Benchmark: Formal Verification of Semantic Segmentation Neural Networks

Neelanjana Pal[1], Seojin Lee[1], and Taylor T. Johnson[1]

[1]Institute for Software Integration and Systems, Vanderbilt University, Nashville, TN, USA {neelanjana.pal,seojin.lee,taylor.johnson}@vanderbilt.edu

Abstract. Formal verification utilizes a rigorous approach to ensure the absence of critical errors and validate models against predefined properties. While significant progress has been made in verification methods for various deep neural networks (DNNs), such as feed-forward neural networks (FFNNs) and convolutional neural networks (CNNs), the application of these techniques to semantic segmentation remains largely unexplored. Semantic segmentation networks are vital in computer vision applications, where they assign semantic labels to individual pixels within an image. Given their deployment in safety-critical domains, ensuring the correctness of these networks becomes paramount. This paper presents a comprehensive benchmark study on applying formal verification techniques to semantic segmentation networks. We explore a diverse set of state-of-the-art semantic segmentation datasets and generate neural network models, including fully-convolutional networks and encoder-decoder architectures. Our investigation encompasses a wide range of verification properties, focusing on the robustness of these models against bounded adversarial vulnerabilities. To evaluate the networks' performance, we employ set-based reachability algorithms to calculate the output reachable set(s) and some state-of-the-art performance measures for a comparative study among the networks. This benchmark paper aims to provide the formal verification community with several semantic segmentation networks and their robustness specifications for future use cases in different neural network verification competitions.

Keywords: Semantic Segmentation · Adversarial Attack · Benchmarking · Reachability · Robustness · VNN-Comp · .

1 Introduction

The Significant Role of Semantic Segmentation. Over the past three decades, image segmentation [34] has been one of the most challenging problems in computer vision. In contrast to tasks like image classification or object recognition, image segmentation operates differently, as it does not rely on prior knowledge of visual concepts or objects within the image. Instead, it assigns a specific category label to every individual pixel in the image. This approach allows the model to accurately delineate distinct regions or objects present in the image, effectively dividing it into meaningful segments. The model creates

a comprehensive and detailed representation of the image's content by associating each pixel with its corresponding category label. This capability to provide pixel-level category information has significant real-world applications [36], such as self-driving vehicles [23, 45], pedestrian detection [5, 12], defect detection [35], therapy planning [15, 46], and computer-aided diagnosis [52, 53]. The segmentation task empowers intelligent systems to grasp spatial positions and make critical judgments by offering detailed semantic information at the pixel level, setting it apart from other common computer vision tasks.

(a) The Original image (b) Semantically Segmented Image

Fig. 1: An example of semantic segmentation vision tasks from CamVid dataset [7].

Deep Neural Networks (DNN) and Adversarial Attacks. Research has shown that even well-trained neural networks (NNs) are vulnerable to minor input modifications (i.e., adversarial attacks) that can cause significant changes in the output [27]. Similar to image classification neural networks, semantic segmentation networks (SSNs) are also known to be vulnerable to adversarial perturbations [50]. While deep neural network (DNN) verification is evolving as a well-established research area with numerous tools and techniques proposed to ensure the safety and robustness specifications of DNNs [24, 48] and neural network-controlled systems [16, 18, 39, 43] most state-of-the-art verification techniques for robustness validation in DNNs primarily focus on variations of classification tasks, often related to images [1, 4, 8, 11, 21, 26, 30, 32, 37, 38, 49, 51]. In recent years, verification of segmentation networks has also gained immense focus from researchers all over [3, 19, 42].

Neural Network Verification Competitions. The proliferation of neural networks (NNs) in safety-critical applications has brought attention to their susceptibility to adversarial examples [33], where even minor input perturbations can significantly alter their outputs. Such perturbations, whether occurring randomly or due to malicious intent, emphasize the crucial need to rigorously analyze the robustness of deep learning systems before deploying them in safety-critical domains. Consequently, numerous methods and software tools

[10,13,17,20] have been developed for this purpose. However, the increasing number and specialization of these tools have made it challenging for practitioners to choose the most suitable one for their needs.

In response to this dilemma, in 2020, a friendly International Competition on Verification of Neural Networks (VNN-Comp 2020) [2,6,28] was conducted to address the issue and allow researchers to compare their neural network verifiers across a wide range of benchmarks. Originally designed as a friendly competition with minimal standardization, the event evolved to introduce more standardization and automation. The goal was to ensure a fair comparison among verifiers on cost-equivalent hardware, utilizing standardized formats for properties and networks. This evolution aimed to facilitate informed decision-making by researchers and practitioners when selecting verification tools for their specific requirements. The VNN-Comp celebrates its 4th iteration this year and successfully presented the results at the Computer Aided Verification 2023 (CAV) conference.

Work Presented In This Paper. Despite the growing interest and competition in robustness verification, there remains a lack of appropriate benchmarks for evaluating different verifiers on semantic segmentation tasks. This research addresses this gap by introducing segmentation networks on two widely used datasets: MNIST [22], M2NIST (which is a multi-digit variant of MNIST suitable for segmentation evaluation) . Additionally, we define the specific properties that need to be verified for these networks.

An essential aspect of our work is its potential utility for the VNN-Comp's upcoming iterations. By providing these well-defined benchmarks for semantic segmentation, we hope to contribute to advancing and standardizing robustness verification techniques in this domain.

Contributions. In summary, this paper makes several key contributions:

1. We introduce NNs designed specifically for two different semantic segmentation datasets, providing a comprehensive benchmark dataset for evaluating their performance.
2. In this benchmark work, we not only showcase fully-convolutional NNs but also include encoder-decoder architectures, offering a diverse set of models to assess their effectiveness in different scenarios.
3. To assess the robustness of these networks against adversarial attacks, we define specific properties and constraints and represent them in vnnlib files, adhering to the guidelines set by VNN-Comp.
4. Furthermore, we present sample verification results, obtained using set-based reachability algorithms and performance measures. These results demonstrate the networks' resilience and provide insights into their performance under different conditions, aiding researchers and practitioners in making informed decisions.

Outline. The paper is structured as follows: Section 2 introduces the benchmark design considerations for this proposal, while Section 3 presents an example

verification task for the MNIST networks against an adversarial attack. Finally, Section 4 provides a summary of the main proposal, discusses its implications, and outlines potential avenues for future research.

2 Benchmark Design

In the following, we describe our benchmark: (i) the overall motivations and philosophy; (ii) the Datasets and their creation; (iii) the networks proposed; (iv) the unknown-bounded adversarial attack; (v) the metrics used for evaluation; and (vi) robustness property specification.

2.1 Philosophy

The motivation behind benchmarking formal verification techniques for semantic segmentation networks arises from the growing significance of deploying these networks in safety-critical applications. By establishing standardized benchmarks and datasets, researchers and practitioners can assess the strengths and limitations of various verification techniques. This enables them to make well-informed decisions when selecting the most appropriate verification methods, considering factors like accuracy, computational overhead, and scalability.

Benchmarking formal verification techniques drives innovation and encourages the development of more reliable and secure deep learning models. It paves the way for integrating formal methods into the training and deployment pipeline, instilling greater confidence in the safety and robustness of semantic segmentation networks for critical applications.

2.2 Datasets

The MNIST and M2NIST datasets provide a solid starting point for conducting image-segmentation benchmarking. In contrast to real-world images, especially those captured from autonomous vehicles, these datasets feature well-isolated digits positioned at the center of the images. Consequently, the task of segmentation is comparatively straightforward. The digits, which are the focal points of interest, exhibit distinct clarity and encounter minimal clutter or occlusion. A significant advantage of the MNIST and M2NIST datasets lies in their provision of well-defined ground-truth annotations. This feature greatly simplifies the accurate assessment of segmentation algorithms, enabling meticulous evaluation. Furthermore, these datasets often serve as valuable tools for conveying and elucidating image segmentation's core principles.

MNIST Dataset The MNIST [22] dataset is a well-known dataset used for training and testing machine learning models, particularly for image classification tasks. It consists of handwritten digit images, where each image is a grayscale image of size 28x28 pixels and corresponding ground-truth-labeled masks representing random digit numbers ranging from 0 to 9. To facilitate our experiments, we divided the dataset into two sets: 50,000 images for training and 10,000 images for testing. Fig. 2 displays sample images from the MNIST dataset.

Fig. 2: Sample images from MNIST dataset

M2NIST Dataset The M2NIST dataset comprises images with dimensions of (64, 84, 1) and corresponding ground-truth-labeled masks depicting multiple (up to three) random digits ranging from 0 to 9. These digits are arranged in a manner that they do not overlap with each other within the images. For our experiments, we divided the original dataset into two sets: 50,000 samples for training and 10,000 for testing. Figure 3 displays sample images from the M2NIST dataset.

Fig. 3: Sample images from M2NIST dataset

2.3 Neural Network Models

MNIST Dataset. For the MNIST dataset, we utilize two pre-trained networks from [42], and we train a third network with 16 layers. The networks consist of an image input layer with the input size of (28, 28, 1) followed by two-dimensional convolution layers, ReLU layers and average-pooling layers.

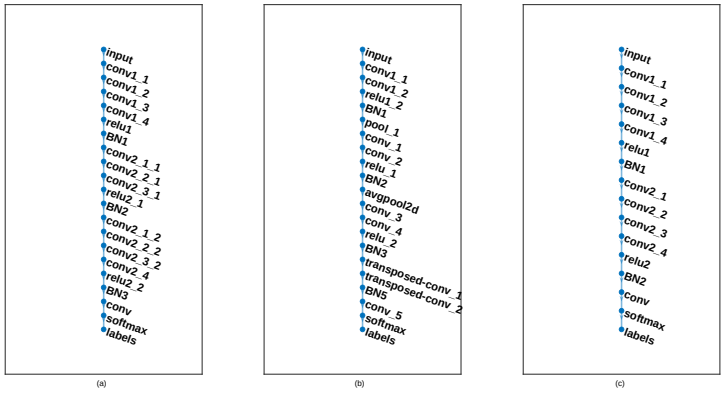

Fig. 4: Benchmark for MNIST Networks

M2NIST Dataset. For the M2NIST Dataset, we propose the three pre-trained networks used in [42] and two newly trained networks, as shown in Fig. 5. The input image size for these networks is changed to (64, 84, 1).

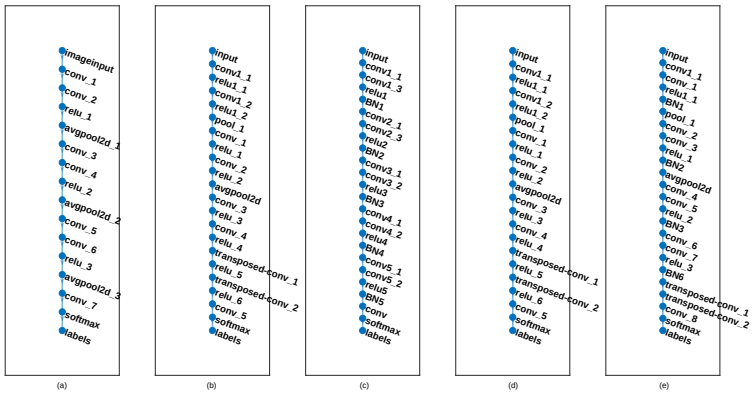

Fig. 5: Benchmark for M2NIST Networks

Table 1: Performances of different networks used for MNIST and M2NIST datasets

$Network_{MNIST}$	$Accuracy_{global}(\%)$	$Accuracy_{mean}(\%)$	IoU_{mean}	$IoU_{weighted}$
mnist_21_iou83	96.88	93.70	0.8335	0.9427
mnist_avg_21	97.28	96.20	0.8675	0.9490
mnist_16	96.93	92.67	0.8376	0.9430
$Network_{M2NIST}$	$Accuracy_{global}(\%)$	$Accuracy_{mean}(\%)$	IoU_{mean}	$IoU_{weighted}$
m2nist_avg_iou62	96.61	88.30	0.6210	0.9464
m2nist_avg_iou75	98.03	97.60	0.7502	0.9660
m2nist_iou72_24	97.86	96.27	0.7271	0.9635
m2nist_avg_22	97.97	98.30	0.7495	0.9650
m2nist_avg_24	97.07	97.86	0.8321	0.9466

The performance measures of each of the proposed networks are shown in Table. 1.

2.4 Segmentation in the Context of Proposed Datasets

The segmentation task in the MNIST and M2NIST datasets involves the process of precisely delineating and identifying individual digits within the given images. The goal is to assign a distinct label to each pixel or region that corresponds to a specific digit. This segmentation is vital for isolating and distinguishing the different digits present in the image, enabling accurate digit recognition and analysis.

MNIST Dataset. In the MNIST dataset, each image depicts a single handwritten digit (0-9). The segmentation task involves precisely outlining the boundaries of the digit, distinguishing it from the background. This process aims to identify the exact spatial extent of the digit, ensuring that every pixel belonging to the digit is correctly labeled while excluding pixels from the surrounding background.

M2NIST Dataset. The M2NIST dataset introduces a slightly more complex scenario. It consists of images containing two or three handwritten digits placed in non-overlapping arrangements. The segmentation task in M2NIST entails accurately segmenting each digit within the image, ensuring that the segmentation boundaries do not cross over into neighboring digits. This task becomes especially challenging when digits are in close proximity, as segmentation algorithms must correctly identify the boundaries between adjacent digits.

The segmentation task in both datasets essentially involves creating pixel-wise masks that outline the boundaries of individual digits. These masks indicate which pixels belong to each digit and which pixels constitute the background. The successful execution of the segmentation task is crucial for subsequent digit recognition, as the isolated digits can then be analyzed, classified, and identified accurately.

2.5 Adversarial Attacks

Inspired by the paper [42], we consider an unknown-bounded adversarial attack (UBAA) on input images in our study. The coefficient vector ϵ, representing the attack strength, is bounded by lower and upper bounds, denoted as $[\underline{\epsilon}, \overline{\epsilon}]$, with each ϵ_i satisfying $\underline{\epsilon_i} \leq \epsilon_i \leq \overline{\epsilon_i}$. We apply this attack concept to images, where pixel values range from 0 to 255, and consider the attack's impact on either a single pixel or multiple pixels within the image. By applying this attack, a set of images is generated, each having variations in pixel values within one or multiple locations, limited by the bounds $[\underline{\epsilon}, \overline{\epsilon}]$.

To illustrate this, we represent an adversarial image x^{adv} as follows:

$$x^{adv} = x^{org} + \Sigma_{i=1}^{n} \epsilon_i \cdot x_i^{attack} \tag{1}$$

where x^{org} and x^{adv} are the original and adversarial images, respectively. The variable n denotes the total number of pixels in the image, and ϵ_i represents the attack coefficient for the pixel at position i.

Within this benchmark study, we subject each image, x, to an UBAA across the test datasets. Here we darken a pixel $x(i, j)$ by 1 unit if its value exceeds a specified threshold, denoted as d. In mathematical terms, the adversarial darkening attack on an image x can be described as follows:

$$x^{adv} = x + \epsilon \cdot x^{noise},\ 1 - \Delta_\epsilon \leq \epsilon \leq 1,$$
$$x^{noise}(i, j) = -1,\ \text{if } x(i, j) > d,\ \text{otherwise } x^{noise}(i, j) = 0.$$

For $\epsilon = 1$, we darken all pixels by 1 unit whose values exceed the threshold d (set to 150 for this work and $\epsilon = 1$), resulting in $x^{adv}(i, j) = x(i, j) - 1$. The size of the input set affected by the attack is determined by Δ_ϵ. A larger value of Δ_ϵ corresponds to a larger input set after applying the attack.

By varying the values of ϵ and d, we generate different robustness properties to verify.

2.6 Evaluation Metrics

For evaluation purposes, we used the traditional concept of Intersection-over-Union (IoU) for both segmentation model performances and model robustness checking. Following [42], we also used the concept of robustness value (RV) and robustness sensitivity (RS).

Robustness Value (RV). An SSN's Robustness Value (RV) characterizes its resilience against an adversarial attack. Specifically, for an unknown bounded adversarial attack applied to an input image, the RV is defined as follows:

$$RV = \frac{N_{robust}}{N_{pixels}} \times 100\%, \tag{2}$$

where N_{robust} is the total number of robust pixels [1] under the attack, and $N_{pixels} = h \cdot w$ is the total number of input image pixels.

Robustness Sensitivity (RS). The Robustness Sensitivity (RS) quantifies the network's susceptibility under the adversarial attack, revealing the average number of pixels in the segmentation output image that are influenced (either becoming non-robust or unknown) when a single pixel in the input image is attacked. The robustness sensitivity of an SSN corresponding to an unknown bounded adversarial attack applied to an input image is defined as

$$RS = \frac{N_{nonrobust} + N_{unknown}}{N_{attackedpixels}}, \tag{3}$$

where $N_{nonrobust}$ is the total number of non-robust pixels under the attack, $N_{unknown}$ is the total number of pixels whose robustness is unknown (i.e., the verifier can not guarantee on the robustness; it may or may not be robust), and $N_{attackedpixels}$ is the total number of attacked pixels of the input image.

Robust Intersection-over-Union (IoU). The robust IoU (R_{IoU}) concept shares similarities with the traditional IoU, a fundamental metric used to evaluate accuracy during the training of semantic segmentation networks (SSNs). The robust IoU of a semantic segmentation network (SSN) when subjected to an unknown-bounded adversarial attack on an input image is calculated as the average IoU of all labels that remain robust under the attack.

Consider a segmentation ground-truth image denoted as x and the verified segmentation image under the adversarial attack as y. Then the IoU (IoU_p) for the p^{th} label in the label images x and y is computed as the intersection of the label images divided by their union for the i^{th} label. , then the R_{IoU} of the SSN is computed by:

$$R_{IoU} = \frac{\Sigma_{p=1}^{L} IoU_p}{L}. \tag{4}$$

[1] In this context, "Robust pixels" refer to those pixels that maintain their correct classification even in the presence of an adversarial attack.

In our context, we leverage the robust IoU concept in conjunction with the robustness value and robustness sensitivity as core metrics to assess the robustness of an SSN when subjected to adversarial attacks within the verification framework. Instead of solely measuring accuracy, these metrics comprehensively evaluate the network's resilience against such attacks.

2.7 Robustness Property Specification

In semantic segmentation examples, while each pixel is assigned a class label, individual pixels do not determine the object classification. Instead, a cluster of pixels with the same class collectively contributes to the final object decision. As a result, the robustness property defined for classification models is not directly applicable to segmentation models. Therefore, we focus on evaluating the Intersection over Union (IoU) measures of the segmentation output image w.r.t its original counterpart.

To characterize the robustness properties of a specific SSN for a given input image, we define its corresponding Robustness Value (RV) and Robustness Sensitivity (RS) within a specified range. This range represents the maximum allowable deviation the SSN is allowed to exhibit to be within a safe region. Consequently, for an adversarial input image set denoted as X^{adv} and its output segmentation image set as Y^{adv}, the robustness property for RV is defined as follows:

$$RV_{min} \leq RV_{org} \leq RV_{max} \tag{5}$$

where $[RV_{min}, RV_{max}]$ is the permissible bounds for the RV and RV_{org} is the actual RV for the unperturbed image.

Similarly, we can get the robustness property for the RS of the same example as:

$$RS_{min} \leq RS_{org} \leq RS_{max} \tag{6}$$

where $[RS_{min}, RS_{max}]$ is the permissible bounds for the RS and RS_{org} is the actual RS for the unperturbed image.

The IoU robustness property can be given by the equation:

$$R_{IoU_{min}} \leq R_{IoU_{org}} \leq R_{IoU_{max}} \tag{7}$$

where $[R_{IoU_{min}}, R_{IoU_{max}}]$ is the permissible bounds for the IoU and $R_{IoU_{org}}$ is the actual IoU for the unperturbed image.

2.8 Verification Property Specifications in vnnlib Files: Illustrated with an Example

vnnlib file format. Following the competition protocol, we propose the robustness specification in a vnnlib file [9]. A vnnlib file is a standard format for representing neural network verification problems. It provides details about the neural network, input constraints, and properties to be verified. The vnnlib file format is widely adopted in formal verification for neural networks [2,14,28,31].

The verification specification in a vnnlib file involves defining properties using a specific syntax. The structure of a vnnlib file typically includes the following components:

1. **Input Constraints:** This section defines the input bounds or constraints for the neural network.
2. **Output Behavior Specification:** This section contains the expected output or behavior of the neural network for the specified input constraints.
3. **Property Specification:** This section specifies the neural network properties to be verified. These properties include safety and robustness verification properties.

Example. To illustrate this format, we consider the example in Fig. 6.

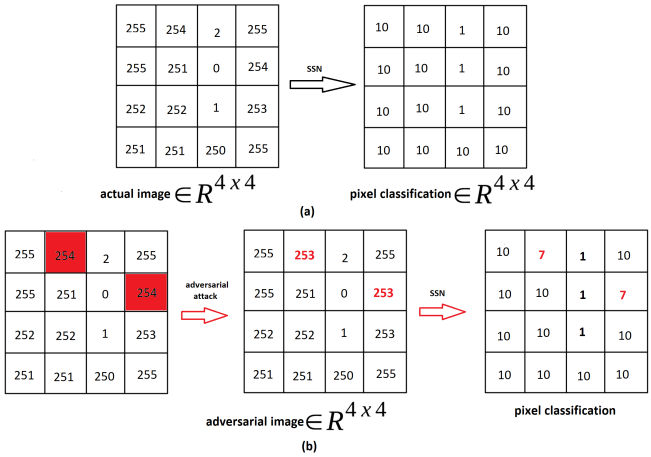

Fig. 6: The robustness verification specification for a Semantic Segmentation Network (SSN) is illustrated as follows: (a) Example image and its corresponding pixel classification. (b) Location of the adversarial attack (red), the pixel darkened by 1 (original value 254), and the resulting pixel classification after the adversarial attack.

In this example, we focus on an input image represented as a 4×4 2-dimensional array with pixel values from 0 to 255. The output of the Semantic Segmentation Network (SSN) pixel classification layer is also a 4×4 2-dimensional array, assigning classes to each pixel. For this image, we highlight the digit '1', with the first three rows in column 3 of the output classified as '1' and the remaining rows labeled as background (represented by '10').

Next, we explore an Unbounded Adversarial Attack (UBAA) on the image, targeting pixels with a value of 254, reducing them to 253. To ensure a bounded attack, we set the upper bound of the attacked image as the original image itself [Fig. 6 (b) left], and the lower bound as the image with darkened pixels [Fig. 6 (b) middle].

To specify the input properties in a vnnlib file, we first flatten the input image column-wise and then define the upper and lower bounds for each pixel representing the attack. In the case of the provided image example, the input properties in a vnnlib file should be structured as depicted in List. 1. This allows for a comprehensive representation of the image and its bounds, facilitating the verification process.

List 1 Input Constraints

```
; Verification Property Specification
(declare-const X_0 Real)
(declare-const X_1 Real)
(declare-const X_2 Real)
(declare-const X_3 Real)
(declare-const X_4 Real)
(declare-const X_5 Real)
(declare-const X_6 Real)
(declare-const X_7 Real)
(declare-const X_8 Real)
(declare-const X_9 Real)
(declare-const X_10 Real)
(declare-const X_11 Real)
(declare-const X_12 Real)
(declare-const X_13 Real)
(declare-const X_14 Real)
(declare-const X_15 Real)
; Unscaled Input 0 : (255, 255)
(assert (<= X_0 255))
(assert (>= X_0 255))

; Unscaled Input 1 : (255, 255)
(assert (<= X_1 255))
(assert (>= X_1 255))

; Unscaled Input 2 : (252, 252)
(assert (<= X_2 252))
(assert (>= X_2 252))

; Unscaled Input 3 : (251, 251)
(assert (<= X_3 251))
(assert (>= X_3 251))

; Unscaled Input 4 : (253, 254)
(assert (<= X_4 254))
(assert (>= X_4 253))

; Unscaled Input 5 : (251, 251)
(assert (<= X_5 251))
(assert (>= X_5 251))

; Unscaled Input 6 : (252, 252)
(assert (<= X_6 252))
(assert (>= X_6 252))

; Unscaled Input 7 : (251, 251)
(assert (<= X_7 251))
(assert (>= X_7 251))

; Unscaled Input 8 : (2, 2)
(assert (<= X_8 2))
(assert (>= X_8 2))

; Unscaled Input 9 : (0, 0)
(assert (<= X_9 0))
(assert (>= X_9 0))

; Unscaled Input 10 : (1, 1)
(assert (<= X_10 1))
(assert (>= X_10 1))

; Unscaled Input 11 : (250, 250)
(assert (<= X_11 250))
(assert (>= X_11 250))

; Unscaled Input 12 : (255, 255)
(assert (<= X_12 255))
(assert (>= X_12 255))

; Unscaled Input 13 : (253, 254)
(assert (<= X_13 254))
(assert (>= X_13 253))

; Unscaled Input 14 : (253, 253)
(assert (<= X_14 253))
(assert (>= X_14 253))

; Unscaled Input 15 : (255, 255)
(assert (<= X_15 255))
(assert (>= X_15 255))
```

Similar to the input specification, for generating the output specification, we also need to flatten the output column-wise as in List. 2.

List 2 Output Behavior Specification

```
(declare-const Y_0 Real)
(declare-const Y_1 Real)
(declare-const Y_2 Real)
(declare-const Y_3 Real)
(declare-const Y_4 Real)
(declare-const Y_5 Real)
(declare-const Y_6 Real)
(declare-const Y_7 Real)
(declare-const Y_8 Real)
(declare-const Y_9 Real)
(declare-const Y_10 Real)
(declare-const Y_11 Real)
(declare-const Y_12 Real)
(declare-const Y_13 Real)
(declare-const Y_14 Real)
(declare-const Y_15 Real)
; pixel classification constraints
(assert(== Y_0 10))
(assert(== Y_1 10))
(assert(== Y_2 10))
(assert(== Y_3 10))
(assert(== Y_4 10))
(assert(== Y_5 10))
(assert(== Y_6 10))
(assert(== Y_7 10))
(assert(== Y_8 1))
(assert(== Y_9 1))
(assert(== Y_10 1))
(assert(== Y_11 10))
(assert(== Y_12 10))
(assert(== Y_13 10))
(assert(== Y_14 10))
(assert(== Y_15 10))
```

In the example, we also make an assumption, as depicted in [Fig. 6 (b) right] that the SSN misclassifies two pixels due to the darkening effect, classifying them as '7' instead of '10'. Consequently, following the definition of robustness measures, we obtain the values for both the unperturbed image and the one corresponding to the UBAA attack as shown below in Table. 2. Here we need to emphasize that following Sec. 2.6 the concept of robustness sensitivity is only valid for an adversarial input.

Table 2

RobustnessMeasures	*Unperturbed*	*Under UBAA*
RV	1	0.8750
RS	-	1
RIOU	1	0.6154

Drawing on the concept of robustness measures for SSN verification against adversarial attacks, we introduce additional output verification properties. These properties are derived from the output pixel classification constraints, as provided in List. 3. For the example shown in Fig. 6 we restrict the considerable RV in $[0.9, 1]$, RIoU in $[0.8, 1]$ and RS to be always ≤ 100, for the output reachable set to be in the safe region. The proposed properties are as follows:

List 3 Verification Property Specification: Robustness Measures

```
; robustness measures specification
(assert(<= RV 1))                (assert(<= RIOU 1))
(assert(>= RV 0.9))              (assert(>= RIOU 0.8))
(assert(<= RS 0.8))
```

3 Evaluation

3.1 Reachability Analysis

To assess the impact of the adversarial attack on each dataset, we employ reachability analysis, a widely used concept [16,25,29,40–42,47]. The perturbed input is represented as a bounded set, and we compute the output reachable set layer-by-layer for the SSN. For the final layer of an SSN, i.e., pixel-classification layer, the pixel-class reachable set at a specific pixel is denoted as $pc(i,j) = \{l_1, ..., l_m\}$. This set is obtained by determining all cross-channel max-point candidates for each pixel in the input set. Consequently, we can obtain the pixel-class reachable set of the layer, which is equivalent to the reachable set of the SSN, denoted as $R_f = [pc(i,j)]_{h \times w}$, i.e., the collection of pixel classes at every index (i; j) [42].

Subsequently, we calculate the Robustness Values (RVs), Robustness Sensitivities (RSs), and Robust Intersection-over-Union (IoU) scores for all the images in the adversarial set based on the output reachable set.

We employ the "approx-star" method for reachability analysis in this paper. This method is preferred due to its computational efficiency, requiring less time and memory than "exact-star" methods. We direct our readers to refer to [40–42] for a more comprehensive understanding of the Star-based reachability analysis.

3.2 Neural Network Verification (NNV) Tool

For calculating the output reachable set using "approx-star," we make use of a readily available tool called "Neural Network Verification (NNV) Tool" [44]. It is a comprehensive set-based framework for verifying neural networks (NNs). It supports multiple reachability algorithms, enabling safety verification and robustness analysis of various deep neural network (DNN) types.

In the context of reachability analysis, the NNV tool computes output reachable sets layer-by-layer, starting from a given input. This input is defined by upper and lower bounds, representing perturbations around the actual input. As the analysis progresses through the layers, the reachable sets at the final layer represent the collection of all possible states of the DNN.

The primary objective of the NNV tool is to determine whether the DNN is deemed "safe." A DNN is considered safe when the specified safety properties determine no intersection between the output sets and the predefined unsafe region. By verifying safety conditions and analyzing robustness, the NNV tool aids in ensuring the reliability and trustworthiness of neural networks in various applications.

3.3 Results

In this section, we present a sample plot [Fig. 7] illustrating the average robustness measures of three MNIST networks, as described in Section 2.3. We conduct the analysis by subjecting 100 random digit images to the UBA attack and calculating the networks' average robustness against this attack with the following details: (1) max number of pixels attacked under UBAA: [1 2 3 4 5] and (2) $\epsilon = 1$ [Sec. 2.5].

When analyzing the outcomes of our experiment, we made several notable observations that shed light on the behavior of different robustness measures under varying degrees of adversarial attacks. These observations provide valuable insights into how these measures respond and behave in the face of adversarial perturbations.

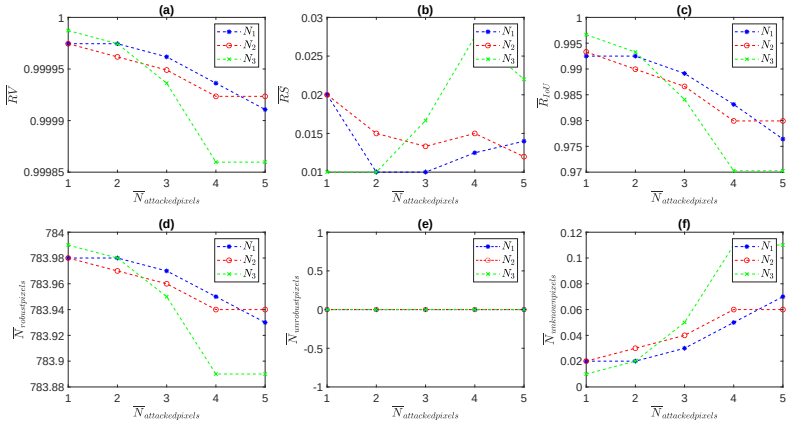

Fig. 7: The average robustness value, sensitivity, and IoU of MNIST SSNs.

Specifically, as we increased the number of adversarial attacks, we noticed a consistent downward trend in both the robustness value and the robust IoU (Intersection-over-Union). The robustness value quantifies the extent to which the SSN's predictions remain accurate after exposure to adversarial perturbations. In our analysis, this value consistently decreased with a greater number of attacks. Similarly, the robust IoU, which measures the overlap between predicted and ground-truth segments, also demonstrated a decreasing pattern with increased attacks. This reduction suggests that the adversarial perturbations adversely affect the network's ability to segment objects within images accurately.

Interestingly, we encountered a nuanced behavior when examining the robustness sensitivity. Unlike the robustness value and robust IoU, the trend in robustness sensitivity was not strictly uniform as the number of attacks increase. Robustness sensitivity gauges how sensitive the SSN's output segmentation is to changes in input pixels due to an attack. Our observations found that this sensitivity did not consistently follow a rigid trend with escalating adversarial attacks. This variability aligns with the inherent nature of robustness sensitiv-

ity, which can be influenced by the distribution and complexity of perturbations introduced by different attacks.

Overall, these observations reaffirm the theoretical definitions and expectations of these robustness measures in the context of various adversarial attacks. The decreasing trends in robustness value and robust IoU highlight the vulnerability of the SSN's segmentation performance to increasing adversarial perturbations. The non-uniform trend in robustness sensitivity emphasizes the intricate interplay between attack characteristics and the network's responsiveness to perturbations, leading to varying degrees of sensitivity under different attack scenarios. Such insights are crucial for understanding the strengths and limitations of these robustness measures and guiding the development of more resilient semantic segmentation networks in the future.

4 Conclusion and Future Ideas

In this paper, we proposed a benchmark framework for the formal verification of semantic segmentation neural networks. Our study aimed to address the challenges in ensuring the safety and reliability of these networks, which are increasingly being utilized in critical applications such as autonomous vehicles, medical imaging, and surveillance systems. By establishing a standardized benchmark, we aimed to facilitate the first step towards a fair comparison of different verification methods and tools, promoting advancements in the field of formal verification for semantic segmentation neural networks. The benchmark framework we presented encompasses a diverse set of neural network architectures, datasets, and verification properties, representing two commonly used datasets: MNIST and M2NIST. We also provided a detailed specification format in vnnlib files to describe the verification properties, enabling the seamless integration of different tools and approaches. This standardization allows researchers and developers to easily evaluate and compare the effectiveness of their verification techniques.

While our proposed benchmark framework represents a significant step towards formal verification in semantic segmentation neural networks, there are several avenues for future research and enhancement. Firstly, we plan to continuously update and expand the benchmark by incorporating new neural network architectures, datasets, and verification properties that emerge in the field. This will ensure that the benchmark remains up-to-date and reflective of the latest advancements in semantic segmentation tasks. Moreover, we aim to collaborate with the research community to gather feedback and incorporate suggestions for improving the benchmark. This will allow us to refine the benchmark based on the practical experiences and insights of researchers working on formal verification for semantic segmentation networks. Furthermore, we intend to conduct comprehensive evaluations and comparisons of existing formal verification methods and tools using the benchmark. By doing so, we can identify the strengths and limitations of different approaches, helping to guide researchers in selecting the most suitable verification techniques for their specific use cases. Lastly, we will explore the integration of novel techniques and advances in formal verifi-

cation to enhance the benchmark's capabilities and coverage. This may include leveraging machine learning-based methods, formal synthesis, and abstraction techniques to address the challenges posed by the complexity and scalability of large-scale semantic segmentation networks.

In conclusion, our proposed benchmark framework for the formal verification of semantic segmentation neural networks lays the foundation for advancing the safety and reliability of these networks in critical applications. We look forward to collaborating with the research community to further refine and expand the benchmark, fostering progress in the field of formal verification for semantic segmentation neural networks.

Acknowledgements. The material presented in this paper is based upon work supported by the National Science Foundation (NSF) through grant numbers 1910017, 2028001, 2220426, and 2220401, and the Defense Advanced Research Projects Agency (DARPA) under contract number FA8750-23-C-0518, and the Air Force Office of Scientific Research (AFOSR) under contract number FA9550-22-1-0019 and FA9550-23-1-0135. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of AFOSR, DARPA, or NSF. We also want to thank our colleagues, Tianshu and Serena for their valuable feedback.

References

1. Anderson, G., Pailoor, S., Dillig, I., Chaudhuri, S.: Optimization and abstraction: A synergistic approach for analyzing neural network robustness. In: Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. p. 731–744. PLDI 2019, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3314221.3314614
2. Bak, S., Liu, C., Johnson, T.: The second international verification of neural networks competition (vnn-comp 2021): Summary and results. arXiv preprint arXiv:2109.00498 (2021)
3. Blum, H., Sarlin, P.E., Nieto, J., Siegwart, R., Cadena, C.: Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving. In: proceedings of the IEEE/CVF international conference on computer vision workshops. pp. 0–0 (2019)
4. Botoeva, E., Kouvaros, P., Kronqvist, J., Lomuscio, A., Misener, R.: Efficient verification of relu-based neural networks via dependency analysis. Proceedings of the AAAI Conference on Artificial Intelligence **34**(04), 3291–3299 (Apr 2020). https://doi.org/10.1609/aaai.v34i04.5729
5. Brazil, G., Yin, X., Liu, X.: Illuminating pedestrians via simultaneous detection & segmentation. In: Proceedings of the IEEE international conference on computer vision. pp. 4950–4959 (2017)
6. Brix, C., Müller, M.N., Bak, S., Johnson, T.T., Liu, C.: First three years of the international verification of neural networks competition (vnn-comp). International Journal on Software Tools for Technology Transfer pp. 1–11 (2023)
7. Brostow, G.J., Fauqueur, J., Cipolla, R.: Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters **30**(2), 88–97 (2009)

8. Dathathri, S., Dvijotham, K., Kurakin, A., Raghunathan, A., Uesato, J., Bunel, R., Shankar, S., Steinhardt, J., Goodfellow, I., Liang, P., Kohli, P.: Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming (2020)
9. Demarchi, S.: VNN-LIB — vnnlib.org. https://www.vnnlib.org/, [Accessed 31-07-2023]
10. Ehlers, R.: Formal verification of piece-wise linear feed-forward neural networks. In: Automated Technology for Verification and Analysis: 15th International Symposium, ATVA 2017, Pune, India, October 3–6, 2017, Proceedings 15. pp. 269–286. Springer (2017)
11. Fazlyab, M., Morari, M., Pappas, G.J.: Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. IEEE Transactions on Automatic Control pp. 1–1 (2020). https://doi.org/10.1109/TAC.2020.3046193
12. Flohr, F., Gavrila, D., et al.: Pedcut: an iterative framework for pedestrian segmentation combining shape models and multiple data cues. In: BMVC (2013)
13. Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., Vechev, M.: Ai2: Safety and robustness certification of neural networks with abstract interpretation. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 3–18. IEEE (2018)
14. Girard-Satabin, J., Alberti, M., Bobot, F., Chihani, Z., Lemesle, A.: Caisar: A platform for characterizing artificial intelligence safety and robustness. arXiv preprint arXiv:2206.03044 (2022)
15. Guo, Y., Gao, Y., Shen, D.: Deformable mr prostate segmentation via deep feature learning and sparse patch matching. IEEE transactions on medical imaging **35**(4), 1077–1089 (2015)
16. Huang, C., Fan, J., Li, W., Chen, X., Zhu, Q.: Reachnn: Reachability analysis of neural-network controlled systems. ACM Transactions on Embedded Computing Systems (TECS) **18**(5s), 1–22 (2019)
17. Huang, X., Kwiatkowska, M., Wang, S., Wu, M.: Safety verification of deep neural networks. In: International Conference on Computer Aided Verification. pp. 3–29. Springer (2017)
18. Ivanov, R., Weimer, J., Alur, R., Pappas, G.J., Lee, I.: Verisig: verifying safety properties of hybrid systems with neural network controllers. In: Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control. pp. 169–178 (2019)
19. Kamann, C., Rother, C.: Benchmarking the robustness of semantic segmentation models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8828–8838 (2020)
20. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J.: Reluplex: An efficient smt solver for verifying deep neural networks. In: International Conference on Computer Aided Verification. pp. 97–117. Springer (2017)
21. Katz, G., Huang, D.A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., et al.: The marabou framework for verification and analysis of deep neural networks. In: International Conference on Computer Aided Verification. pp. 443–452. Springer (2019)
22. LeCun, Y.: The mnist database of handwritten digits. http://yann. lecun. com/exdb/mnist/ (1998)
23. Li, B., Liu, S., Xu, W., Qiu, W.: Real-time object detection and semantic segmentation for autonomous driving. In: MIPPR 2017: Automatic Target Recognition and Navigation. vol. 10608, pp. 167–174. SPIE (2018)

24. Liu, C., Arnon, T., Lazarus, C., Strong, C., Barrett, C., Kochenderfer, M.J., et al.: Algorithms for verifying deep neural networks. Foundations and Trends® in Optimization **4**(3-4), 244–404 (2021)
25. Lomuscio, A., Maganti, L.: An approach to reachability analysis for feed-forward relu neural networks. arXiv preprint arXiv:1706.07351 (2017)
26. Mohapatra, J., Weng, T.W., Chen, P.Y., Liu, S., Daniel, L.: Towards verifying robustness of neural networks against a family of semantic perturbations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
27. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2574–2582 (2016)
28. Müller, M.N., Brix, C., Bak, S., Liu, C., Johnson, T.T.: The third international verification of neural networks competition (vnn-comp 2022): summary and results. arXiv preprint arXiv:2212.10376 (2022)
29. Ruan, W., Huang, X., Kwiatkowska, M.: Reachability analysis of deep neural networks with provable guarantees. arXiv preprint arXiv:1805.02242 (2018)
30. Ruan, W., Wu, M., Sun, Y., Huang, X., Kroening, D., Kwiatkowska, M.: Global robustness evaluation of deep neural networks with provable guarantees for the l_0 norm. arXiv preprint arXiv:1804.05805 (2018)
31. Shriver, D., Elbaum, S., Dwyer, M.B.: Dnnv: A framework for deep neural network verification. In: International Conference on Computer Aided Verification. pp. 137–150. Springer (2021)
32. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. Proceedings of the ACM on Programming Languages **3**(POPL), 41 (2019)
33. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
34. Szeliski, R.: Computer vision: algorithms and applications 2nd edition (2021)
35. Tao, X., Zhang, D., Ma, W., Liu, X., Xu, D.: Automatic metallic surface defect detection and recognition with convolutional neural networks. Applied Sciences **8**(9), 1575 (2018)
36. Thoma, M.: A survey of semantic segmentation. arXiv preprint arXiv:1602.06541 (2016)
37. Tjeng, V., Xiao, K.Y., Tedrake, R.: Evaluating robustness of neural networks with mixed integer programming. In: International Conference on Learning Representations (2019)
38. Tran, H.D., Bak, S., Xiang, W., Johnson, T.T.: Verification of deep convolutional neural networks using imagestars. In: 32nd International Conference on Computer-Aided Verification (CAV). Springer (July 2020)
39. Tran, H.D., Cei, F., Lopez, D.M., Johnson, T.T., Koutsoukos, X.: Safety verification of cyber-physical systems with reinforcement learning control. In: ACM SIGBED International Conference on Embedded Software (EMSOFT'19). ACM (October 2019)
40. Tran, H.D., Manzanas Lopez, D., Musau, P., Yang, X., Nguyen, L.V., Xiang, W., Johnson, T.T.: Star-based reachability analysis of deep neural networks. In: Formal Methods–The Next 30 Years: Third World Congress, FM 2019, Porto, Portugal, October 7–11, 2019, Proceedings 3. pp. 670–686. Springer (2019)

41. Tran, H.D., Musau, P., Lopez, D.M., Yang, X., Nguyen, L.V., Xiang, W., Johnson, T.T.: Star-based reachability analsysis for deep neural networks. In: 23rd International Symposisum on Formal Methods (FM'19). Springer International Publishing (October 2019)
42. Tran, H.D., Pal, N., Musau, P., Lopez, D.M., Hamilton, N., Yang, X., Bak, S., Johnson, T.T.: Robustness verification of semantic segmentation neural networks using relaxed reachability. In: Computer Aided Verification: 33rd International Conference, CAV 2021, Virtual Event, July 20–23, 2021, Proceedings, Part I 33. pp. 263–286. Springer (2021)
43. Tran, H.D., Xiang, W., Johnson, T.T.: Verification approaches for learning-enabled autonomous cyber-physical systems. IEEE Design & Test (2020)
44. Tran, H.D., Yang, X., Lopez, D.M., Musau, P., Nguyen, L.V., Xiang, W., Bak, S., Johnson, T.T.: Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In: International Conference on Computer Aided Verification. pp. 3–17. Springer (2020)
45. Tseng, Y.H., Jan, S.S.: Combination of computer vision detection and segmentation for autonomous driving. In: 2018 IEEE/ION Position, Location and Navigation Symposium (PLANS). pp. 1047–1052. IEEE (2018)
46. Wang, Z., Wei, L., Wang, L., Gao, Y., Chen, W., Shen, D.: Hierarchical vertex regression-based segmentation of head and neck ct images for radiotherapy planning. IEEE Transactions on Image Processing **27**(2), 923–937 (2017)
47. Xiang, W., Johnson, T.T.: Reachability analysis and safety verification for neural network control systems. arXiv preprint arXiv:1805.09944 (2018)
48. Xiang, W., Musau, P., Wild, A.A., Lopez, D.M., Hamilton, N., Yang, X., Rosenfeld, J., Johnson, T.T.: Verification for machine learning, autonomy, and neural networks survey. arXiv preprint arXiv:1810.01989 (2018)
49. Xie, X., Kersting, K., Neider, D.: Neuro-symbolic verification of deep neural networks. arXiv preprint arXiv:2203.00938 (2022)
50. Yuan, X., He, P., Zhu, Q., Li, X.: Adversarial examples: Attacks and defenses for deep learning. IEEE transactions on neural networks and learning systems **30**(9), 2805–2824 (2019)
51. Zhang, H., Weng, T.W., Chen, P.Y., Hsieh, C.J., Daniel, L.: Efficient neural network robustness certification with general activation functions. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31, pp. 4939–4948. Curran Associates, Inc. (2018)
52. Zhu, X., Suk, H.I., Lee, S.W., Shen, D.: Subspace regularized sparse multitask learning for multiclass neurodegenerative disease identification. IEEE Transactions on Biomedical Engineering **63**(3), 607–618 (2015)
53. Zhu, X., Suk, H.I., Shen, D.: A novel matrix-similarity based loss function for joint regression and classification in ad diagnosis. NeuroImage **100**, 91–105 (2014)