \middle| \, \|\boldsymbol{w}' \odot \boldsymbol{\delta}\|_p \leq r_{\mathcal{D}_n}(\underline{p_A}) \right\}, \\ &w_i' = \frac{1}{t \cdot \left(w^S + w_i^I \right)} \ i = 1, 2 \ldots d, \end{aligned} \tag{8}$$

where $\boldsymbol{\delta}$ is the perturbation, $r_{\mathcal{D}_n}(\underline{p_A})$ is the ℓ_p robustness radius of randomized smoothing [15], [73]. *Then we have:*

$$\forall \boldsymbol{x} + \boldsymbol{\delta} \in \Omega, \ g(\boldsymbol{x}) = g(\boldsymbol{x} + \boldsymbol{\delta}). \tag{9}$$

Based on Equation 4, the dimension-wise robustness radius vector $\boldsymbol{r}$ of Ω in Theorem 1 can be formulated as follows:

$$r_i = w_i \cdot r_{\mathcal{D}_n}(\underline{p_A}) \quad i = 1, 2 \ldots d. \tag{10}$$

Based on Equation 5, the dimension-heterogeneous robustness radius R of Ω in Theorem 1 can be formulated as follows:

$$R = \frac{1}{d} \sum_{i=1}^{d} w_i \cdot r_{\mathcal{D}_n}(\underline{p_A}). \tag{11}$$

Equation 10 and Equation 11 are derived in Appendix B-A.

C. *General Distribution Transformer*

Model design. Consider normalized noise $\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n$ and feature noise $\boldsymbol{\varepsilon}_f \sim \mathcal{D}_f$. Let $F_n : \mathbb{R}^d \to [0, 1]$ and $F_f : \mathbb{R}^d \to [0, 1]$ be the cumulative distribution function (CDF) of $\boldsymbol{\varepsilon}_n$ and $\boldsymbol{\varepsilon}_f$ respectively. Because all dimensions of $\boldsymbol{\varepsilon}_n$ and $\boldsymbol{\varepsilon}_f$ are independent, F_n and F_f can be decomposed as $F_n(\boldsymbol{\varepsilon}_n) = \prod_{i=1}^{d} F_{n,i}(\varepsilon_{n,i})$, $F_f(\boldsymbol{\varepsilon}_f) = \prod_{i=1}^{d} F_{f,i}(\varepsilon_{f,i})$.

It should be noted that for those CDFs which are not strictly monotone increasing in $\mathbb{R}$ (e.g., Uniform), we should change the domain of definition from $\mathbb{R}$ to its subset in which the probability of noise is not zero and the CDF is strictly monotone increasing (e.g., $[-\lambda, \lambda)$ in $\mathcal{U}(-\lambda, \lambda)$). Therefore, in this paper, both of $F_{n,i}$ and $F_{f,i}$ are strictly monotone increasing and reversible. For ease of understanding and no loss of generality, we will only consider that F_n and F_f are strictly monotone increasing in $\mathbb{R}$ in the rest of this paper.

General Distribution Transformer Ψ^G can be defined as:

$$\Psi_i^G(\varepsilon_{n,i}) = F_{f,i}^{-1} \circ F_{n,i}(\varepsilon_{n,i}) \quad i = 1, 2 \ldots d, \tag{12}$$

where Ψ_i^G is the sub-formula of Ψ^G in the i^{th} dimension. Ψ^G has four properties: a. Distribution transformation, b. Strictly monotone increasing in all dimensions, c. Reversibility, d. Odd in all dimensions. These properties will be used to prove the robustness guarantee with the General Distribution Transformer. More details are given in Appendix A-B.

To provide tighter robustness guarantee in the traffic analysis domain, we set $\mathcal{D}_f$ as the weighted linear sum of multiple anisotropic special distributions. $F_{f,i}$ can be formulated in the form of $F_{f,i}^{-1}$ as:

$$\begin{aligned} &F_{f,i}^{-1}(p_i) = t \cdot \sum_{k=1}^{K} \left(w_k^S + w_{i,k}^I\right) \left(H_f^{(k)}\right)^{-1}(p_i) \quad i = 1, 2 \ldots d, \\ &p_i = F_{n,i}(\varepsilon_{n,i}) \quad i = 1, 2 \ldots d, \end{aligned} \tag{13}$$

where $H_f^{(k)} : \mathbb{R} \to [0, 1]$ is the CDF of the k^{th} special distribution and K is the number of special distributions. t, w_k^S and $w_{i,k}^I$ are trainable parameters of F_f, and satisfy $t \geq 0$, $w_k^S \geq 0$, $w_{i,k}^I \geq 0$. t is the scale factor shared by all special noise distributions in all dimensions. w_k^S is the weight parameters Shared by the k^{th} special distribution in all dimensions. $w_{i,k}^I$ is the weight parameters only used by the k^{th} special distribution in the i^{th} dimension Independently. $H_f^{(k)}$ needs to satisfy four conditions: a. Bounded, b. Strictly monotone increasing and reversible, c. Continuity, d. Symmetric distribution. These properties can be used to select feature noise distribution functions. More details are given in Appendix A-C.

The General Distribution Transformer gives us more chances to choose suitable distribution functions to provide tighter robustness guarantee. $H_f^{(k)}$ could be not only common distribution functions (e.g., Gaussian distribution), but also other special distribution functions which satisfy the above conditions (e.g., Sigmoid distribution). More available distribution functions are provided in Appendix A-C.

Robustness guarantee. The robustness guarantee with the General Distribution Transformer is introduced in Theorem 2 as follows, whose detailed proof is provided in Appendix B-B. For that, we need to extend the smoothed traffic analyzer $g(\boldsymbol{x})$ to $g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f)$ by regarding feature noise $\boldsymbol{\varepsilon}_f$ as a variable.

Theorem 2. *Given a smoothed traffic analyzer g and the General Distribution Transformer Ψ^G. Suppose that the robustness region Ω satisfies:*

$$\Omega = \left\{ \boldsymbol{x} + \boldsymbol{\delta} \,\middle|\, \left\| \left(\Psi^G\right)^{-1}(\boldsymbol{\delta}) \right\|_p \leq r_{\mathcal{D}_n}(p_A) \right\}, \tag{14}$$

where $\boldsymbol{\delta}$ is the perturbation, $r_{\mathcal{D}_n}(p_A)$ is the ℓ_p robustness radius of randomized smoothing [15], [73]. *Then we have:*

```
Algorithm 1: Certifying Robustness with Distribution Transformer
   Input:
   Certification set $\mathcal{X} = \{\boldsymbol{x}^{(i)}\}$ $i = 1 \ldots N$. $f$. $\Psi$. Noised sample number $n_0$, $n$.
   Output:
   Certification results $\mathcal{C}_\mathcal{A}$, $\mathcal{V}$, $\mathcal{R}$.
1  $\mathcal{C}_\mathcal{A}, \mathcal{V}, \mathcal{R} \leftarrow \varnothing, \varnothing, \varnothing$
2  for $i \leftarrow 1$ to $N$ do
3      $c_A^{(i)} \leftarrow \arg\max_{c \in \mathcal{Y}} \frac{1}{n_0} \sum_{j=1}^{n_0} \mathbb{I}\{f(\boldsymbol{x}^{(i)} + \boldsymbol{\varepsilon}_f^{(j)}) = c\}$
           $\triangleright$ $\boldsymbol{\varepsilon}_f^{(j)} \sim \mathcal{D}_f$
4      $n_A^{(i)} \leftarrow \sum_{j=1}^{n} \mathbb{I}\{f(\boldsymbol{x}^{(i)} + \boldsymbol{\varepsilon}_f^{(j)}) = c_A\}$
5      $p_A^{(i)} \leftarrow$ LowerConfidenceBound $(n_A^{(i)}, n, \alpha)$
6      $c_A^{(i)} \leftarrow c_A^{(i)}$ if $p_A^{(i)} \geq 0.5$ else ABSTAIN
7      if $\Psi$ is Linear then
8          $\boldsymbol{r}^{(i)} \leftarrow \boldsymbol{w} * r_{\mathcal{D}_n}(p_A^{(i)})$ if $p_A^{(i)} \geq 0.5$ else $\mathbf{0}$
9      end
10     if $\Psi$ is General then
11         $\boldsymbol{r}^{(i)} \leftarrow \Psi(r_{\mathcal{D}_n}(p_A^{(i)}) * \mathbf{1})$ if $p_A^{(i)} \geq 0.5$ else $\mathbf{0}$
12     end
13     $R^{(i)} \leftarrow \frac{1}{d} \sum_{k=1}^{d} r_k^{(i)}$
14     $\mathcal{C}_\mathcal{A}, \mathcal{V}, \mathcal{R} \leftarrow \mathcal{C}_\mathcal{A} \cup \{c_A^{(i)}\}, \mathcal{V} \cup \{\boldsymbol{r}^{(i)}\}, \mathcal{R} \cup \{R^{(i)}\}$
15 end
16 return $\mathcal{C}_\mathcal{A}, \mathcal{V}, \mathcal{R}$
```

$$\begin{aligned} &\forall \boldsymbol{x} + \boldsymbol{\delta} \in \Omega,\; g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f) = g(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{\varepsilon}'_f), \\ &\varepsilon_{f,i} = \Psi_i^G(\varepsilon_{n,i}) \quad i = 1, 2 \ldots d, \\ &\varepsilon'_{f,i} = \Psi_i^G\left(\left(\Psi_i^G\right)^{-1}(\delta_i) + \varepsilon_{n,i}\right) - \delta_i \quad i = 1, 2 \ldots d, \end{aligned} \tag{15}$$

where $\varepsilon_{f,i}$ and $\varepsilon'_{f,i}$ are feature noise, $\varepsilon_{n,i}$ is normalized noise, Ψ_i^G is the sub-formula of Ψ^G in the i^{th} dimension.

Based on Equation 4, the dimension-wise robustness radius vector $\boldsymbol{r}$ of Ω in Theorem 2 can be formulated as follows:

$$r_i = \Psi_i^G(r_{\mathcal{D}_n}(p_A)) \quad i = 1, 2 \ldots d. \tag{16}$$

Based on Equation 5, the dimension-heterogeneous robustness radius R of Ω in Theorem 2 can be formulated as follows:

$$R = \frac{1}{d} \sum_{i=1}^{d} \Psi_i^G(r_{\mathcal{D}_n}(p_A)). \tag{17}$$

Equation 16 and Equation 17 are derived in Appendix B-B.

D. Certification Procedure with Distribution Transformer

The procedure of certifying robustness with the Distribution Transformer is shown in Algorithm 1. Firstly, we identify the predicted class c_A based on Equation 2 and estimate the lower confidence bound of c_A probability based on Equation 3 (line 3 - line 6). Secondly, we calculate the dimension-wise robustness radius vector $\boldsymbol{r}$ and the dimension-heterogeneous robustness radius R based on Equation 10, 11 (line 7 - line 9, line 13) or Equation 16, 17 (line 10 - line 13).

To be noticed, if $p_A < 0.5$ holds, we execute $c_A \leftarrow$ ABSTAIN and $\boldsymbol{r} \leftarrow \mathbf{0}$ for the following reasons [15]. In exact randomized smoothing, we need the lower confidence bound of the top class probability p_A and the upper confidence bound of the runner-up class probability p_B. To simplify calculation, we always use $1 - p_A$ to estimate p_B. To satisfy $p_A \geq p_B$, we need $p_A \geq 1 - p_A$ (i.e., $p_A \geq 0.5$). When $p_A < 0.5$ holds, the estimation is invalid and the local robustness is too weak. We will abstain from making robustness certification with this sample and execute $c_A \leftarrow \mathsf{ABSTAIN}$ and $\boldsymbol{r} \leftarrow \mathbf{0}$.

V. Optimizing Parameters of Noise Distributions

In this section, we discuss how to optimize the feature noise $\boldsymbol{\varepsilon}_f$ generated by the Distribution Transformer. We call the relative noise distributions between different dimensions as the noise shape, and call the overall scale of the noise distributions in all dimensions as the noise scale. We introduce two algorithms for optimizing the noise shape (§V-A) and the noise scale (§V-B) of the feature noise $\boldsymbol{\varepsilon}_f$ respectively. As shown in Figure 1, we should firstly optimize the noise shape and then optimize the noise scale.

A. Optimizing Noise Shape

In this section, we introduce a gradient based algorithm for optimizing the noise shape. In this algorithm, we optimize the weight parameters $w_k^S, w_{i,k}^I$ $_{k=1...K\ i=1...d}$ except the scale factor t. In other words, we set the optimized parameter set as $\Theta = \left\{w_k^S, w_{i,k}^I\right\}$ $_{k=1...K\ i=1...d}$ and fix $t = 1$. To be noticed, we have $K = 1$ in the Linear Distribution Transformer.

Motivation. Noise shape should adapt to highly heterogeneous traffic features to provide tighter robustness guarantee. This problem can be converted into generating feature noise as close to the classification boundary as possible. However, DL-based models are highly nonlinear, so the classification boundary cannot be formulated trivially [26], [27], [79]. Inspired by GAN [23], [4], the Distribution Transformer is treated as a generator, and the traffic analyzer is treated as a discriminator. *Different from GAN, the parameters of the traffic analyzer are fixed, and only the parameter set Θ of the Distribution Transformer is updated based on gradients back-propagated from the traffic analyzer.*

Algorithm. Given a training dataset $\left\{\boldsymbol{x}^{(i)}\right\}$ $_{i=1...N}$, the loss function $\mathcal{L}$ for optimizing the noise shape is constructed as:

$$\begin{aligned}
&\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n}\left(\mathcal{L}_w\left(\boldsymbol{x}^{(i)},\boldsymbol{\varepsilon}_n\right)+\mathcal{L}_c\left(\boldsymbol{x}^{(i)},\boldsymbol{\varepsilon}_n\right)\right)+\lambda\Lambda(\Theta),\\
&\mathcal{L}_w\left(\boldsymbol{x}^{(i)},\boldsymbol{\varepsilon}_n\right)=\mathbb{I}\left\{f\left(\boldsymbol{x}^{(i)}+\Psi\left(\boldsymbol{\varepsilon}_n\right)\right)\neq f\left(\boldsymbol{x}^{(i)}\right)\right\}\cdot\\
&\qquad\qquad\mathcal{L}_{C,w}\left(s\left(\boldsymbol{x}^{(i)}+\Psi\left(\boldsymbol{\varepsilon}_n\right)\right),f\left(\boldsymbol{x}^{(i)}\right)\right),\\
&\mathcal{L}_c\left(\boldsymbol{x}^{(i)},\boldsymbol{\varepsilon}_n\right)=\mathbb{I}\left\{f\left(\boldsymbol{x}^{(i)}+\Psi\left(\boldsymbol{\varepsilon}_n\right)\right)= f\left(\boldsymbol{x}^{(i)}\right)\right\}\cdot\\
&\qquad\qquad\mathcal{L}_{C,c}\left(s\left(\boldsymbol{x}^{(i)}+\Psi\left(\boldsymbol{\varepsilon}_n\right)\right),f\left(\boldsymbol{x}^{(i)}\right)\right),\\
&\Lambda(\Theta)=\sum_{w\in\Theta}\log\left(1+e^{-w}\right),
\end{aligned} \tag{18}$$

where $\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n$ is the normalized noise. Θ is the optimized parameter set of the Distribution Transformer Ψ. $\mathbb{I}$ is the indicator function. f is the base traffic analyzer. s is the classification score function of f (e.g., RMSE of `Kitsune` [51], softmax probabilities of `ACID` [16]). $\mathcal{L}$ consists of three parts including the loss function for wrongly-classified noised samples $\mathcal{L}_w$, the loss function for correctly-classified noised samples $\mathcal{L}_c$, the regularizer $\Lambda(\Theta)$ with a hyper-parameter λ controlling the regularization strength.

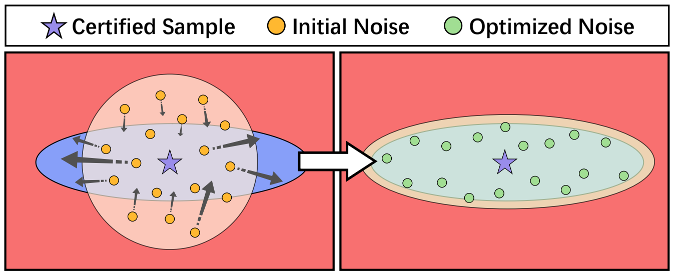


Fig. 3: Optimizing noise shape.

A two dimensional toy example is shown in Figure 3. Minimizing $\mathcal{L}_w$ will make wrongly-classified noised samples (above and below) move towards the correct decision region. Minimizing $\mathcal{L}_c$ will make correctly-classified noised samples (left and right) move towards the wrong decision region. In this way, noised samples will be close to the classification boundary in multiple dimensions. Both $\mathcal{L}_{C,w}$ and $\mathcal{L}_{C,c}$ are derived from the loss function of the specific traffic analyzer. Some practical cases of $\mathcal{L}_{C,w}$ and $\mathcal{L}_{C,c}$ are provided in Section VI. To be noticed, a noised sample cannot exist in both $\mathcal{L}_w$ and $\mathcal{L}_c$ at the same time, but must exist in one of them.

Although minimizing $\mathcal{L}_w$ and $\mathcal{L}_c$ can make noised samples close to the classification boundary in most dimensions, the Distribution Transformer still performs poorly in some dimensions. Thus we introduce a regularizer $\Lambda(\Theta)$. Minimizing $\Lambda(\Theta)$ will maximize the weight parameters in Θ, and will not destroy the effect of $\mathcal{L}_w$ and $\mathcal{L}_c$. It can provide sufficiently tight robustness guarantee in all dimensions.

B. Optimizing Noise Scale

In this section, we introduce a gradient based searching algorithm for optimizing the noise scale. In this algorithm, we fix the weight parameters $w_k^S, w_{i,k}^I$ $_{k=1...K\ i=1...d}$ as the values obtained in Section V-A and only optimize the scale factor t. To be noticed, we have $K = 1$ in the Linear Distribution Transformer.

Motivation. According to [52], the tightness of the robustness guarantee depends on the noise scale. Both of too small and too large noise scale will generate loose robustness guarantee. Therefore, we initialize $t = 0$ and update t in the direction of $\operatorname{sgn}\left(\frac{\partial \bar{R}}{\partial t}\right)$ till $\bar{R}$ has the maximum value (where $\bar{R}$ is the mean of dimension-heterogeneous robustness radii R of the training dataset $\mathcal{X}$).

Algorithm. The procedure of optimizing the noise scale is shown in Algorithm 2. We initialize t to 0 (line 1) and initialize the sign record variable s_{last} to 1 (line 2) due to $\operatorname{sgn}\left(\left.\frac{\partial \bar{R}}{\partial t}\right|_{t=0}\right) = 1$. In each loop iteration, the procedure can be divided into three parts. Firstly, because $\bar{R}$ is not differentiable to t, we add a small perturbation δ_t to t and estimate the gradient of $\bar{R}$ (line 4 - line 8). In the line 5, $\mathbb{I}\left\{c_A^{(i)} = f\left(\boldsymbol{x}^{(i)}\right)\right\}$ is a term against wrong predictions with large robustness radii. Same goes for the line 7. Secondly, we judge whether $\operatorname{sgn}\left(\frac{\partial \bar{R}}{\partial t}\right)$ in this iteration keeps the same as that in the last iteration. If not, we will decay the perturbation δ_t and the update step size γ with the decay factor τ (line 9 - line 14).

Algorithm 2: Optimizing Noise Scale

```
Input:
Training dataset $\mathcal{X} = \left\{\boldsymbol{x}^{(i)}\right\}$ $i = 1 \dots N$. $f$. $\Psi$ with
optimized noise shape. Noised sample number $n_0$, $n$.
Initial perturbation $\delta_t$. Initial update step size $\gamma$.
Decay factor $\tau$. Maximum iteration time $T_i$.
Maximum decay time $T_d$.
Output:
Optimization results $t$, $\mathcal{C}_{\mathcal{A}}$, $\mathcal{V}$, $\mathcal{R}$.
1  $t \leftarrow 0$
2  $s_{last} \leftarrow 1$
3  while $T_i > 0 \wedge T_d > 0$ do
4      $\mathcal{C}_{\mathcal{A}}, \mathcal{V}, \mathcal{R} \leftarrow$ Certify$(\mathcal{X}, f, \Psi_t, n_0, n)$
5      $\bar{R} \leftarrow \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{c_A^{(i)} = f\left(\boldsymbol{x}^{(i)}\right)\right\} * R^{(i)}$
           $\triangleright$ $c_A^{(i)} \in \mathcal{C}_{\mathcal{A}}, \boldsymbol{x}^{(i)} \in \mathcal{X}, R^{(i)} \in \mathcal{R}$ $i = 1 \dots N$
6      $\mathcal{C}'_{\mathcal{A}}, \mathcal{V}', \mathcal{R}' \leftarrow$ Certify$(\mathcal{X}, f, \Psi_{t+\delta_t}, n_0, n)$
7      $\bar{R}' \leftarrow \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{c_A'^{(i)} = f\left(\boldsymbol{x}^{(i)}\right)\right\} * R'^{(i)}$
           $\triangleright$ $c_A'^{(i)} \in \mathcal{C}'_{\mathcal{A}}, \boldsymbol{x}^{(i)} \in \mathcal{X}, R'^{(i)} \in \mathcal{R}'$ $i = 1 \dots N$
8      $g \leftarrow \frac{\bar{R}' - \bar{R}}{\delta_t}$
9      if $\operatorname{sgn}(g) * s_{last} = -1$ then
10         $\delta_t \leftarrow \tau * \delta_t$
11         $\gamma \leftarrow \tau * \gamma$
12         $s_{last} \leftarrow \operatorname{sgn}(g)$
13         $T_d \leftarrow T_d - 1$
14     end
15     $t \leftarrow t + \gamma * \operatorname{sgn}(g)$
16     $T_i \leftarrow T_i - 1$
17 end
18 $\mathcal{C}_{\mathcal{A}}, \mathcal{V}, \mathcal{R} \leftarrow$ Certify$(\mathcal{X}, f, \Psi_t, n_0, n)$
19 return $t, \mathcal{C}_{\mathcal{A}}, \mathcal{V}, \mathcal{R}$
```

Thirdly, we update t with the update step size γ in the direction of $\operatorname{sgn}\left(\frac{\partial \bar{R}}{\partial t}\right)$ (line 15). When the iteration time reaches T_i or the decay time reaches T_d, the algorithm terminates (line 3).

VI. Practical Application Strategy for BARS

In this section, we introduce some practical application strategies for implementing BARS in three traffic analyzers.

Filtering out unforeseen classes. In zero-positive NIDSs (Kitsune [51]) and concept drift detection systems (CADE [74]), the outlier space is large and sparse. Samples from unforeseen classes are quite far away from the classification boundary. It is not necessary to certify their local robustness. Therefore, BARS will not certify the samples which are classified into unforeseen classes by traffic analyzers.

The setting of the normalized noise distribution $\mathcal{D}_n$. We mainly discuss ℓ_2 robustness guarantee for $\mathcal{D}_n$, which is one of the most important ℓ_p robustness guarantees. We set $\mathcal{D}_n$ as the standard Gaussian distribution $\mathcal{N}(0, I)$, which is a state-of-the-art noise distribution for ℓ_2 robustness guarantee [15].

Reducing certification delay. During building the Distribution Transformer, we do not use the exact implementation for Gaussian CDF Φ. For reducing certification delay, inspired by [13], [75], [1], we use the weighted Sigmoid function $\sigma(\alpha \cdot z)$ to approximate Φ based on least squares approximation:

$$\begin{aligned} &\arg\min_{\alpha} \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left(\|\sigma(\alpha \cdot z) - \Phi(z)\|_2^2\right), \\ &\text{s.t.} \qquad \sigma(\alpha \cdot z) = \frac{1}{1 + e^{-\alpha \cdot z}}. \end{aligned} \tag{19}$$

$\mathcal{L}_{C,w}$ and $\mathcal{L}_{C,c}$ of three practical traffic analyzers in Equation 18. As stated in Section V-A, minimizing $\mathcal{L}_{C,w}$ will cause correct classification. Minimizing $\mathcal{L}_{C,c}$ will cause wrong classification.

$\mathcal{L}_{C,w}$ and $\mathcal{L}_{C,c}$ of zero-positive NIDSs (Kitsune) and concept drift detection systems (CADE) are formulated as follows:

$$\begin{aligned} \mathcal{L}_{C,w}(s) &= s - \phi, \\ \mathcal{L}_{C,c}(s) &= \phi - s, \end{aligned} \tag{20}$$

where s denotes $s(\boldsymbol{x} + \Psi(\boldsymbol{\varepsilon}_n))$. ϕ is the detection threshold. In zero-positive NIDSs (Kitsune), s is RMSE of the noised sample. ϕ is the threshold for RMSE. In concept drift detection systems (CADE), s is the final detection score of the noised sample. ϕ is the threshold for detecting outlier samples.

$\mathcal{L}_{C,w}$ and $\mathcal{L}_{C,c}$ of supervised multi-classification systems (ACID [16]) are formulated as follows:

$$\begin{aligned} \mathcal{L}_{C,w}(s, f) &= \frac{1}{|\mathcal{Y}|}\sum_{c \in \mathcal{Y}} \log\left(1 + e^{(\mathbb{I}\{c \neq f\} - \mathbb{I}\{c = f\}) \cdot s_c}\right), \\ \mathcal{L}_{C,c}(s, f) &= \frac{1}{|\mathcal{Y}|}\sum_{c \in \mathcal{Y}} \log\left(1 + e^{(\mathbb{I}\{c = f\} - \mathbb{I}\{c \neq f\}) \cdot s_c}\right), \end{aligned} \tag{21}$$

where s denotes $s(\boldsymbol{x} + \Psi(\boldsymbol{\varepsilon}_n))$ and f denotes $f(\boldsymbol{x})$. In supervised multi-classification systems (ACID), s is the softmax probabilities of the noised sample. f is the classification result of the original training sample.

Class-specific Distribution Transformer. Because traffic feature distributions in different classes are significantly different. To provide tight robustness guarantee, for multi-class datasets, we build a Distribution Transformer for each class separately. Samples from a specific class share the feature noise $\boldsymbol{\varepsilon}_f$ generated by the special Distribution Transformer.

Noise Data Augmentation Retraining. Inspired by [15], [52], we can retrain the base traffic analyzer f with noise data augmentation to improve performance and robustness. The data augmentation loss function $\mathcal{L}'$ is constructed as:

$$\mathcal{L}' = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{\varepsilon}_f \sim \mathcal{D}_f}\left(\mathcal{L}^{\circ}\left(\boldsymbol{x}^{(i)} + \boldsymbol{\varepsilon}_f, y^{(i)}\right)\right), \tag{22}$$

where $\mathcal{L}^{\circ}$ is the base loss function of f, and $\boldsymbol{\varepsilon}_f$ is the feature noise generated by BARS. Minimizing $\mathcal{L}'$ will retrain f to classify noised samples $x^{(i)} + \boldsymbol{\varepsilon}_f$ into the ground truth $y^{(i)}$ of their original samples $x^{(i)}$. The retraining is optional.

Robustness radius for certification dataset. To quantitatively describe the robustness guarantee of the traffic analyzer under a certification dataset, inspired by [15], we extend the dimension-wise robustness radius vector $\boldsymbol{r}$ (Equation 4, Equation 10, Equation 16) to its mean $\bar{\boldsymbol{r}}$ in a dataset. We define the j^{th} dimension of $\bar{\boldsymbol{r}}$ as $\bar{r}_j = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{c_A^{(i)} = y^{(i)}\right\} \cdot r_j^{(i)}$, where $\mathbb{I}\left\{c_A^{(i)} = y^{(i)}\right\}$ is the term against wrong predictions with large robustness radii. Besides, we extend the dimension-heterogeneous robustness radius R (Equation 5, Equation 11, Equation 17) to its mean in a dataset which is called

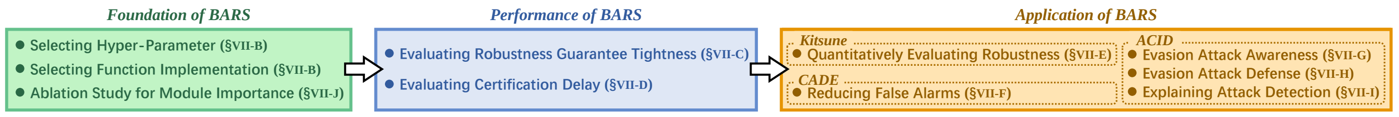

Foundation of BARS
● Selecting Hyper-Parameter (§VII-B)
● Selecting Function Implementation (§VII-B)
● Ablation Study for Module Importance (§VII-J)
Performance of BARS
● Evaluating Robustness Guarantee Tightness (§VII-C)
● Evaluating Certification Delay (§VII-D)
Application of BARS
Kitsune
● Quantitatively Evaluating Robustness (§VII-E)
CADE
● Reducing False Alarms (§VII-F)
ACID
● Evasion Attack Awareness (§VII-G)
● Evasion Attack Defense (§VII-H)
● Explaining Attack Detection (§VII-I)

Fig. 4: The roadmap of experiments.

TABLE III: The sizes of experiment datasets and the special distribution functions H_f used for building feature noise distribution functions F_f for BARS-G in different traffic classes and traffic analyzers. In Kitsune, each attack-specific dataset consists of benign traffic and malicious traffic. In CADE and ACID, the datasets consist of corresponding four classes from IDS2018.

	Kitsune [51]		CADE [74] (IDS2018 [57])				ACID [16] (IDS2018 [57])			
	Mirai	SSDP-Flood	Benign	SSH-Bruteforce	DoS-Hulk	Infiltration	Benign	FTP-Bruteforce	DDoS-HOIC	Botnet-Zeus&Ares
Training	55000	110000	52996	9385	34789	7390	52996	12590	53476	22584
Certification	709137	1390000	13249	2346	8697	1847	13249	3148	13369	5646
H_f	ISRU	ISRU	ISRU	ISRU Gaussian Arctan	ISRU Gaussian	ISRU Gaussian Arctan	ISRU	ISRU	ISRU	ISRU

as *mean robustness radius* (**MRR**). We define MRR as $\frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{c_A^{(i)}=y^{(i)}\right\}\cdot R^{(i)}$. Between traffic analyzers, larger MRR indicates stronger robustness. Between certification methods, larger MRR indicates tighter robustness guarantee.

VII. Experiments

In this section, we firstly introduce the experimental setup (§VII-A). Then we evaluate hyper-parameter values and function implementations (§VII-B). Then we perform two comparison experiments of BARS and baseline methods (§VII-C, §VII-D). Besides, we provide five application cases of BARS (§VII-E, §VII-F, §VII-G, §VII-H, §VII-I). Finally, we perform the ablation study of algorithm modules in BARS (§VII-J). As shown in Figure 4, we provide the experiment roadmap of this paper to show the relationships of the experiments.

A. Experimental Setup

DL-based traffic analysis systems. We evaluate the performance of BARS in three DL-based traffic analyzers: Kitsune [51] (zero-positive NIDS), CADE [74] (concept drift detection system), ACID [16] (supervised multi-classification system).

The experiment datasets are shown in Table III. The datasets for Kitsune are based on the open-source datasets in the original paper. These datasets are built with the traffic in a IoT network. We use two typical attack-specific datasets, Mirai and SSDP-Flood. We use the first 55000 and 110000 packets for training respectively and use the others for certification. The datasets for CADE and ACID are based on IDS2018 [57]. IDS2018 use flow features to profile network traffic in a large network testbed. In CADE, we certify the Benign class, the SSH-Bruteforce class, the DoS-Hulk class with the New Infiltration dataset which treats the Infiltration class as the drift class. We certify the Infiltration class with the New DoS-Hulk dataset which treats the DoS-Hulk class as the drift class. In ACID, we build a four-class dataset with the Benign class, the FTP-Bruteforce class, the DDoS-HOIC class, the Botnet-Zeus&Ares class from IDS2018. In both of CADE and ACID, we split the training-certification sets with a ratio of 8:2.

To accurately certify the robustness of the three traffic analyzers, we implement them based on their open-source code (Kitsune[2], CADE[3], ACID[4]). We use the parameter values in the original papers. All three traffic analyzers achieve the similar performances to the original papers.

[2]Kitsune: https://github.com/ymirsky/Kitsune-py
[3]CADE: https://github.com/whyisyoung/CADE
[4]ACID: https://github.com/Mobile-Intelligence-Lab/ACID

The settings of BARS. For the normalized noise distribution $\mathcal{D}_n$ of BARS-L and BARS-G, as stated in Section VI, we mainly discuss ℓ_2 robustness guarantee and set $\mathcal{D}_n$ as the standard Gaussian distribution $\mathcal{N}(0, I)$. For experiment fairness, the settings of BARS-L and BARS-G are the same except for feature noise distribution functions F_f. For the feature noise distribution $\mathcal{D}_f$ of BARS-L, according to Section IV-B, F_f of BARS-L is the same as the function of $\mathcal{D}_n$ (Gaussian distribution function Φ). For the feature noise distribution $\mathcal{D}_f$ of BARS-G, due to the differences between traffic classes and the differences between traffic analyzers, we should separately build a suitable F_f with the special distribution functions H_f in Table X for each traffic class and each traffic analyzer. The settings of H_f are shown in Table III. Besides, we set the noised sample number for identifying c_A, estimating p_A, optimizing the noise shape as $n_0 = 100$, $n = 10000$, $n_t = 1000$ respectively. We use the optimizer Adam [38] for optimizing the noise shape.

Baseline methods. We compare BARS with baseline certification methods including CROWN-IBP [77] (linear relaxation certification), α-CROWN [72], β-CROWN [66] (complete certification), Vanilla Randomized Smoothing [15] (probabilistic certification). **V**anilla **R**andomized **S**moothing is abbreviated as **V.R.S.** in this paper. To fairly compare BARS with baseline methods, we use the mature open-source implementations for baseline methods. For CROWN-IBP, α-CROWN, β-CROWN, we use the newest open-source implementation auto_LiRPA[5] which optimizes these methods with general computational graphs [71]. For V.R.S., we use the open-source implementation[6] in the original paper. For experiment fairness, we use the noise optimization algorithms of BARS to optimize V.R.S.. For experiment fairness, we certify ℓ_2 robustness in all baseline methods, which is consistent with the normalized noise of BARS.

Software and hardware. We implement BARS with PyTorch under Python 3. Experiments are conducted on a Dell PowerEdge R740 server with 24-core Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz, 251GB RAM, one NVIDIA GeForce RTX 3090 GPU.

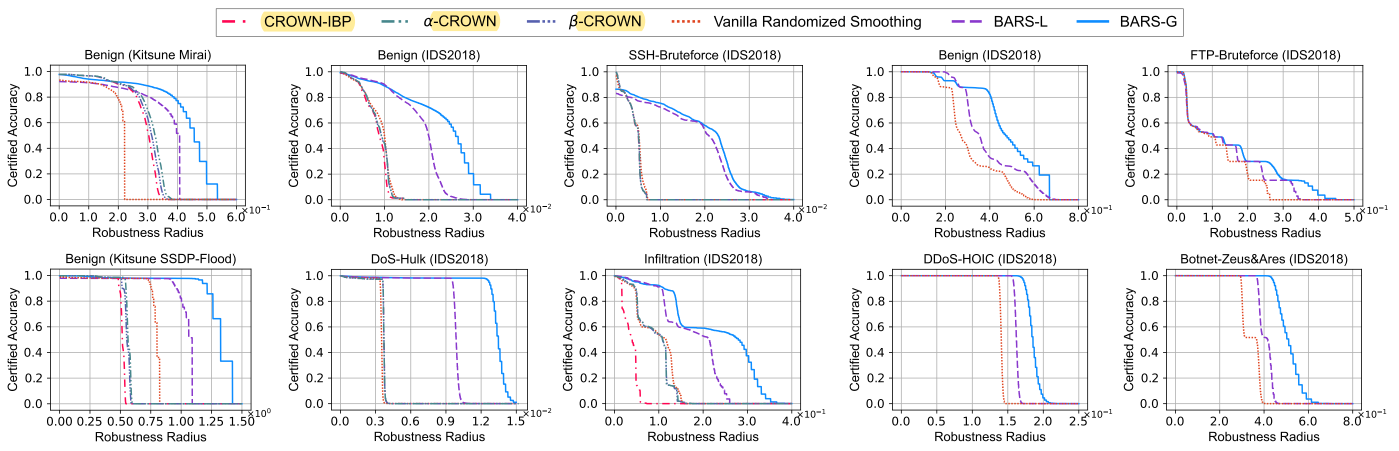
B. Hyper-parameter Value and Function Implementation

In this section, we analyze the influence of two important hyper-parameters and Gaussian distribution function implementations on the performance of BARS. The experiment

[5]auto_LiRPA: https://github.com/KaidiXu/auto_LiRPA
[6]V.R.S.: https://github.com/locuslab/smoothing

TABLE IV: MRR of BARS−G under different noised sample numbers n for estimating p_A and different regularizer weights λ for optimizing noise shapes. Certification delays (sec.) and MRR of BARS−G under exact and approximate Gaussian distribution function implementations. Param., Func., Appro. denote "Parameter", "Function", "Approximate" respectively.

Kitsune [51]			CADE [74] (IDS2018 [57])					ACID [16] (IDS2018 [57])				
Param./Func.	Mirai	SSDP-Flood	Param./Func.	Benign	SSH-Bruteforce	DoS-Hulk	Infiltration	Param./Func.	Benign	FTP-Bruteforce	DDoS-HOIC	Botnet-Zeus&Ares
$n = 1000$	0.2561	0.7448	$n = 1000$	0.0150	0.0153	0.0105	0.1606	$n = 1000$	0.3130	0.1083	0.1367	0.3436
$n = 4000$	0.3571	1.0766	$n = 4000$	0.0201	0.0173	0.0124	0.2058	$n = 4000$	0.4190	0.1364	0.1702	0.4538
$n = 7000$	0.3961	1.2193	$n = 7000$	0.0219	0.0179	0.0130	0.2213	$n = 7000$	0.4588	0.1457	0.1803	0.4908
$n = 10000$	0.4213	1.3059	$n = 10000$	0.0229	0.0182	0.0133	0.2305	$n = 10000$	0.4835	0.1513	0.1856	0.5111
$\lambda = 1 \times 10^{-5}$	0.3687	1.1907	$\lambda = 1 \times 10^{-3}$	0.0136	0.0092	0.0080	0.1371	$\lambda = 1 \times 10^{-4}$	0.4642	0.1475	0.1779	0.4625
$\lambda = 1 \times 10^{-3}$	0.4213	1.3059	$\lambda = 1 \times 10^{-1}$	0.0229	0.0182	0.0133	0.2305	$\lambda = 1 \times 10^{-2}$	0.4835	0.1513	0.1856	0.5111
$\lambda = 1 \times 10^{-1}$	0.3086	1.1541	$\lambda = 1 \times 10^{1}$	0.0171	0.0071	0.0060	0.1022	$\lambda = 1 \times 10^{0}$	0.3681	0.1301	0.1623	0.4059
Exact (Delay/s)	186.47	188.05	Exact (Delay/s)	1543.15	686.23	1326.03	552.09	Exact (Delay/s)	1571.79	494.23	1562.98	882.35
Exact (MRR)	0.1639	0.4764	Exact (MRR)	0.0100	0.0077	0.0116	0.1631	Exact (MRR)	0.2297	0.0693	0.0741	0.2050
Appro. (Delay/s)	20.17	20.04	Appro. (Delay/s)	10.87	3.44	10.39	2.77	Appro. (Delay/s)	34.82	12.67	37.88	23.49
Appro. (MRR)	0.4213	1.3059	Appro. (MRR)	0.0229	0.0182	0.0133	0.2305	Appro. (MRR)	0.4835	0.1513	0.1856	0.5111



(a) Kitsune [51]. (b) CADE [74]. (c) ACID [16].

Fig. 5: Certified accuracy and robustness radius of different robustness certification methods in various traffic analysis systems.

results are shown in Table IV. Due to space limit, we mainly analyze BARS−G. As stated in Section VII-A, the corresponding settings of BARS−L are the same as those of BARS−G.

Important hyper-parameter. As n increases, MRR will increase and the growth rate of MRR will decrease. To obtain tight robustness guarantee with high certification efficiency, we set $n = 10000$ in three traffic analyzers. Both of too small and too large λ will generate the loose robustness guarantee. Thus we select the suitable value for λ in each traffic analyzer.

Gaussian distribution function implementation. Based on the practical application strategy in Section VI, certification delays can be significantly reduced by the approximate Gaussian distribution function with tighter robustness guarantee.

C. Evaluating Robustness Guarantee Tightness

Experiment motivation. When the traffic analyzer is certifiably robust in the robustness region, robustness guarantee tightness is important to measure whether the robustness region is suitable enough for the traffic analyzer [15], [73].

Experiment design. In this experiment, we compare the robustness guarantee tightness of BARS with baseline methods in three traffic analyzers based on certified accuracy and robustness radius. Inspired by [15], we can define the certified accuracy y under the dimension-heterogeneous robustness guarantee. It is the fraction of the certified dataset which is classified correctly and has larger robustness radius R than the given least robustness radius x. Given the least robustness radius x, higher certified accuracy y indicates tighter robustness guarantee of certification methods. We define the certified accuracy-robustness radius curve as $\mathrm{y} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left\{c_A^{(i)} = y^{(i)} \wedge R^{(i)} > \mathrm{x}\right\}$, where y is the certified accuracy at the least robustness radius x.

In CROWN−IBP [77], α-CROWN [72], β-CROWN [66], for plotting the complete curve, we need to run them at many times with different perturbations (100 times in this paper). Thus they have the time complexity $\mathcal{O}(N)$. While in V.R.S. [15] and BARS, we just need to run them at a time with the time complexity $\mathcal{O}(1)$. The dimension-heterogeneous version of CROWN−IBP, α-CROWN, β-CROWN should separately use different perturbations in different dimensions, which will lead to combination explosion. Thus for plotting the complete curve, they have the time complexity $\mathcal{O}\left(N^d\right)$. It is difficult to implement them in a limited time. In future work, we will try to extend them to the dimension-heterogeneous version.

Experiment results. The experiment results are shown in Figure 5. From the results, we have the following observations:

- BARS significantly outperforms baseline methods in all certified DL-based traffic analyzers. It can be attributed to that baseline methods only focus on ℓ_p robustness guarantee which is loose under heterogeneous traffic features.
- BARS−G outperforms BARS−L due to superposition distributions for feature noise.
- CROWN−IBP cannot be used for certifying the robustness of ACID [16]. Because ACID uses activation functions based on the Sine function. However, there is no reasonable linear relaxation method for these activation functions so far. It reflects the poor universality and flexibility of linear relaxation certification. α-CROWN, β-CROWN cannot certify the robustness of ACID due to the same reason.

TABLE V: Comparison of certification delay (sec.) and MRR of different robustness certification methods in various traffic analysis systems. The data format of table cells is "Delay (MRR)". BARS−G (#) denotes #-distribution BARS−G.

Method	Kitsune [51]		CADE [74] (IDS2018 [57])				ACID [16] (IDS2018 [57])			
	Mirai	SSDP-Flood	Benign	SSH-Bruteforce	DoS-Hulk	Infiltration	Benign	FTP-Bruteforce	DDoS-HOIC	Botnet-Zeus&Ares
CROWN−IBP	193.85 (0.2805)	132.34 (0.5140)	59.15 (0.0080)	24.51 (0.0041)	53.99 (0.0036)	23.27 (0.0359)	–	–	–	–
α-CROWN	14122.29 (0.3003)	10251.44 (0.5634)	5792.02 (0.0085)	1491.54 (0.0041)	3398.24 (0.0036)	1096.30 (0.0896)	–	–	–	–
β-CROWN	10055.05 (0.2902)	7362.00 (0.5514)	1480.16 (0.0085)	375.28 (0.0041)	938.11 (0.0036)	283.78 (0.0893)	–	–	–	–
V.R.S.	13.21 (0.1952)	14.19 (0.7815)	5.32 (0.0088)	1.18 (0.0043)	4.82 (0.0035)	0.95 (0.0941)	29.66 (0.3117)	10.40 (0.1107)	35.74 (0.1408)	18.09 (0.3416)
BARS−L	13.79 (0.3414)	14.29 (1.0305)	5.81 (0.0178)	1.23 (0.0167)	5.04 (0.0097)	1.05 (0.1752)	29.88 (0.3953)	11.39 (0.1336)	34.94 (0.1622)	17.73 (0.4070)
BARS−G (1)	**20.17 (0.4213)**	**20.04 (1.3059)**	**10.87 (0.0229)**	2.41 (0.0138)	9.63 (0.0132)	1.95 (0.2302)	**34.82 (0.4835)**	**12.67 (0.1513)**	**37.88 (0.1856)**	**23.49 (0.5111)**
BARS−G (2)	21.41 (0.3546)	20.65 (1.1297)	11.68 (0.0195)	2.59 (0.0170)	**10.39 (0.0133)**	2.13 (0.2244)	35.89 (0.4229)	12.24 (0.1489)	39.86 (0.1846)	23.13 (0.4409)
BARS−G (3)	27.00 (0.3487)	25.63 (1.1310)	15.52 (0.0189)	**3.44 (0.0182)**	13.61 (0.0120)	**2.77 (0.2305)**	39.55 (0.3869)	13.43 (0.0977)	43.97 (0.1322)	24.80 (0.3146)

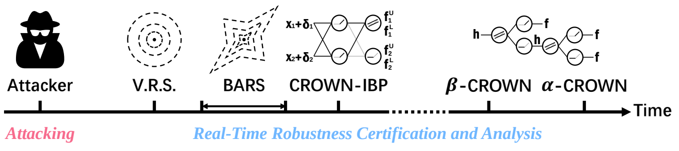


Fig. 6: Real-Time Robustness Certification.

D. Evaluating Certification Delay

Experiment motivation. Certification efficiency is also important in practical application, especially for real-time certification. Certification delay determines whether operators can rapidly respond to attack activities (e.g., evasion attack [54]).

Experiment design. As shown in Figure 6, in this experiment, we compare the certification delay and MRR (defined in Section VI) of BARS with baseline methods in three traffic analyzers. To be aware of evasion attack, we need to obtain the largest robustness radius by plotting a certified accuracy-robustness radius curve as shown in Figrue 5. In real-time certification, it is not realistic to certify all samples of datasets. Therefore, when the dataset size is large enough, we sample 1000 samples in Kitsune [51] and sample 10000 samples in CADE [74] and ACID [16]. Otherwise, we use the entire dataset. To reduce experiment errors and obtain accurate certification delays, we repeat each experiment for 10 times. Then we drop the outlier values and record the mean of non-outlier values as the final result. In BARS−G, the certification delay depends on the number of special distributions. According to Table III, we should evaluate the certification delays of 1, 2, 3-distribution BARS−G in all traffic analyzers. In different traffic analyzers, each #-distribution BARS−G uses the fixed special distributions according to Table III. 1-distribution BARS−G uses ISRU. 2-distribution BARS−G uses ISRU, Gaussian. 3-distribution BARS−G uses ISRU, Gaussian, Arctan.

Experiment results. The experiment results are shown in Table V. From the results, we have the following observations:

- The certification delays of BARS are lower than baseline methods except for V.R.S. [15]. Because noised sample classification of BARS can be implemented in parallel. But compared with V.R.S., BARS uses optimized noise distributions with more computation and parameters.
- The certification delays of complete certification (α-CROWN [72], β-CROWN [66]) are highest. This reflects its weak real-time capability due to high computational complexity.
- As the number of special distributions increases, the certification delays of BARS−G increase due to more computation and parameters. However, the increase is negligible compared with complete certification delays.
- MRR of BARS−G has no causal relationship with the number of special distributions. In Kitsune Mirai, MRR decreases with the increase of the special distribution number. In CADE SSH-Bruteforce, MRR increases with the increase of the special distribution number. In each certification scene, we select the optimal settings of special distributions.

E. Quantitatively Evaluating Robustness

Experiment motivation. During practical deployment, security operators should select suitable hyper-parameter values to balance detection performance and robustness of traffic analyzers [5]. For this purpose, BARS can be used to quantitatively evaluate robustness under different hyper-parameter values.

Experiment design. In this experiment, we use BARS−G to quantitatively evaluate the robustness of Kitsune [51] (zero-positive NIDS). Firstly, we evaluate the robustness of Kitsune under different detection thresholds ϕ. We use MRR (defined in Section VI) and F_1 score to measure the robustness and the detection performance respectively. To guarantee the universality of the experiment results, we repeat the experiment under 32 AEs and 16 AEs. Secondly, we evaluate the robustness of Kitsune under different AE numbers which depend on the maximum cluster feature number m. Inspired by the coefficient of variation [21], we define *Coefficient of Variation for Robustness Radius* (**CVR**) as follows:

$$\text{CVR} = \text{Mean}\left(\mathbb{I}\left\{c_A^{(i)} = y^{(i)}\right\} \cdot \frac{\text{Std}\left(r_j^{(i)}\right)_{j=1\ldots d}}{\text{Mean}\left(r_j^{(i)}\right)_{j=1\ldots d}}\right)_{i=1\ldots N}, \tag{23}$$

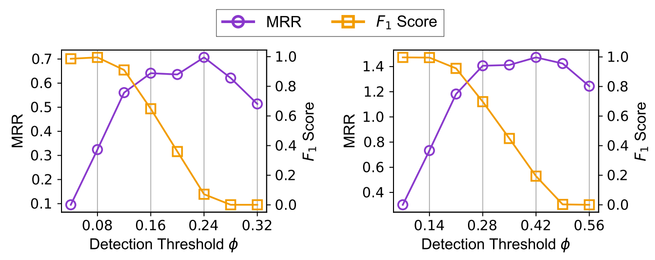
where $r_j^{(i)}$ is the robustness radius in the j^{th} dimension. We use CVR to measure the variability in the robustness radii of different features. Large CVR means that the model can learn the difference between heterogeneous features and can fit the target data precisely. To obtain accurate conclusions, we treat F_1 score as the controlled variable and fix it as 0.98 ± 0.005 by setting suitable detection thresholds ϕ.

Experiment results. Figure 7 is the results of the first experiment. When F_1 score reaches its maximum, MRR is not large enough. It indicates that when we select the detection threshold ϕ, we should consider both of performance and robustness.

Table VI is the results of the second experiment. As the AE number increases, MRR firstly decreases and then increases, and CVR firstly increases and then decreases. It indicates that Kitsune with a small number of AEs has stronger robustness, and has weaker fitting capability. It quantitatively reflects the trade off between robustness and fitting capability.

F. Reducing False Alarms

Experiment motivation. During practical deployment, CADE [74] (concept drift detection system) may not be able to learn all patterns of known classes [61]. It will cause false alarms. Inspired by [15], [52], we can reduce false alarms by retraining CADE with BARS noise data augmentation.



(a) 32 AEs ($m = 7$).
(b) 16 AEs ($m = 16$).

Fig. 7: The MRR and F_1 score of `Kitsune` under different detection thresholds ϕ.

TABLE VI: Comparison of `Kitsune` with different AE numbers in robustness and fitting capability.

AE Number	m	MRR	CVR	F_1 Score
1	100	3.4749	0.1409	0.9796
2	80	4.5540	0.2525	0.9793
4	75	4.2375	0.3740	0.9797
8	43	2.3316	0.6326	0.9806
16	16	0.9923	0.6729	0.9806
32	7	0.4628	0.8025	0.9802
64	2	2.5844	0.3210	0.9784
100	1	2.2312	0.2712	0.9782

Experiment design. In this experiment, we reduce the false alarms of `CADE` in the New Infiltration dataset which treats the Infiltration class as the drift class. According to Table VII, *False Positive Rate* (**FPR**) in the Benign class is highest. To balance effectiveness and efficiency, we use noised Benign samples as augmented data to retrain `CADE`. With the optimized noise of `BARS`, we can reduce FPR and keep *False Negative Rate* (**FNR**) not increasing. We compare `BARS` with `V.R.S.` [15]. Because the model of `CADE` is not softmax structure, the certified robust training of `CROWN-IBP` [77] can hardly be used for it. We also compare `BARS` with two feature selection methods, `WAFS` [76] and `DAFFS` [10].

Experiment results. The experiment results are shown in Table VII. `BARS` outperforms `V.R.S.`. It can be attributed to that the optimized noise of `BARS` adapts to heterogeneous features, and will reduce FPR of `CADE` in all classes. `BARS` outperforms `WAFS` and `DAFFS`. It can be attributed to that noise data augmentation can generate smoother classification boundaries than these feature selection methods.

G. Evasion Attack Awareness

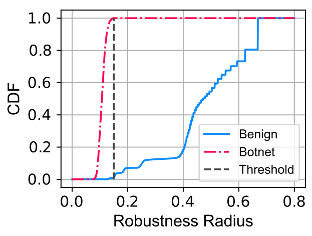
Experiment motivation. As stated in Section II-A, traffic analyzers always run in adversarial environments [54]. We can be aware of evasion attack with `BARS` in real time. Because evasion samples with malicious attributes have different data distributions from clean benign samples. Using the smoothing noise optimized based on clean benign samples, evasion samples will obtain small robustness radii.

Experiment design. In this experiment, we use `BARS` to be aware of the evasion attack of Botnet-Zeus&Ares samples in `ACID` [16] (supervised multi-classification system). Following the existing evasion attack study, Blind Adversarial Perturbations [54], we manipulate Botnet-Zeus&Ares samples to make `ACID` classify them into the Benign class. **B**lind **A**dversarial **P**erturbations is abbreviated as **B.A.P.** in this paper. For guaranteeing effective attack, we implement B.A.P. based on its open-source code[7]. According to [54], B.A.P. cannot be trivially used in non-differentiable functions of traffic features, such as CICFlowMeter [41] of `ACID`. Following [3], [29], we transfer it to perform evasion attack in the feature space, and remap feature perturbations according to feature types. We restrict the perturbation $\boldsymbol{\delta}$ to $\|\boldsymbol{\delta}\|_\infty \leq 0.4$.

[7]B.A.P.: https://github.com/SPIN-UMass/BLANKET

TABLE VII: Reducing false alarms of `CADE` with different methods. Lower is better for all metrics.

Method	FPR				FNR
	Benign	SSH-Bruteforce	DoS-Hulk	Total	Total
Vanilla	0.0495	0.0418	0.0110	0.0350	0.0000
`WAFS`	0.0298	0.0388	0.0230	0.0282	0.0000
`DAFFS`	0.0403	0.0426	0.0138	0.0310	0.0000
`V.R.S.`	0.0296	0.0226	0.0230	0.0266	0.0000
`BARS-L`	0.0315	0.0119	0.0094	0.0217	0.0000
`BARS-G`	0.0283	0.0128	0.0066	0.0190	0.0000



(a) `BARS-G` evasion awareness.

Method	Precision	Recall	F_1 Score
`V.R.S.`	0.6861	0.8380	0.7544
`BARS-L`	0.9819	0.9181	0.9489
`BARS-G`	0.9455	1.0000	0.9720

(b) Evasion awareness method comparison. Higher is better for all metrics.

Fig. 8: Evasion attack awareness for `ACID` Botnet-Zeus&Ares detection with different methods.

For simulating practical operations, we preset a detection threshold according to robustness radius distributions of the Benign class in Figure 5c. After suffering evasion attack, we consider samples with smaller robustness radii than the threshold as evasion samples. We compare `BARS` with `V.R.S.` [15]. As stated in Section VII-C, `CROWN-IBP` [77], α-`CROWN` [72], β-`CROWN` [66] cannot be used for `ACID`. According to the Benign class in Figure 5c, we preset the detection threshold for `BARS-G`, `BARS-L`, `V.R.S.` as 0.15, 0.2, 0.17 respectively.

Experiment results. As shown in Figure 8a, with the robustness radii based on `BARS-G`, we can use a robustness radius threshold to distinguish Botnet-Zeus&Ares samples from Benign samples. As shown in Figure 8b, `BARS` outperforms `V.R.S.`. It can be attributed to that the dimension-heterogeneous smoothing noise of `BARS` will learn more information of Benign sample distributions.

H. Evasion Attack Defense

Experiment motivation. To improve the robustness of traffic analyzers in adversarial environments [54], we retrain them with `BARS` noise data augmentation inspired by [15], [52].

Experiment design. In this experiment, we retrain `ACID` [16] (supervised multi-classification system) with `BARS` noise data augmentation to reduce the evasion success rate of FTP-Bruteforce samples which reflects robustness improvement. We evaluate `BARS` in three evasion methods including Random (Uniform noise perturbations), PGD [48] (gradient based perturbations), B.A.P. [54] (GAN based perturbations). The implementation of B.A.P. is the same as that in Section VII-G. For experiment fairness, we restrict the perturbation $\boldsymbol{\delta}$ to $\|\boldsymbol{\delta}\|_\infty \leq 0.2$ in all three evasion methods. We compare `BARS` with `V.R.S.` [15]. Due to the same reason stated in Section VII-C, the certified robust training of `CROWN-IBP` [77] cannot be used for `ACID`.

TABLE VIII: Explaining DDoS-HOIC detection of ACID.

(a) Important features under BARS−G.

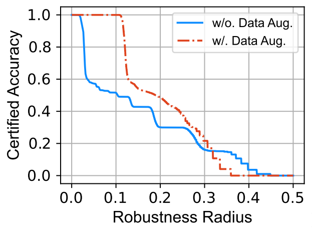
Robustness Radius	Name	Description
5.1728×10^{-2}	Init Fwd Win Byts	Total number of bytes sent in initial window in forward direction.
9.3542×10^{-2}	Fwd IAT Max	Maximum time between two packets sent in forward direction.
1.8561×10^{-1}	Mean Radius	Mean robustness radius in all dimensions.

(b) Evaluating explanation fidelity based on the worsened performance of DDoS-HOIC detection. Lower is better.

Metric	Vanilla	Random	BARS−L	BARS−G
Precision	1.0000	0.9928	0.9423	0.9064
Recall	1.0000	0.9040	0.7918	0.7707
F_1 Score	1.0000	0.9397	0.8605	0.8330

TABLE IX: MRR of BARS−G under different module combinations. The data format of table cells is “MRR (Rank)”. D.T., Opt. Shape, Opt. Scale denote “the Distribution Transformer”, “Optimizing Noise Shape”, “Optimizing Noise Scale” respectively. Higher MRR indicates that left modules are more important.

Module			Kitsune [51]		CADE [74] (IDS2018 [57])				ACID [16] (IDS2018 [57])			
D.T.	Opt. Shape	Opt. Scale	Mirai	SSDP-Flood	Benign	SSH-Bruteforce	DoS-Hulk	Infiltration	Benign	FTP-Bruteforce	DDoS-HOIC	Botnet-Zeus&Ares
✓	✗	✗	0.0957 (5)	0.0980 (5)	0.0000 (7)	0.0000 (7)	0.0000 (7)	0.0629 (7)	0.2407 (5)	0.1260 (3)	0.0922 (5)	0.2476 (5)
✗	✓	✗	0.0267 (7)	0.0195 (6)	0.0072 (6)	0.0032 (6)	0.0028 (6)	0.0763 (6)	0.0267 (6)	0.1107 (5)	0.0000 (6)	0.0000 (6)
✗	✗	✓	0.1962 (3)	0.7797 (4)	0.0087 (5)	0.0043 (3)	0.0035 (4)	0.0915 (5)	0.3130 (3)	0.1110 (4)	0.1466 (3)	0.3415 (4)
✓	✓	✗	0.0350 (6)	0.0053 (7)	0.0106 (2)	0.0056 (2)	0.0072 (2)	0.1623 (2)	0.0018 (7)	0.0000 (7)	0.0000 (6)	0.0000 (6)
✓	✗	✓	0.2666 (2)	0.9118 (2)	0.0097 (3)	0.0040 (5)	0.0037 (3)	0.0969 (3)	0.3937 (2)	0.1268 (2)	0.1780 (2)	0.4476 (2)
✗	✓	✓	0.1952 (4)	0.7815 (3)	0.0088 (4)	0.0043 (3)	0.0035 (4)	0.0941 (4)	0.3117 (4)	0.1107 (5)	0.1408 (4)	0.3416 (3)
✓	✓	✓	0.4213 (1)	1.3059 (1)	0.0229 (1)	0.0182 (1)	0.0133 (1)	0.2305 (1)	0.4835 (1)	0.1513 (1)	0.1856 (1)	0.5111 (1)



(a) BARS−G data augmentation.

Method	Random	PGD	B.A.P.
Vanilla	0.3069	1.0000	1.0000
V.R.S.	0.1131	0.4835	1.0000
BARS−L	0.2789	0.4476	0.8475
BARS−G	0.2024	0.4006	0.8475

(b) Evasion success rate comparison. Lower is better under an attack method.

Fig. 9: Evasion attack defense for ACID FTP-Bruteforce detection with different methods.

Experiment results. As shown in Figure 9a, with BARS−G noise data augmentation, a large number of weakly robust samples have significantly increased robustness radii, and only a small number of strongly robust samples have decreased robustness radii. As shown in Figure 9b, under PGD and B.A.P., BARS outperforms V.R.S.. It can be attributed to the dimension-heterogeneous noise data augmentation of BARS. It can separately improve robustness of different features according to the data scale and the data distribution of each feature. Under Random, V.R.S. outperforms BARS due to isotropic noise perturbations of Random. Nevertheless, its vanilla evasion success rate is lowest.

I. Explaining Attack Detection

Experiment motivation. DL-based traffic analyzers are generally regarded as black boxes with high detection performance. Their detection results can be explained based on the mean dimension-wise robustness radius vector $\bar{\boldsymbol{r}}$ (defined in Section VI) of BARS. Because the detection results of DL-based traffic analyzers are more sensitive to features with small $\bar{r}_j$. These features can be regarded as important features for explanation.

Experiment design. In this experiment, we use BARS to explain the DDoS-HOIC detection of ACID [16] (supervised multi-classification system). We firstly calculate $\bar{\boldsymbol{r}}$ based on all samples classified into the DDoS-HOIC class, and select important features with small $\bar{r}_j$. Then inspired by [67], to evaluate the fidelity of BARS, we replace the original feature values with Uniform noise ($\mathcal{U}(0,1)$ for 0-1 normalization features in this paper) in these important features. Worse performance of ACID indicates that these features are more important. We compare BARS with Random selection in which the same number of features are randomly selected as important features. We repeat experiments of Random selection for 100 times and record the mean performance.

Experiment results. The two important features are shown in Table VIIIa. It reflects that DDoS-HOIC attack sends large volumes of traffic at high rates to make victims unavailable. As shown in Table VIIIb, worse performances caused by BARS−L and BARS−G indicate their selected features are important.

J. Ablation Study

Experiment motivation and design. To illustrate the importance of each algorithm module in BARS, we compare MRR (defined in Section VI) of BARS−G under different module combinations. Three modules include the Distribution Transformer, optimizing the noise shape, optimizing the noise scale. Consider that the only difference between BARS−G and BARS−L is the feature noise distribution. Due to space limit, we mainly perform ablation study for BARS−G.

Experiment results. The experiment results are shown in Table IX. Higher MRR indicates that left algorithm modules are more important. BARS−G with three modules performs best in all three traffic analyzers. It illustrates that all three modules contribute to providing tight robustness guarantee. Observing BARS−G with two modules, in Kitsune [51] and ACID [16], the combination of the Distribution Transformer and optimizing the noise scale is important. In CADE [74], the combination of the Distribution Transformer and optimizing the noise shape is important. Observing BARS−G with one module, optimizing the noise scale is most important in all three traffic analyzers. In Kitsune and ACID, optimizing the noise shape makes the least contribution. In CADE, the Distribution Transformer makes the least contribution.

VIII. Discussion

In this section, we will discuss applications of BARS for more domains, limitation and future work.

Applications for more domains. As stated in Section I, BARS can be used for other DL-based heterogeneous tabular data analysis systems (e.g., malware detection [12], spam URL detection [11], KPI anomaly detection [62]). We will try to apply BARS in these domains in future work. However, BARS cannot be used for sequence data analysis systems (e.g., log anomaly detection [18], [50]) and graph data analysis systems (e.g., lateral movement detection [8], [37]). Because their formal analysis for robustness guarantee is different from that in tabular data based systems [19], [32]. We will extend BARS to these domains through further formal analysis.

Limitation and future work. First, we only search semi-global weight parameters $w_k^S, w_{i,k}^I$ and a semi-global scale factor t for each class. The reasons are: (1) We trade off performance and overhead; (2) If search special $w_k^S, w_{i,k}^I, t$ for each sample in the training dataset, we cannot select $w_k^S, w_{i,k}^I, t$ for certified samples (i.e., low practicability). In future work, we will implement special $w_k^S, w_{i,k}^I, t$ for each training sample with low overhead and high practicability. Second, we only consider a limited number of noise distributions in Appendix A-A and A-C. In future work, we will explore more distributions for improving the performance of BARS. Third, we will explore more practical application cases, such as detecting misclassification between different malicious attacks.

IX. Related Work

DL-based traffic analysis systems. DL-based traffic analysis systems have been extensively studied in recent years. Network intrusion detection systems (NIDS) [51], [49] are used for detecting malicious traffic. Concept drift traffic detection systems [74] (new class traffic detection) are used for detecting traffic from unforeseen classes. Traffic multi-classification systems [16], [65], [60], [56] are used for classifying traffic into a known class.

Robustness certification. Against the vulnerability of deep learning, a large number of studies focused on robustness certification in recent years [45]. It can be divided into global certification and local certification. Global certification focuses on global robustness properties [11], [43]. However, it cannot provide the local robustness guarantee and cannot certify the robustness in real time. Besides, there are mainly three categories of local certification methods: (1) Complete certification mainly includes Branch and Bound [72], [66], satisfiability modulo theories (SMT) [34], [31], mixed-integer linear program (MILP) [64], [20]. They attempt to obtain the exact robustness guarantee with high computational complexity. (2) Linear relaxation certification relaxes the nolinear activation functions of neural networks and has higher efficiency [77], [71], [78], [47], [68], [25], [69], [70]. However, its universality and flexibility are weak because the new relaxation expressions need to be derived again. (3) Probabilistic certification adds noise perturbations to certified samples and certifies robustness based on the classification results of these noised samples [15], [73], [44], [52], [46]. They are sensitive to noise distributions.

Existing certification methods mainly focus on other domains, such as CV, NLP. Due to the characteristics of traffic analysis tasks, they perform far from perfectly in this domain.

X. Conclusion

In this paper, we propose BARS, a general robustness certification framework for DL-based traffic analysis systems based on boundary-adaptive randomized smoothing. To generate boundary-adaptive smoothing noise, BARS uses the Distribution Transformer for generating optimized noise and uses two gradient based algorithms for optimizing the noise. Through evaluating BARS extensively in three practical DL-based traffic analysis systems, we illustrate that BARS can provide the tight robustness guarantee with high efficiency and outperforms the baseline methods significantly. Through five practical application cases, we illustrate the extensive application prospect of BARS. Future work is planned to further improve the performance of BARS, evaluate it in more traffic analyzers, exploit more application scenarios.

Acknowledgment

We would like to heartfeltly thank the anonymous reviewers for their constructive comments. We are also grateful to Han Zhang, Shize Zhang, Bin Xiong, Chenxin Duan, as well as all other members from NMGroup and CNPT-Lab in Tsinghua University. This work is supported by the National Natural Science Foundation of China under Grant No. 62172251. Zhiliang Wang is the corresponding author of this paper.

References

[1] M. Abderrahmane and B. Kamel, "Two new approximations to standard normal distribution function," *Journal of Applied and Computational Mathematics*, vol. 5, p. 328, 2016.

[2] A. Abusnaina, R. Jang, A. Khormali, D. Nyang, and D. Mohaisen, "Dfd: adversarial learning-based approach to defend against website fingerprinting," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 2459–2468.

[3] G. Apruzzese and M. Colajanni, "Evading botnet detectors based on flows and random forest with adversarial samples," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–8.

[4] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

[5] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Dos and don'ts of machine learning in computer security," in *Proc. of the USENIX Security Symposium*, 2022.

[6] A. Bahramali, M. Nasr, A. Houmansadr, D. Goeckel, and D. Towsley, "Robust adversarial attacks against dnn-based wireless communication systems," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 126–140.

[7] BlackBerry, "Blackberry 2022 threat report," https://www.blackberry.com/us/en/forms/enterprise/report-bb-2022-threat-report-aem, 2022, accessed December 3, 2022.

[8] B. Bowman, C. Laprade, Y. Ji, and H. H. Huang, "Detecting lateral movement in enterprise computer networks with unsupervised graph {AI}," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*, 2020, pp. 257–268.

[9] L. D. Brown, T. T. Cai, and A. DasGupta, "Interval estimation for a binomial proportion," *Statistical science*, vol. 16, no. 2, pp. 101–133, 2001.

[10] P. P. Chan, Y. Liang, F. Zhang, and D. S. Yeung, "Distribution-based adversarial filter feature selection against evasion attack," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[11] Y. Chen, S. Wang, Y. Qin, X. Liao, S. Jana, and D. Wagner, "Learning security classifiers with verified global robustness properties," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 477–494.

[12] Y. Chen, S. Wang, D. She, and S. Jana, "On training robust {PDF} malware classifiers," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2343–2360.

[13] A. Choudhury, "A simple approximation to the area under standard normal curve," *Mathematics and Statistics*, vol. 2, no. 3, pp. 147–149, 2014.

[14] K. T. Co, L. Muñoz-González, S. de Maupeou, and E. C. Lupu, "Procedural noise adversarial examples for black-box attacks on deep convolutional networks," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 275–289.

[15] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *International Conference on Machine Learning*. PMLR, 2019, pp. 1310–1320.

[16] A. F. Diallo and P. Patras, "Adaptive clustering-based malicious traffic classification at the network edge," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[17] M. Du, Z. Chen, C. Liu, R. Oak, and D. Song, "Lifelong anomaly detection through unlearning," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1283–1297.

[18] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1285–1298.

[19] T. Du, S. Ji, L. Shen, Y. Zhang, J. Li, J. Shi, C. Fang, J. Yin, R. Beyah, and T. Wang, "Cert-rnn: Towards certifying the robustness of recurrent neural networks," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 516–534.

[20] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep feedforward neural networks," in *NASA Formal Methods Symposium*. Springer, 2018, pp. 121–138.

[21] B. S. Everitt and A. Skrondal, "The cambridge dictionary of statistics," 2010.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, 2016.

[23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[24] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[25] S. Gowal, K. D. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli, "Scalable verified training for provably robust image classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4842–4851.

[26] W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. Xing, "Lemna: Explaining deep learning based security applications," in *proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 2018, pp. 364–379.

[27] D. Han, Z. Wang, W. Chen, Y. Zhong, S. Wang, H. Zhang, J. Yang, X. Shi, and X. Yin, "Deepaid: Interpreting and improving deep learning-based anomaly detection in security applications," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 3197–3217.

[28] D. Han, Z. Wang, Y. Zhong, W. Chen, J. Yang, S. Lu, X. Shi, and X. Yin, "Evaluating and improving adversarial robustness of machine learning-based network intrusion detectors," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2632–2647, 2021.

[29] M. J. Hashemi, G. Cusack, and E. Keller, "Towards evaluation of nidss in adversarial setting," in *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks*, 2019, pp. 14–21.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[31] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *International conference on computer aided verification*. Springer, 2017, pp. 3–29.

[32] J. Jia, B. Wang, X. Cao, and N. Z. Gong, "Certified robustness of community detection against adversarial structural perturbation via randomized smoothing," in *Proceedings of The Web Conference 2020*, 2020, pp. 2718–2724.

[33] P. Josef, S. Skipper, T. Jonathan, and statsmodels developers, "The documentation of statsmodels.stats.proportion.proportion_confint in the python library statsmodels," https://www.statsmodels.org/stable/generated/statsmodels.stats.proportion.proportion_confint.html, 2022, accessed November 2, 2022.

[34] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," in *International conference on computer aided verification*. Springer, 2017, pp. 97–117.

[35] S. M. Kay, *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.

[36] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.

[37] I. J. King and H. H. Huang, "Euler: Detecting network lateral movement via scalable temporal link prediction," in *Network and Distributed System Security Symposium*, 2022.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[39] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[40] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[41] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features." in *ICISSp*, 2017, pp. 253–262.

[42] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 656–672.

[43] K. Leino, Z. Wang, and M. Fredrikson, "Globally-robust neural networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6212–6222.

[44] B. Li, C. Chen, W. Wang, and L. Carin, "Certified adversarial robustness with additive noise," *Advances in neural information processing systems*, vol. 32, 2019.

[45] L. Li, X. Qi, T. Xie, and B. Li, "Sok: Certified robustness for deep neural networks," *arXiv preprint arXiv:2009.04131*, 2020.

[46] L. Li, M. Weber, X. Xu, L. Rimanic, B. Kailkhura, T. Xie, C. Zhang, and B. Li, "Tss: Transformation-specific smoothing for robustness certification," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 535–557.

[47] Z. Lyu, C.-Y. Ko, Z. Kong, N. Wong, D. Lin, and L. Daniel, "Fastened crown: Tightened neural network robustness certificates," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5037–5044.

[48] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[49] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018.

[50] W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun *et al.*, "Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs." in *IJCAI*, vol. 19, no. 7, 2019, pp. 4739–4745.

[51] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. [Online]. Available: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_03A-3_Mirsky_paper.pdf

[52] J. Mohapatra, C.-Y. Ko, L. Weng, P.-Y. Chen, S. Liu, and L. Daniel, "Hidden cost of randomized smoothing," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 4033–4041.

[53] M. Nasr, A. Bahramali, and A. Houmansadr, "Deepcorr: Strong flow correlation attacks on tor using deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1962–1976.

[54] ——, "Defeating {DNN-Based} traffic analysis systems in {Real-Time} with blind adversarial perturbations," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 2705–2722.

[55] M. Nasr, A. Houmansadr, and A. Mazumdar, "Compressive traffic analysis: A new paradigm for scalable traffic analysis," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 2053–2069.

[56] V. Rimmer, D. Preuveneers, M. Juárez, T. van Goethem, and W. Joosen, "Automated website fingerprinting through deep learning," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. [Online]. Available: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_03A-1_Rimmer_paper.pdf

[57] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

[58] Z. Shi, H. Zhang, K.-W. Chang, M. Huang, and C.-J. Hsieh, "Robustness verification for transformers," in *International Conference on Learning Representations*, 2019.

[59] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[60] P. Sirinam, M. Imani, M. Juarez, and M. Wright, "Deep fingerprinting: Undermining website fingerprinting defenses with deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1928–1943.

[61] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE symposium on security and privacy*. IEEE, 2010, pp. 305–316.

[62] M. Sun, Y. Su, S. Zhang, Y. Cao, Y. Liu, D. Pei, W. Wu, Y. Zhang, X. Liu, and J. Tang, "Ctf: Anomaly detection in high-dimensional time series with coarse-to-fine model transfer," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[63] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[64] V. Tjeng, K. Y. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=HyGIdiRqtm

[65] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, 2019.

[66] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter, "Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29 909–29 921, 2021.

[67] A. Warnecke, D. Arp, C. Wressnegger, and K. Rieck, "Evaluating explanation methods for deep learning in security," in *2020 IEEE european symposium on security and privacy (EuroS&P)*. IEEE, 2020, pp. 158–174.

[68] L. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon, "Towards fast computation of certified robustness for relu networks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5276–5285.

[69] E. Wong and Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5286–5295.

[70] E. Wong, F. Schmidt, J. H. Metzen, and J. Z. Kolter, "Scaling provable adversarial defenses," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[71] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh, "Automatic perturbation analysis for scalable certified robustness and beyond," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1129–1141, 2020.

[72] K. Xu, H. Zhang, S. Wang, Y. Wang, S. Jana, X. Lin, and C.-J. Hsieh, "Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers," in *International Conference on Learning Representation (ICLR)*, 2021.

[73] G. Yang, T. Duan, J. E. Hu, H. Salman, I. Razenshteyn, and J. Li, "Randomized smoothing of all shapes and sizes," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 693–10 705.

[74] L. Yang, W. Guo, Q. Hao, A. Ciptadi, A. Ahmadzadeh, X. Xing, and G. Wang, "Cade: Detecting and explaining concept drift samples for security applications," in *USENIX Security Symposium*, 2021.

[75] R. Yerukala and N. K. Boiroju, "Approximations to standard normal distribution function," *International Journal of Scientific & Engineering Research*, vol. 6, no. 4, pp. 515–518, 2015.

[76] F. Zhang, P. P. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE transactions on cybernetics*, vol. 46, no. 3, pp. 766–777, 2015.

[77] H. Zhang, H. Chen, C. Xiao, S. Gowal, R. Stanforth, B. Li, D. Boning, and C.-J. Hsieh, "Towards stable and efficient training of verifiably robust neural networks," in *International Conference on Learning Representations*, 2019.

[78] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," *Advances in neural information processing systems*, vol. 31, 2018.

[79] X. Zhang, N. Wang, H. Shen, S. Ji, X. Luo, and T. Wang, "Interpretable deep learning under fire," in *29th {USENIX} Security Symposium ({USENIX} Security 20)*, 2020.

Appendix A
Supplementary for Distribution Transformer

Here, we will provide some supplementaries for the Distribution Transformer in Section IV. Firstly, we introduce a condition for normalized noise distributions and an available probability distribution for normalized noise (§A-A). Secondly, we introduce some properties of the General Distribution Transformer (§A-B). Thirdly, we introduce some conditions for feature noise distributions and available probability distributions for feature noise (§A-C).

A. Condition and Available Probability Distribution for Normalized Noise

As stated in Section IV-A, the normalized noise distribution $\mathcal{D}_n$ can be isotropic classical distributions, such as isotropic Gaussian $\mathcal{N}\left(0, \sigma^2 I\right)$. Besides, $\mathcal{D}_n$ needs to satisfy a condition as follows:

a. **Symmetric distribution.** Normalized noise distributes symmetrically about the origin in all dimensions. It means that for $\forall \varepsilon_{n,i} \in \mathbb{R}$, we have $F_{n,i}\left(-\varepsilon_{n,i}\right) = 1 - F_{n,i}\left(\varepsilon_{n,i}\right)$.

The normalized noise distribution function of isotropic Gaussian $\mathcal{N}\left(0, \sigma^2 I\right)$ can be formulated as $F_{n,i}(\varepsilon_{n,i}) = \Phi\left(\frac{\varepsilon_{n,i}}{\sigma}\right)$. Considering $p_A \geq 0.5$, the ℓ_p robustness radius $r_{\mathcal{D}_n}\left(p_A\right)$ of isotropic Gaussian $\mathcal{N}\left(0, \sigma^2 I\right)$ can be formulated as $\|\boldsymbol{\delta}\|_2 \leq \sigma \Phi^{-1}\left(p_A\right)$ [15]. It should be noted that there are more available probability distributions beyond it [73], which demonstrates the extensibility of BARS.

TABLE X: Available special feature noise distributions.

Distribution	Original	$H_f^{(k)}\left(\varepsilon_{f,i}\right)$	$\left(H_f^{(k)}\right)^{-1}\left(p_i\right)$
Gaussian	$\Phi\left(z\right)$	$\Phi\left(\varepsilon_{f,i}\right)$	$\Phi^{-1}\left(p_i\right)$
Arctan	$\tan^{-1}\left(z\right)$	$\frac{1}{\pi}\tan^{-1}\left(\varepsilon_{f,i}\right)+\frac{1}{2}$	$\tan\left(\pi p_i-\frac{\pi}{2}\right)$
ISRU	$\frac{z}{\sqrt{1+z^2}}$	$\frac{1}{2}\cdot\frac{\varepsilon_{f,i}}{\sqrt{1+\varepsilon_{f,i}^2}}+\frac{1}{2}$	$\frac{2p_i-1}{\sqrt{1-\left(2p_i-1\right)^2}}$
Sigmoid	$\frac{1}{1+e^{-z}}$	$\frac{1}{1+e^{-\varepsilon_{f,i}}}$	$-\log\left(\frac{1}{p_i}-1\right)$

B. Property of General Distribution Transformer

General Distribution Transformer Ψ^G in Equation 12 has four properties as follows:

a. **Distribution transformation.** For $\forall\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n, \boldsymbol{\varepsilon}_f=\Psi^G\left(\boldsymbol{\varepsilon}_n\right)$, we have $\boldsymbol{\varepsilon}_f\sim\mathcal{D}_f$.

b. **Strictly monotone increasing in all dimensions.** Because $F_{n,i}$ and $F_{f,i}$, are both strictly monotone increasing, Ψ_i^G is strictly monotone increasing, which means that for $\forall\varepsilon_{n,i},\varepsilon'_{n,i}\in\mathbb{R}$ and $\varepsilon_{n,i}<\varepsilon'_{n,i}$, we have $\Psi_i^G\left(\varepsilon_{n,i}\right)<\Psi_i^G\left(\varepsilon'_{n,i}\right)$.

c. **Reversibility.** Because Ψ_i^G is strictly monotone increasing, Ψ^G is a reversible transformation.

d. **Odd in all dimensions.** Based on the condition a of F_n in Appendix A-A and the condition d of F_f in Appendix A-C, we have $\Psi_i^G\left(-\varepsilon_{n,i}\right)=-\Psi_i^G\left(\varepsilon_{n,i}\right)$.

C. Condition and Available Probability Distribution for Feature Noise

Special feature noise distribution function $H_f^{(k)}$ in Equation 13 needs to satisfy four conditions as follows:

a. **Bounded.** For $\forall\varepsilon_{f,i}\in\mathbb{R}$, we have $0\leq H_f^{(k)}\left(\varepsilon_{f,i}\right)\leq 1$, $\lim\limits_{\varepsilon_{f,i}\to-\infty}H_f^{(k)}\left(\varepsilon_{f,i}\right)=0$, $\lim\limits_{\varepsilon_{f,i}\to+\infty}H_f^{(k)}\left(\varepsilon_{f,i}\right)=1$.

b. **Strictly monotone increasing and reversibility.** For $\forall\varepsilon_{f,i},\varepsilon'_{f,i}\in\mathbb{R}$ and $\varepsilon_{f,i}<\varepsilon'_{f,i}$, we have $H_f^{(k)}\left(\varepsilon_{f,i}\right)<H_f^{(k)}\left(\varepsilon'_{f,i}\right)$. Therefore, $H_f^{(k)}$ should satisfy reversibility.

c. **Continuity.** For $\forall\varepsilon_{f,i}\in\mathbb{R}$, we have $\lim\limits_{\varepsilon'_{f,i}\to\varepsilon_{f,i}}H_f^{(k)}\left(\varepsilon'_{f,i}\right)=H_f^{(k)}\left(\varepsilon_{f,i}\right)$.

d. **Symmetric distribution.** Feature noise distributes symmetrically about the origin in all dimensions. It means that for $\forall\varepsilon_{f,i}\in\mathbb{R}$, we have $H_{f,i}\left(-\varepsilon_{f,i}\right)=1-H_{f,i}\left(\varepsilon_{f,i}\right)$. Then we have $\forall\varepsilon_{f,i}\in\mathbb{R}, F_{f,i}\left(-\varepsilon_{f,i}\right)=1-F_{f,i}\left(\varepsilon_{f,i}\right)$.

We also provide a series of available special distribution functions for feature noise in Table X. It should be noted that there are more available distribution functions beyond them, which demonstrates the extensibility of BARS. In Table X, there is no settable parameter, which simplifies the settings of distribution functions.

Appendix B
Theoretical Analysis of Robustness Guarantee with Distribution Transformer

Here, we first prove the robustness guarantee of the Linear Distribution Transformer in Theorem 1 and derive its robustness radii in Equation 10 and Equation 11 (§B-A). Then we prove the robustness guarantee of the General Distribution Transformer in Theorem 2 and derive its robustness radii in Equation 16 and Equation 17 (§B-B).

A. Proof for Linear Distribution Transformer

Proof of Theorem 1. Recall the smoothed traffic analyzer g in Equation 6. For ease of understanding, we firstly consider $f\left(\boldsymbol{x}+\boldsymbol{\varepsilon}_f\right)$. Recall the definition of the Linear Distribution Transformer Ψ^L in Equation 7. We have $\boldsymbol{\varepsilon}_f=\boldsymbol{w}\odot\boldsymbol{\varepsilon}_n$, where $\boldsymbol{\varepsilon}_f$ is the feature noise and $\boldsymbol{\varepsilon}_n$ is the normalized noise. Then we can get:

$$f\left(\boldsymbol{x}+\boldsymbol{\varepsilon}_f\right)=f\left(\boldsymbol{x}+\boldsymbol{w}\odot\boldsymbol{\varepsilon}_n\right). \tag{24}$$

Based on Equation 24, for a certain input sample $\boldsymbol{x}$, we can introduce a new virtual variable $\boldsymbol{x}_n$ and construct a new virtual function f_n:

$$f_n\left(\boldsymbol{x}_n\right)=f\left(\boldsymbol{x}+\boldsymbol{w}\odot\left(\boldsymbol{x}_n+\boldsymbol{\varepsilon}_n\right)\right), \tag{25}$$

where $\boldsymbol{x}$ is independent of $\boldsymbol{x}_n$ and can be be treated as a constant of $f_n\left(\boldsymbol{x}_n\right)$. Based on Equation 25, we can get a virtual smoothed classifier g_n of the virtual variable $\boldsymbol{x}_n$:

$$g_n\left(\boldsymbol{x}_n\right)=\arg\max_{c\in\mathcal{Y}}\mathbb{P}_{\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n}\left(f\left(\boldsymbol{x}+\boldsymbol{w}\odot\left(\boldsymbol{x}_n+\boldsymbol{\varepsilon}_n\right)\right)=c\right). \tag{26}$$

Based on $\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n$, we can make ℓ_p robustness certification for $g_n\left(\boldsymbol{x}_n\right)$. That means:

$$\forall\left\|\boldsymbol{\delta}_n\right\|_p\leq r_{\mathcal{D}_n}\left(p_A\right),\ g_n\left(\boldsymbol{x}_n\right)=g_n\left(\boldsymbol{x}_n+\boldsymbol{\delta}_n\right), \tag{27}$$

where $\boldsymbol{\delta}_n$ is the perturbation for $\boldsymbol{x}_n$ in $g_n\left(\boldsymbol{x}_n\right)$, and $r_{\mathcal{D}_n}(p_A)$ is the ℓ_p robustness radius of randomized smoothing [15], [73]. We substitute $\boldsymbol{x}_n=\mathbf{0}$ into Equation 27 and get:

$$\forall\left\|\boldsymbol{\delta}_n\right\|_p\leq r_{\mathcal{D}_n}\left(p_A\right),\ g_n\left(\mathbf{0}\right)=g_n\left(\mathbf{0}+\boldsymbol{\delta}_n\right). \tag{28}$$

Observing the left side of Equation 28, we have:

$$\begin{aligned}g_n\left(\mathbf{0}\right)&=\arg\max_{c\in\mathcal{Y}}\mathbb{P}_{\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n}\left(f\left(\boldsymbol{x}+\boldsymbol{w}\odot\boldsymbol{\varepsilon}_n\right)=c\right)\\&=g\left(\boldsymbol{x}\right),\end{aligned} \tag{29}$$

where g is the smoothed traffic analyzer with the Distribution Transformer defined in Equation 6.

Observing the right side of Equation 28, we have:

$$\begin{aligned}g_n\left(\mathbf{0}+\boldsymbol{\delta}_n\right)&=\arg\max_{c\in\mathcal{Y}}\mathbb{P}_{\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n}\left(f\left(\boldsymbol{x}+\boldsymbol{w}\odot\left(\boldsymbol{\delta}_n+\boldsymbol{\varepsilon}_n\right)\right)=c\right)\\&=\arg\max_{c\in\mathcal{Y}}\mathbb{P}_{\boldsymbol{\varepsilon}_n\sim\mathcal{D}_n}\left(f\left(\boldsymbol{x}+\boldsymbol{w}\odot\boldsymbol{\delta}_n+\boldsymbol{w}\odot\boldsymbol{\varepsilon}_n\right)=c\right)\\&=g\left(\boldsymbol{x}+\boldsymbol{w}\odot\boldsymbol{\delta}_n\right).\end{aligned} \tag{30}$$

Combining Equation 28, Equation 29, Equation 30, we get:

$$\forall\left\|\boldsymbol{\delta}_n\right\|_p\leq r_{\mathcal{D}_n}\left(p_A\right),\ g\left(\boldsymbol{x}\right)=g\left(\boldsymbol{x}+\boldsymbol{w}\odot\boldsymbol{\delta}_n\right). \tag{31}$$

We define the perturbation $\boldsymbol{\delta}$ for $\boldsymbol{x}$ in $g\left(\boldsymbol{x}\right)$ as $\boldsymbol{\delta}=\boldsymbol{w}\odot\boldsymbol{\delta}_n$. Besides, we define $\boldsymbol{w}'$ as $w'_i=\frac{1}{w_i}\ i=1,2\ldots d$. Then Equation 31 can be converted into:

$$\forall\left\|\boldsymbol{w}'\odot\boldsymbol{\delta}\right\|_p\leq r_{\mathcal{D}_n}\left(p_A\right),\ g\left(\boldsymbol{x}\right)=g\left(\boldsymbol{x}+\boldsymbol{\delta}\right). \tag{32}$$

Based on Equation 32, we can get the the robustness region Ω of $g\left(\boldsymbol{x}\right)$:

$$\Omega=\left\{\boldsymbol{x}+\boldsymbol{\delta}\,\middle|\,\left\|\boldsymbol{w}'\odot\boldsymbol{\delta}\right\|_p\leq r_{\mathcal{D}_n}\left(p_A\right)\right\}. \tag{33}$$

Deriving Equation 10 and Equation 11. According to Equation 33, given a perturbation $\boldsymbol{\delta}$ satisfying $\boldsymbol{x}+\boldsymbol{\delta}\in\Omega$, we have:

$$\left|w'_i\cdot\delta_i\right|\leq\left(\sum_{j=1}^{d}\left|w'_j\cdot\delta_j\right|^p\right)^{\frac{1}{p}}\leq r_{\mathcal{D}_n}\left(p_A\right)\quad i=1,2\ldots d, \tag{34}$$

where the left equality holds when $\forall j\neq i,\left|\delta_j\right|=0$ is satisfied. The right equality holds when $\left|w'_i\cdot\delta_i\right|$ holds its maximum value in the robustness region. Due to $t \geq 0$, $w^S \geq 0$, $w_i^I \geq 0$, we have $w_i' \geq 0$. Then Equation 34 can be converted into:

$$|\delta_i| \leq \frac{1}{w_i'} \cdot r_{\mathcal{D}_n}(p_A) \quad i = 1, 2 \ldots d, \tag{35}$$

where the equality holds when $|\delta_i|$ holds its maximum value in the robustness region. Substituting $w_i = \frac{1}{w_i'}$ $i = 1, 2 \ldots d$ into Equation 35, we have:

$$|\delta_i| \leq w_i \cdot r_{\mathcal{D}_n}(p_A) \quad i = 1, 2 \ldots d. \tag{36}$$

Recall the definition of the dimension-wise robustness radius vector $\boldsymbol{r}$ in Equation 4. We have:

$$r_i = w_i \cdot r_{\mathcal{D}_n}(p_A) \quad i = 1, 2 \ldots d. \tag{37}$$

Recall the definition of the dimension-heterogeneous robustness radius R in Equation 5. We have:

$$R = \frac{1}{d} \sum_{i=1}^{d} w_i \cdot r_{\mathcal{D}_n}(p_A). \tag{38}$$

B. Proof for General Distribution Transformer

Proof of Theorem 2. Recall the smoothed traffic analyzer g in Equation 6. For ease of understanding, we firstly consider $f(\boldsymbol{x} + \boldsymbol{\varepsilon}_f)$. With the General Distribution Transformer Ψ^G, it can be converted into:

$$f(\boldsymbol{x} + \boldsymbol{\varepsilon}_f) = f\left(\boldsymbol{x} + \Psi^G(\boldsymbol{\varepsilon}_n)\right), \tag{39}$$

where $\boldsymbol{\varepsilon}_f$ is the feature noise and $\boldsymbol{\varepsilon}_n$ is the normalized noise. Based on Equation 39, for a certain input sample $\boldsymbol{x}$, we can introduce a new virtual variable $\boldsymbol{x}_n$ and construct a new virtual function f_n:

$$f_n(\boldsymbol{x}_n) = f\left(\boldsymbol{x} + \Psi^G(\boldsymbol{x}_n + \boldsymbol{\varepsilon}_n)\right), \tag{40}$$

where $\boldsymbol{x}$ is independent of $\boldsymbol{x}_n$ and can be treated as a constant of $f_n(\boldsymbol{x}_n)$. Based on Equation 40, we can get a virtual smoothed classifier g_n of the virtual variable $\boldsymbol{x}_n$:

$$g_n(\boldsymbol{x}_n) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n}\left(f\left(\boldsymbol{x} + \Psi^G(\boldsymbol{x}_n + \boldsymbol{\varepsilon}_n)\right) = c\right). \tag{41}$$

Based on $\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n$, we can make ℓ_p robustness certification for $g_n(\boldsymbol{x}_n)$. That means:

$$\forall \|\boldsymbol{\delta}_n\|_p \leq r_{\mathcal{D}_n}(p_A), \ g_n(\boldsymbol{x}_n) = g_n(\boldsymbol{x}_n + \boldsymbol{\delta}_n), \tag{42}$$

where $\boldsymbol{\delta}_n$ is the perturbation for $\boldsymbol{x}_n$ in $g_n(\boldsymbol{x}_n)$, $r_{\mathcal{D}_n}(p_A)$ is the ℓ_p robustness radius of randomized smoothing [15], [73]. We substitute $\boldsymbol{x}_n = \mathbf{0}$ into Equation 42 and get:

$$\forall \|\boldsymbol{\delta}_n\|_p \leq r_{\mathcal{D}_n}(p_A), \ g_n(\mathbf{0}) = g_n(\mathbf{0} + \boldsymbol{\delta}_n). \tag{43}$$

Observing the left side of Equation 43, we have:

$$g_n(\mathbf{0}) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n}\left(f\left(\boldsymbol{x} + \Psi^G(\boldsymbol{\varepsilon}_n)\right) = c\right). \tag{44}$$

Observing the right side of Equation 43, we have:

$$g_n(\mathbf{0} + \boldsymbol{\delta}_n) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n}\left(f\left(\boldsymbol{x} + \Psi^G(\boldsymbol{\delta}_n + \boldsymbol{\varepsilon}_n)\right) = c\right). \tag{45}$$

We define the perturbation $\boldsymbol{\delta}$ for $\boldsymbol{x}$ in $g(\boldsymbol{x})$ as $\boldsymbol{\delta} = \Psi^G(\boldsymbol{\delta}_n)$. Then Equation 45 can be converted into:

$$\begin{aligned} &g_n(\mathbf{0} + \boldsymbol{\delta}_n) \\ &= \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\varepsilon}_n \sim \mathcal{D}_n}\left(f\left(\boldsymbol{x} + \boldsymbol{\delta} + \Psi^G\left(\left(\Psi^G\right)^{-1}(\boldsymbol{\delta}) + \boldsymbol{\varepsilon}_n\right) - \boldsymbol{\delta}\right) = c\right). \end{aligned} \tag{46}$$

To obtain the robustness guarantee with the General Distribution Transformer, we need to extend the smoothed traffic analyzer $g(\boldsymbol{x})$ to $g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f)$ by regarding feature noise $\boldsymbol{\varepsilon}_f$ as a variable. Then Equation 44 can be converted into:

$$\begin{aligned} &g_n(\mathbf{0}) = g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f), \\ &\varepsilon_{f,i} = \Psi_i^G(\varepsilon_{n,i}) \ i = 1, 2 \ldots d. \end{aligned} \tag{47}$$

Equation 46 can be converted into:

$$\begin{aligned} &g_n(\mathbf{0} + \boldsymbol{\delta}_n) = g\big(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{\varepsilon}_f'\big), \\ &\varepsilon_{f,i}' = \Psi_i^G\left(\left(\Psi_i^G\right)^{-1}(\delta_i) + \varepsilon_{n,i}\right) - \delta_i \ i = 1, 2 \ldots d. \end{aligned} \tag{48}$$

Combining Equation 43, Equation 47, Equation 48, we get:

$$\begin{aligned} &\forall \left\|\left(\Psi^G\right)^{-1}(\boldsymbol{\delta})\right\|_p \leq r_{\mathcal{D}_n}(p_A), \ g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f) = g\big(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{\varepsilon}_f'\big), \\ &\varepsilon_{f,i} = \Psi_i^G(\varepsilon_{n,i}) \ i = 1, 2 \ldots d, \\ &\varepsilon_{f,i}' = \Psi_i^G\left(\left(\Psi_i^G\right)^{-1}(\delta_i) + \varepsilon_{n,i}\right) - \delta_i \ i = 1, 2 \ldots d. \end{aligned} \tag{49}$$

Based on Equation 49, we can get the robustness region Ω of $g(\boldsymbol{x}, \boldsymbol{\varepsilon}_f)$:

$$\Omega = \left\{\boldsymbol{x} + \boldsymbol{\delta} \,\middle|\, \left\|\left(\Psi^G\right)^{-1}(\boldsymbol{\delta})\right\|_p \leq r_{\mathcal{D}_n}(p_A)\right\}. \tag{50}$$

Deriving Equation 16 and Equation 17. According to Equation 50, given a perturbation $\boldsymbol{\delta}$ satisfying $\boldsymbol{x} + \boldsymbol{\delta} \in \Omega$, we have:

$$\left|\left(\Psi_i^G\right)^{-1}(\delta_i)\right| \leq \left(\sum_{j=1}^{d}\left|\left(\Psi_j^G\right)^{-1}(\delta_j)\right|^p\right)^{\frac{1}{p}} \leq r_{\mathcal{D}_n}(p_A) \ i = 1, 2 \ldots d, \tag{51}$$

where Ψ_i^G is the sub-formula of Ψ^G in the i^{th} dimension. The left equality holds when $\forall j \neq i, \left|\left(\Psi_j^G\right)^{-1}(\delta_j)\right| = 0$ is satisfied. The right equality holds when $\left|\left(\Psi_i^G\right)^{-1}(\delta_i)\right|$ holds its maximum value in the robustness region.

Based on the property b and the property d of Ψ^G in Section IV-C (Appendix A-B), we have:

$$\left|\left(\Psi_i^G\right)^{-1}(\delta_i)\right| = \left(\Psi_i^G\right)^{-1}(|\delta_i|) \quad i = 1, 2 \ldots d. \tag{52}$$

We substitute Equation 52 into Equation 51, and get:

$$\left(\Psi_i^G\right)^{-1}(|\delta_i|) \leq r_{\mathcal{D}_n}(p_A) \ i = 1, 2 \ldots d. \tag{53}$$

Based on the property b of Ψ^G in Section IV-C (Appendix A-B), Equation 53 can be converted into:

$$\Psi_i^G\left(\left(\Psi_i^G\right)^{-1}(|\delta_i|)\right) \leq \Psi_i^G(r_{\mathcal{D}_n}(p_A)) \quad i = 1, 2 \ldots d. \tag{54}$$

Equation 54 can be simplified to:

$$|\delta_i| \leq \Psi_i^G(r_{\mathcal{D}_n}(p_A)) \quad i = 1, 2 \ldots d, \tag{55}$$

where the equality holds when $|\delta_i|$ holds its maximum value in the robustness region.

Recall the definition of the dimension-wise robustness radius vector $\boldsymbol{r}$ in Equation 4. We have:

$$r_i = \Psi_i^G(r_{\mathcal{D}_n}(p_A)) \ i = 1, 2 \ldots d. \tag{56}$$

Recall the definition of the dimension-heterogeneous robustness radius R in Equation 5. We have:

$$R = \frac{1}{d} \sum_{i=1}^{d} \Psi_i^G(r_{\mathcal{D}_n}(p_A)). \tag{57}$$