Improving Neural Network Malware Classifiers

Prof. Kevin Leach
kevin.leach@vanderbilt.edu

Prof. Taylor Johnson
taylor.johnson@vanderbilt.edu

Institute for Software Integrated Systems, Vanderbilt University

National Security Agency

January 11, 2024

VANDERBILT UNIVERSITY

Overview

- **Malware** is pervasive – millions of new samples are discovered each year
  - There are **too many samples** uncovered each year to *manually reverse engineer* all of them

Global detections 2018-2019			
	2018	**2019**	**% Change**
Overall	50,170,502	50,510,960	1%
Business	8,498,934	9,599,305	13%
Consumer	41,671,568	40,911,655	-2%

MalwareBytes 2020 State of Malware Report

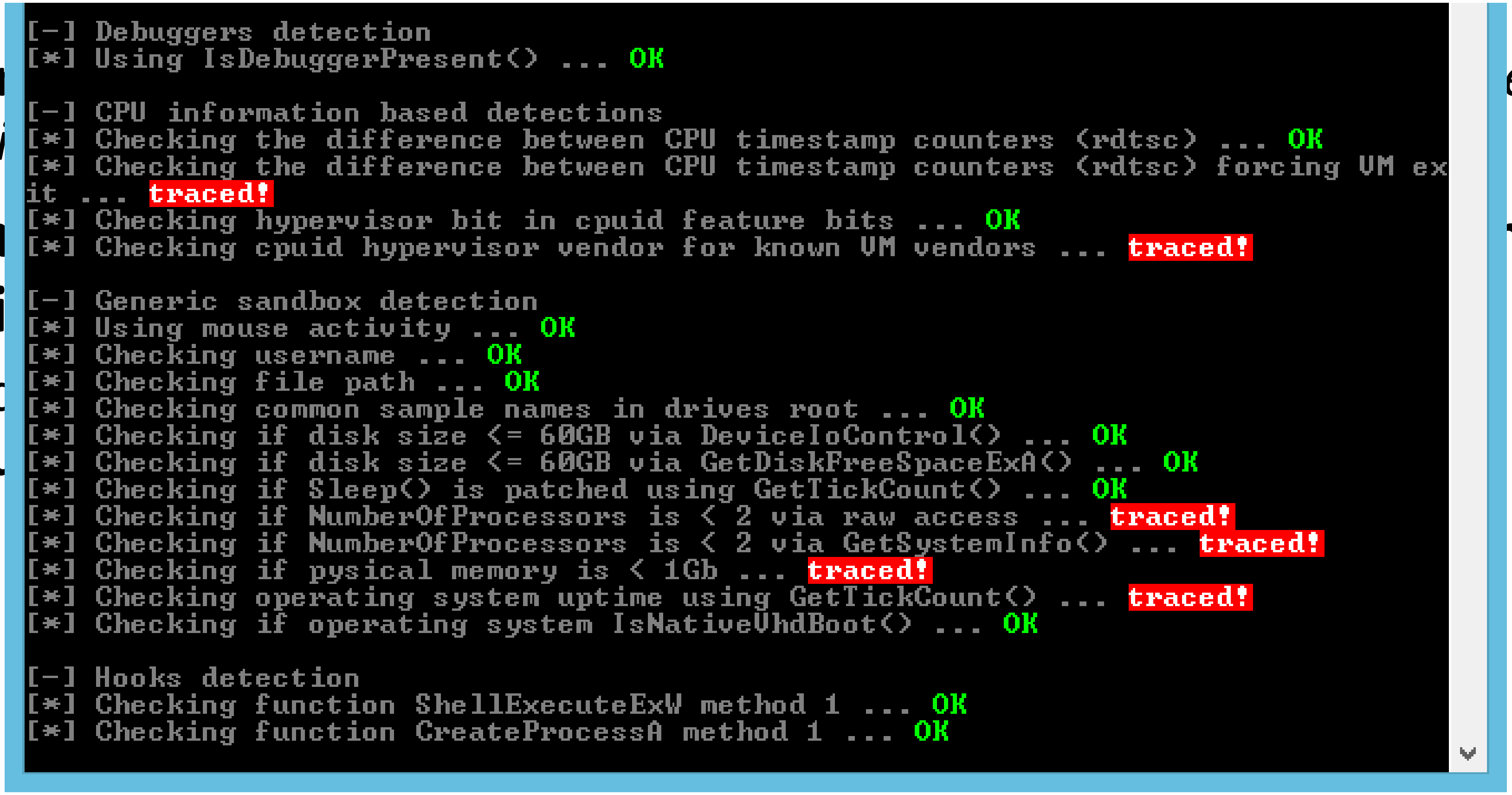
Overview

- **Malware** is pervasive – millions of new samples are discovered each year
  - There are **too many samples** uncovered each year to *manually reverse engineer* all of them
- **Automated malware analysis** depends on effective **triage and classification**
  - Modern malware samples exhibit **stealthiness** and **complex static obfuscation**

VANDERBILT UNIVERSITY

Overview

- **Malware** is pervasive – millions of new samples are discovered each year
  - Ther *erse engi*
- **Autom nd classifi**
  - Moc **obfu**



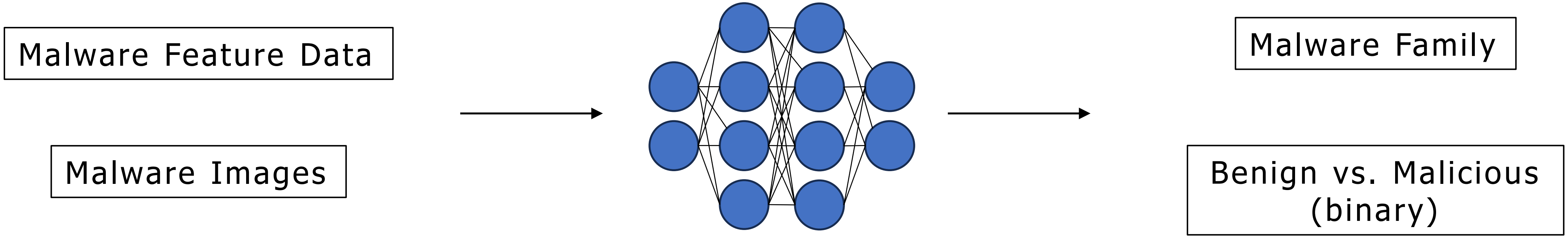
Overview

- **Malware** is pervasive – millions of new samples are discovered each year
  - There are **too many samples** uncovered each year to *manually reverse engineer* all of them
- **Automated malware analysis** depends on effective **triage and classification**
  - Modern malware samples exhibit **stealthiness** and **complex static obfuscation**
- **Neural malware classifiers** lack *verifiability* and *robustness* against stealthiness and obfuscation

VANDERBILT UNIVERSITY

Malware Classification with Neural Networks

- Neural Networks are a popular means of classification:
  - Benign vs. malicious
  - Malware family



- Neural networks lack explainability, robustness, and verifiability (for malware analysis)

VANDERBILT UNIVERSITY

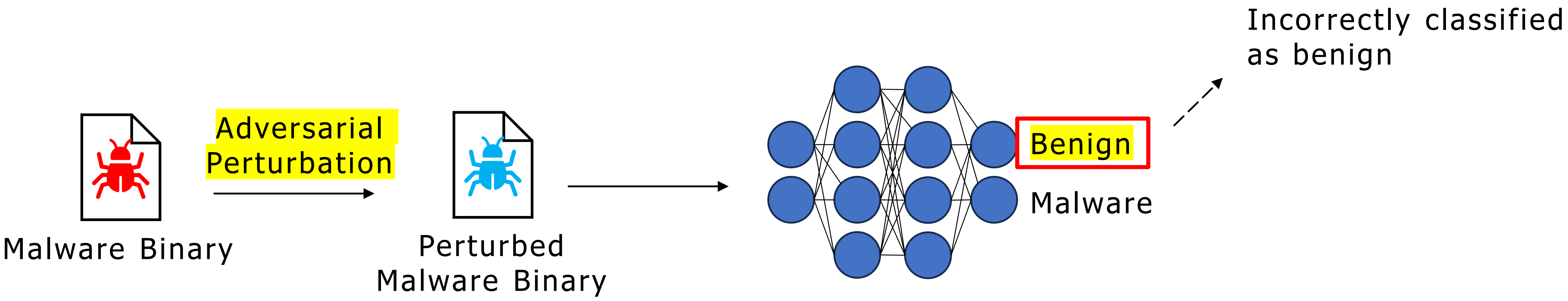
Outline

- Malware Analysis and Classification
- Adversarial Perturbation
- Semantics-aware Augmentation
- Verification of Neural Classifiers

VANDERBILT UNIVERSITY

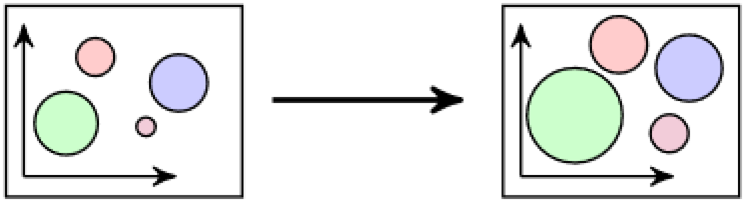
Adversarial Perturbation

- Adversary can *perturb* input sample to cause **incorrect classification**



Assuring Malware Classification with Augmentation

- Augmentation via perturbation is widely-used to improve machine learning under sparse data

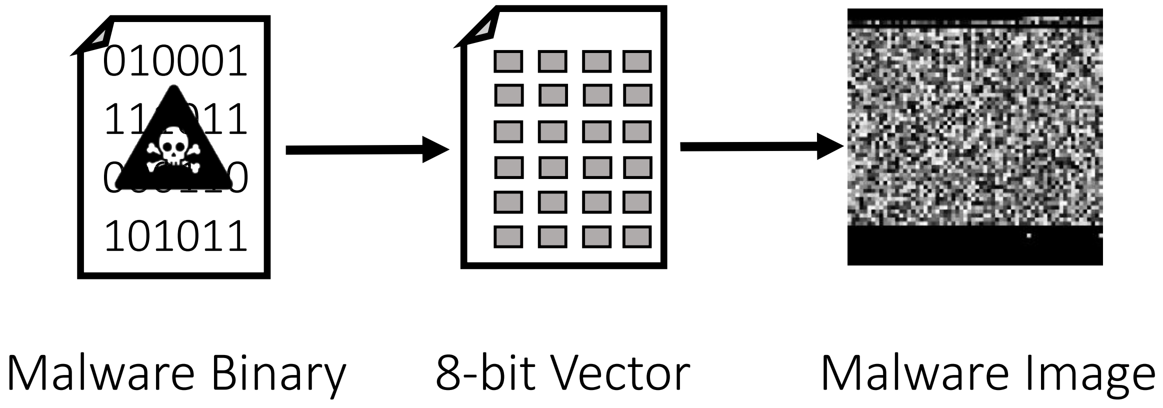


- By introducing *small changes* to a sample, the hope is to **cover more** of the feature space to **improve training**
  - Providing more assurance about the correctness of the classifier

VANDERBILT UNIVERSITY

Malware Classification with Neural Networks

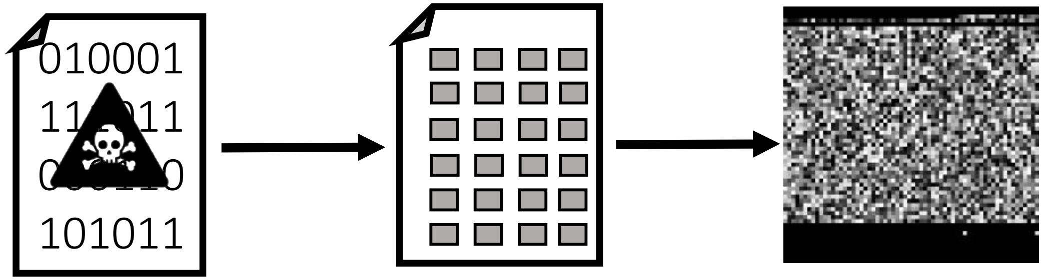
- Two high level classification approaches
  - **1. Malware images** (*byteplots)* leverage computer vision approaches (CNNs)



VANDERBILT UNIVERSITY

Malware Classification with Neural Networks

- Two high level classification approaches
  - **1. Malware images** (*byteplots)* leverage computer vision approaches (CNNs)



Malware Binary 8-bit Vector Malware Image

- **Problem:** Verification and robustness measured with respect to *perturbed byteplots…*
  - What does that mean?

VANDERBILT UNIVERSITY

Malware Classification with Neural Networks

- Two high level classification approaches
  - **2. Static** and **dynamic** features extracted from input binary (BODMAS)

Feature Type	Count	Max Range	Example
Continuous	5	[5.0, 2.0e5]	Entropy
Categorical	8	[0.0, 6.5e4]	Machine type
Discrete Large	34	[0.0, 4.3e9]	Byte distribution
Binary	5	[0, 1]	Presence of section

Feature Type	Count	Max Range	Example
Hash categorical	500	[-650, 15]	Hash of original file
Hash discrete	1531	[-8.0e6, 1.6e9]	Hash of system type
Memory	16	[0.0, 4.0e9]	Size of file
Null	222	[-31.0, 60.0]	other

VANDERBILT UNIVERSITY

Malware Classification with Neural Networks

- Two high level classification approaches
  - **2. Static** and **dynamic** features extracted from input binary (BODMAS)

Feature Type	Count	Max Range	Example
Continuous	5	[5.0, 2.0e5]	Entropy
Categorical	8	[0.0, 6.5e4]	Machine type
Discrete Large	34	[0.0, 4.3e9]	Byte distribution
Binary	5	[0, 1]	Presence of section

Feature Type	Count	Max Range	Example
Hash categorical	500	[-650, 15]	Hash of original file
Hash discrete	1531	[-8.0e6, 1.6e9]	Hash of system type
Memory	16	[0.0, 4.0e9]	Size of file
Null	222	[-31.0, 60.0]	other

- **Problem:** how do we perturb data meaningfully?

VANDERBILT UNIVERSITY

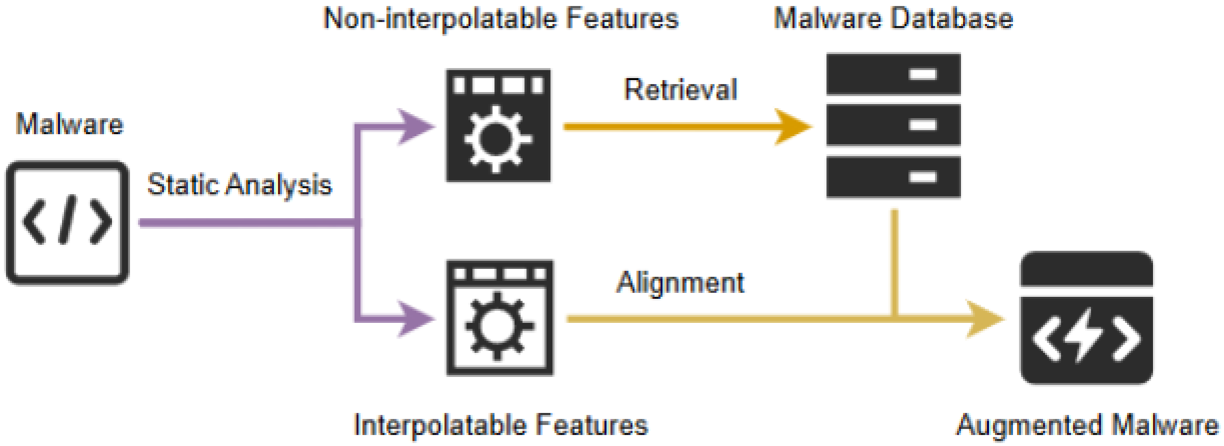
Outline

- Malware Analysis and Classification
- Adversarial Perturbation
- Semantics-aware Augmentation
- Verification of Neural Classifiers

VANDERBILT UNIVERSITY

Semantics-aware Augmentation and Verification

- Leverage distinction between **interpolatable** and **non-interpolatable** features



- **Interpolatable:** quantities like length, entropy, number of sections
- **Non-interpolatable:** hash values, strings

VANDERBILT UNIVERSITY

Semantics-aware Augmentation

- 95% of **top-5** neighbors of *every sample* are in the same **family**
  - Thus, we can *mix* a sample with its neighbors that are likely the same family

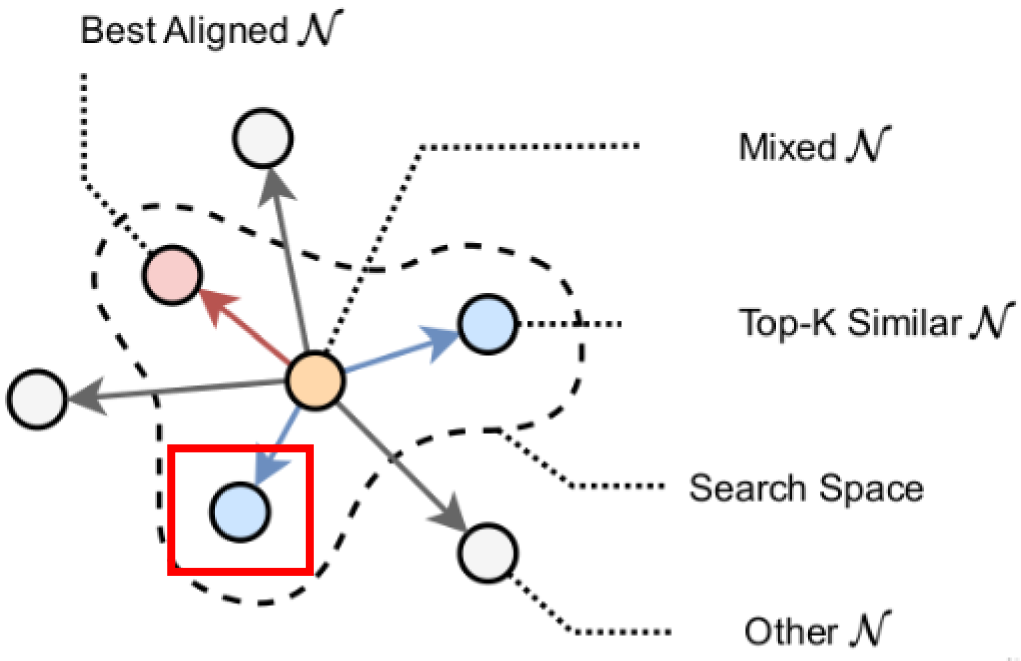
- Features of neighbors can be *borrowed* to produce a new variant in the feature space
  - This mixture results in a more realistic sample (in the feature space)

- **Insight:** we adapt **MixUp** from computer vision literature
  - Challenge classifier with *hard variants* generated by mixing feature space

VANDERBILT UNIVERSITY

Semantics-aware Augmentation

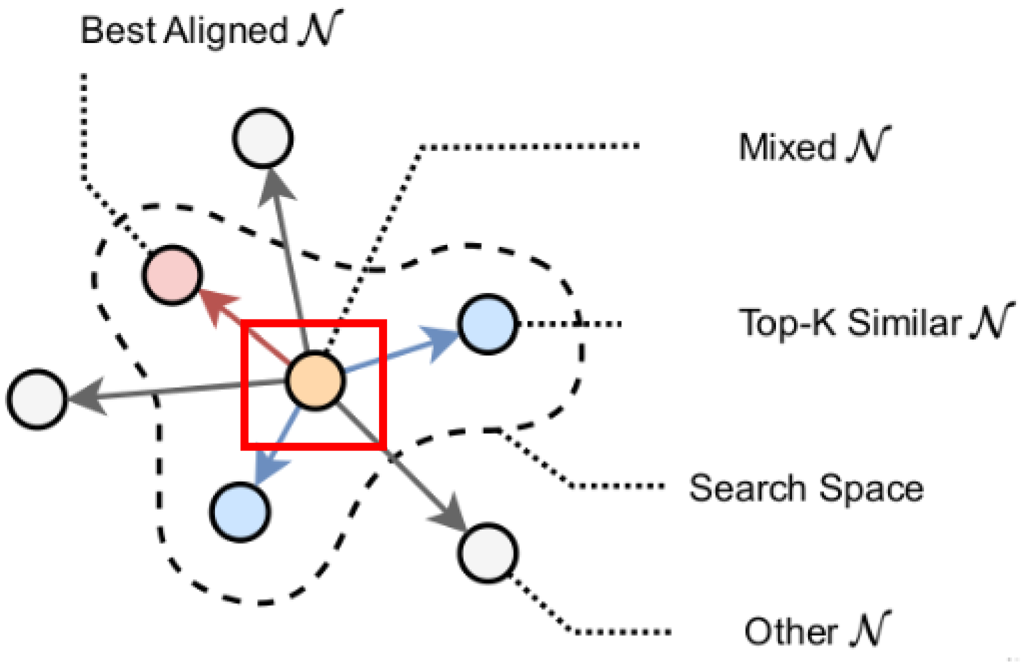
1. Given **input sample ($\boldsymbol{s_i}$)**, identify random neighbor (s_i') and embed both



VANDERBILT UNIVERSITY

Semantics-aware Augmentation

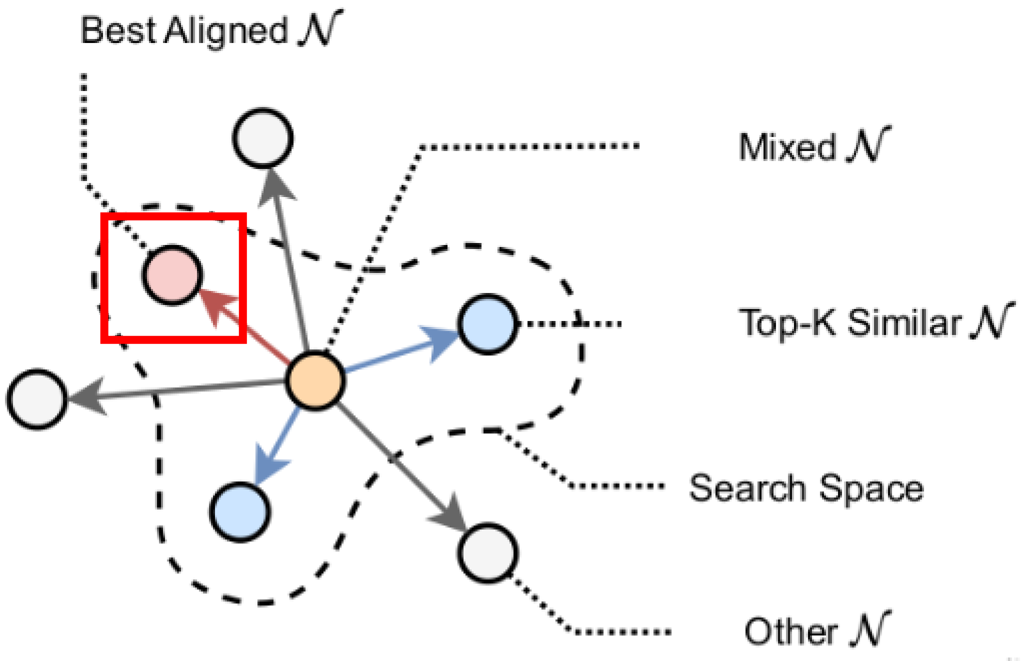
1. Given **input sample ($\boldsymbol{s_i}$)**, identify random neighbor (s_i') and embed both
2. Apply **mixup** by combining features from random neighbor:
   - $\tilde{s}_i = \alpha s_i + (1 - \alpha) s_i'; \quad 0 \leq \alpha \leq 1$



VANDERBILT UNIVERSITY

Semantics-aware Augmentation

1. Given **input sample ($\boldsymbol{s_i}$)**, identify random neighbor (s_i') and embed both
2. Apply **mixup** by combining features from random neighbor:
   - $\tilde{s}_i = \alpha s_i + (1-\alpha)s_i'; \quad 0 \leq \alpha \leq 1$
3. For **non-interpolatable features,** identify nearest *concrete value* in neighbor starting with $\tilde{s}_i$.
   - For example: s_i loads `win32.dll`
   - $\tilde{s}_i$ might load `shell32.dll` instead



VANDERBILT UNIVERSITY

Using Augmentation for Neural Verification

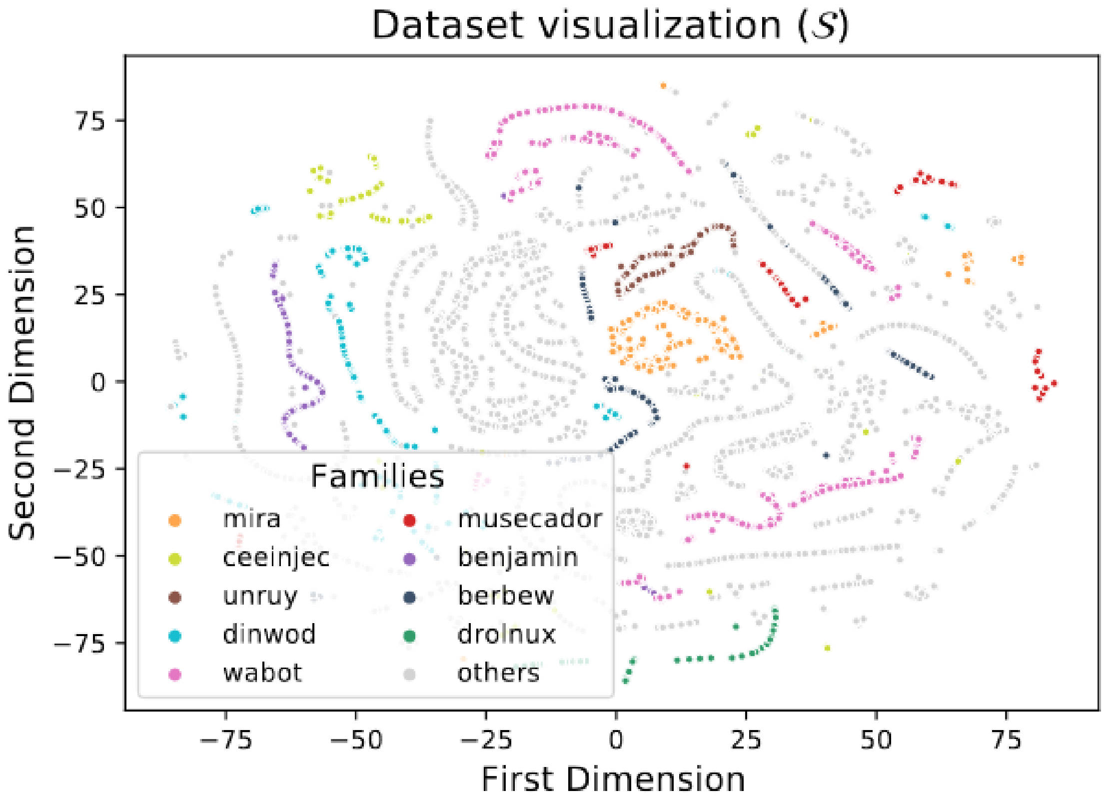
- The **mixed samples** we generate can serve as **hard examples** from which we:
  1. Improve training of subsequent classification
     - When malware corpora are sparsely-labeled
     - When malware corpora become outdated
     - When malware corpora require significant reverse engineering effort

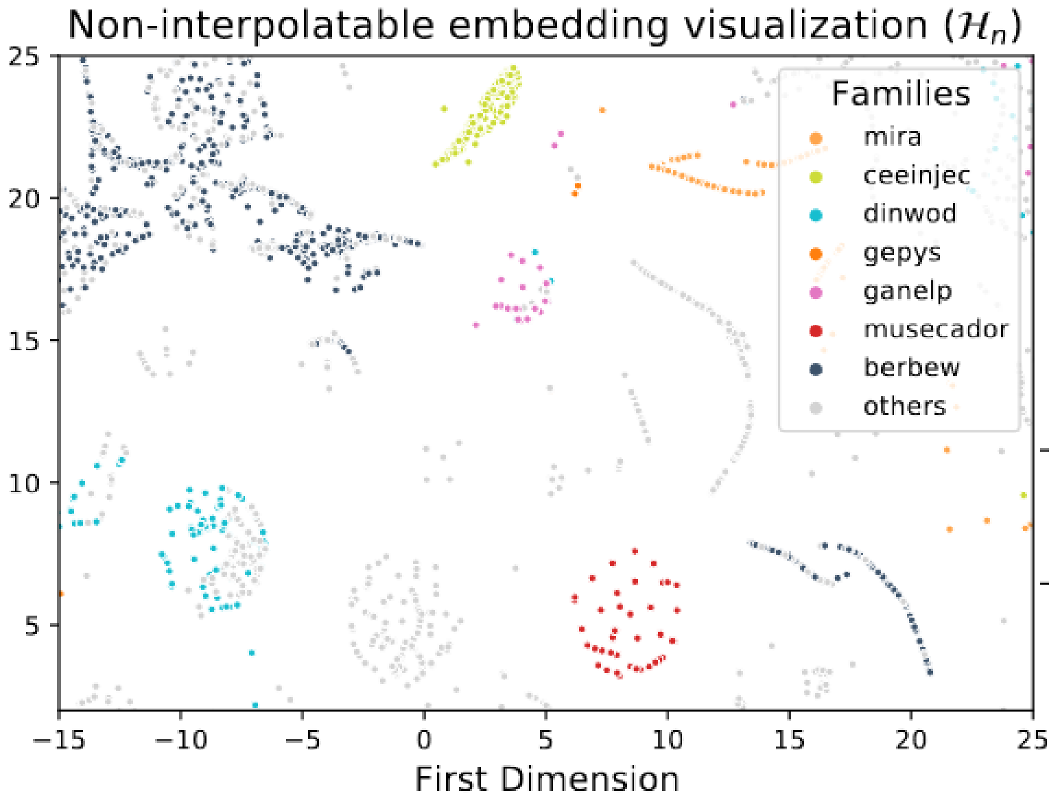
  2. Provide stronger verification guarantees of neural classifiers
     - When verification requires hard samples for bootstrapping
     - When classifiers require robustness bounds

VANDERBILT
UNIVERSITY

Preliminary Results

- Non-interpolatable features cluster in the embedding space

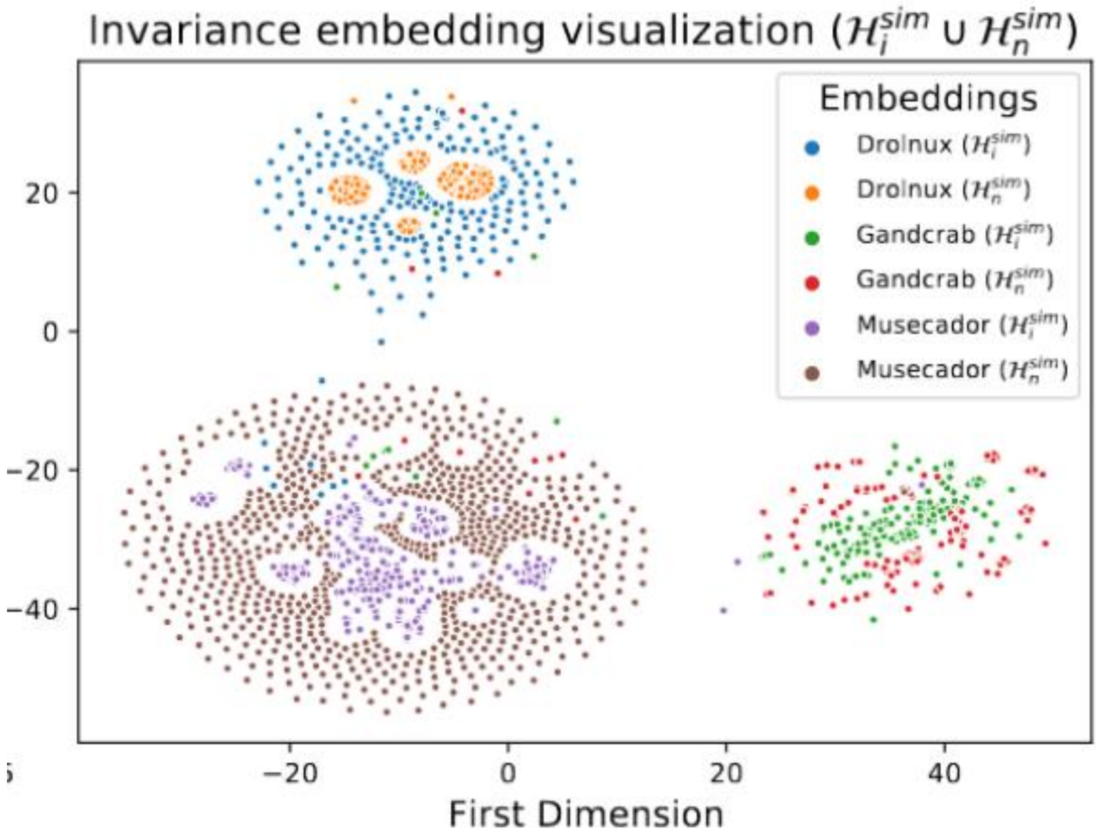




VANDERBILT UNIVERSITY

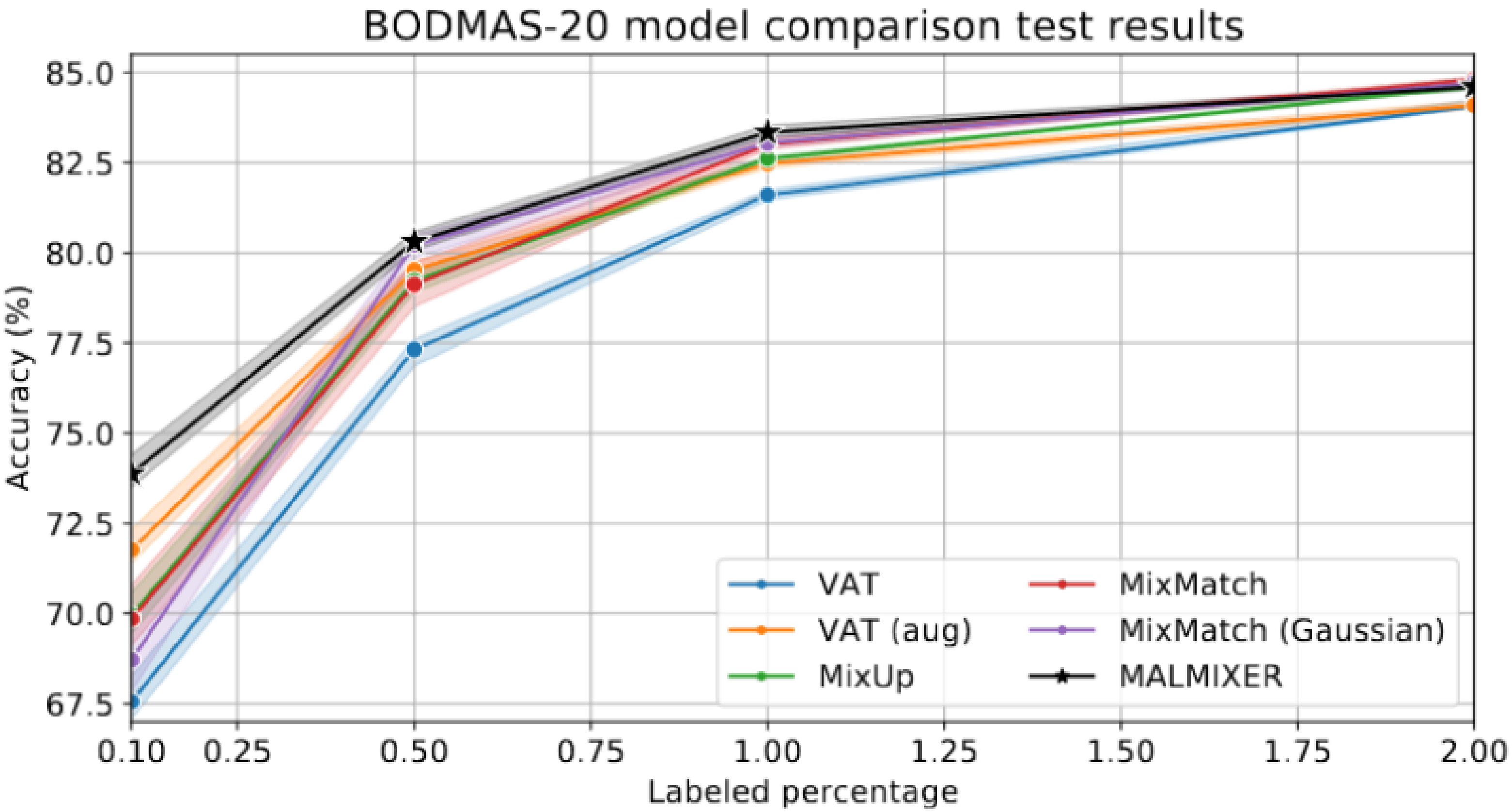
Preliminary Results: MalMixer

- MalMixer produces new samples in the embedding space near the same family



Preliminary Results: MalMixer

- MalMixer can help improve classification performance in low-resource settings



VANDERBILT UNIVERSITY

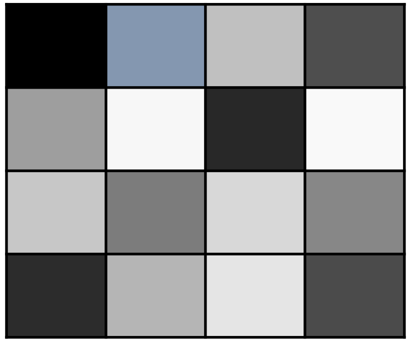
Outline

- Malware Analysis and Classification
- Adversarial Perturbation
- Semantics-aware Augmentation
- Verification of Neural Classifiers

VANDERBILT
UNIVERSITY

Malware Byteplot Robustness Example

4x4 Grayscale Image



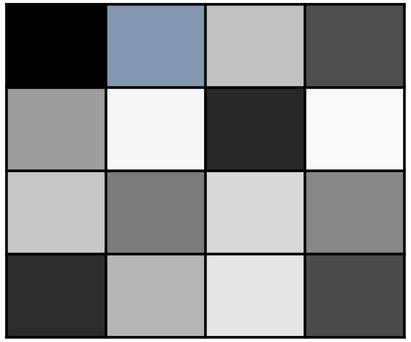
$y_{\text{true}} = A$

VANDERBILT UNIVERSITY

Malware Byteplot Robustness Example

Standard Performance Metrics

4x4 Grayscale Image



$y_{\text{true}} = A$

$\downarrow$

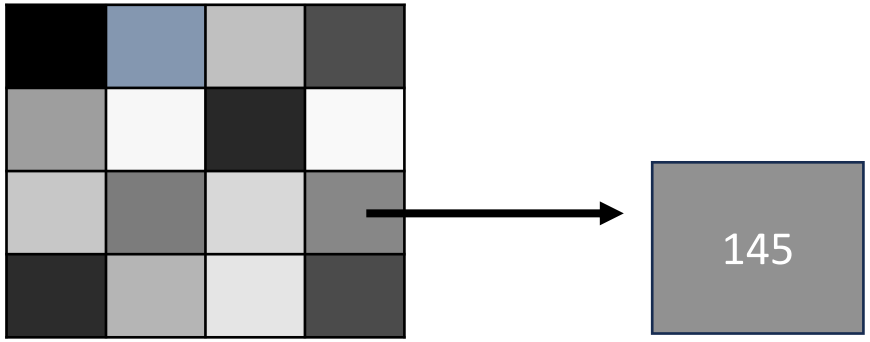
$\boxed{y_{\text{pred}} = A}$

VANDERBILT UNIVERSITY

Malware Byteplot Robustness Example

Robustness Performance: $\boldsymbol{\epsilon = 2}$

4x4 Grayscale Image

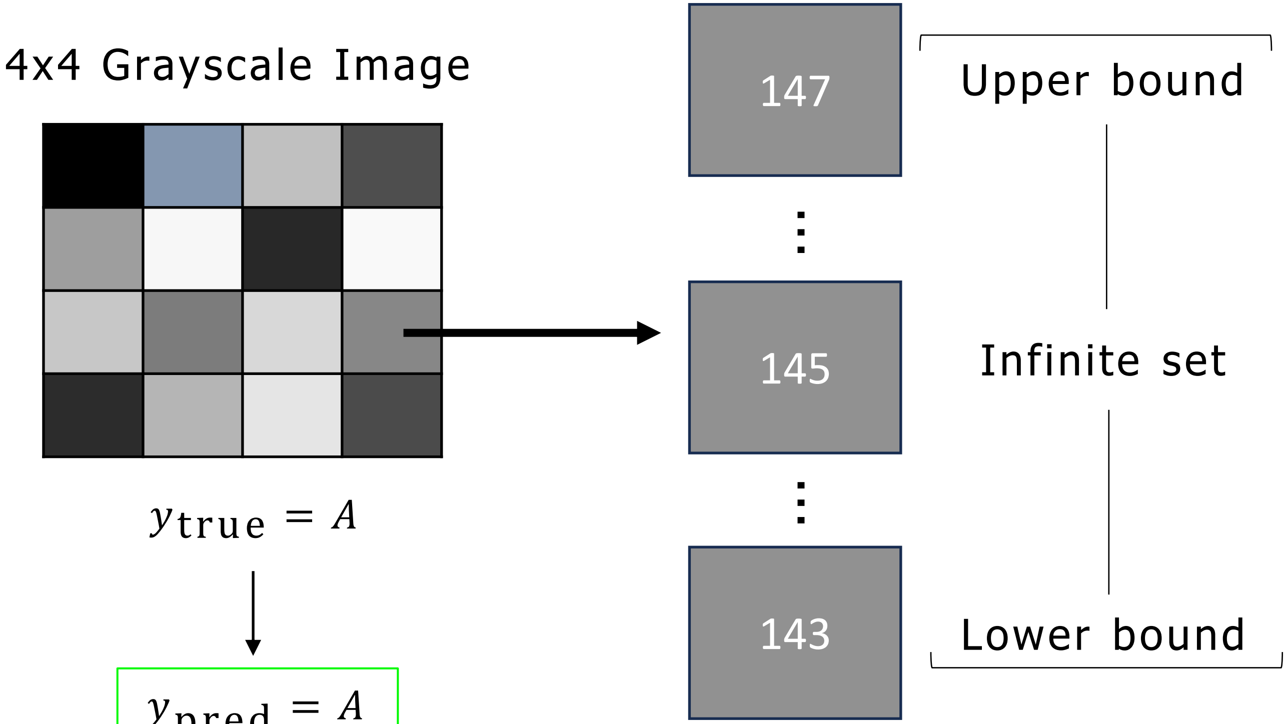


$y_{\text{true}} = A$

$y_{\text{pred}} = A$

Malware Byteplot Robustness Example

Robustness Performance: $\epsilon = 2$



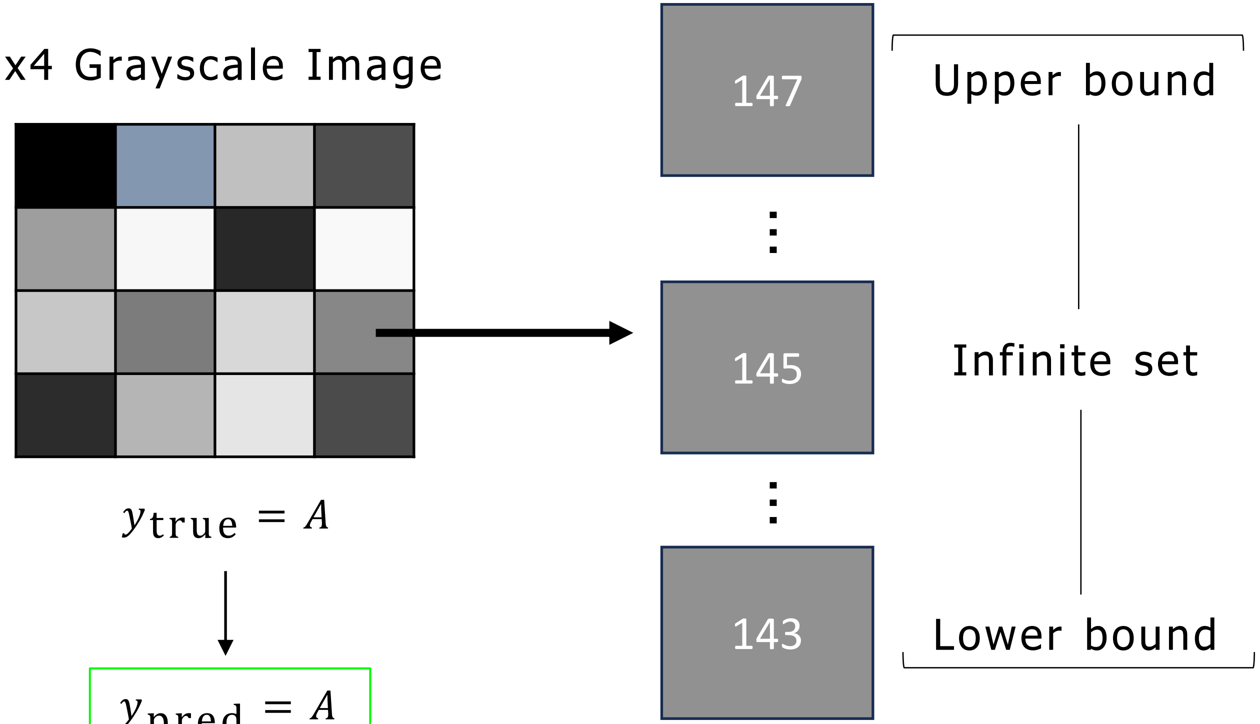
$y_{\mathrm{true}} = A$

$\boxed{y_{\mathrm{pred}} = A}$

Malware Byteplot Robustness Example

Robustness Performance: $\epsilon = 2$

4x4 Grayscale Image



$y_{\text{true}} = A$

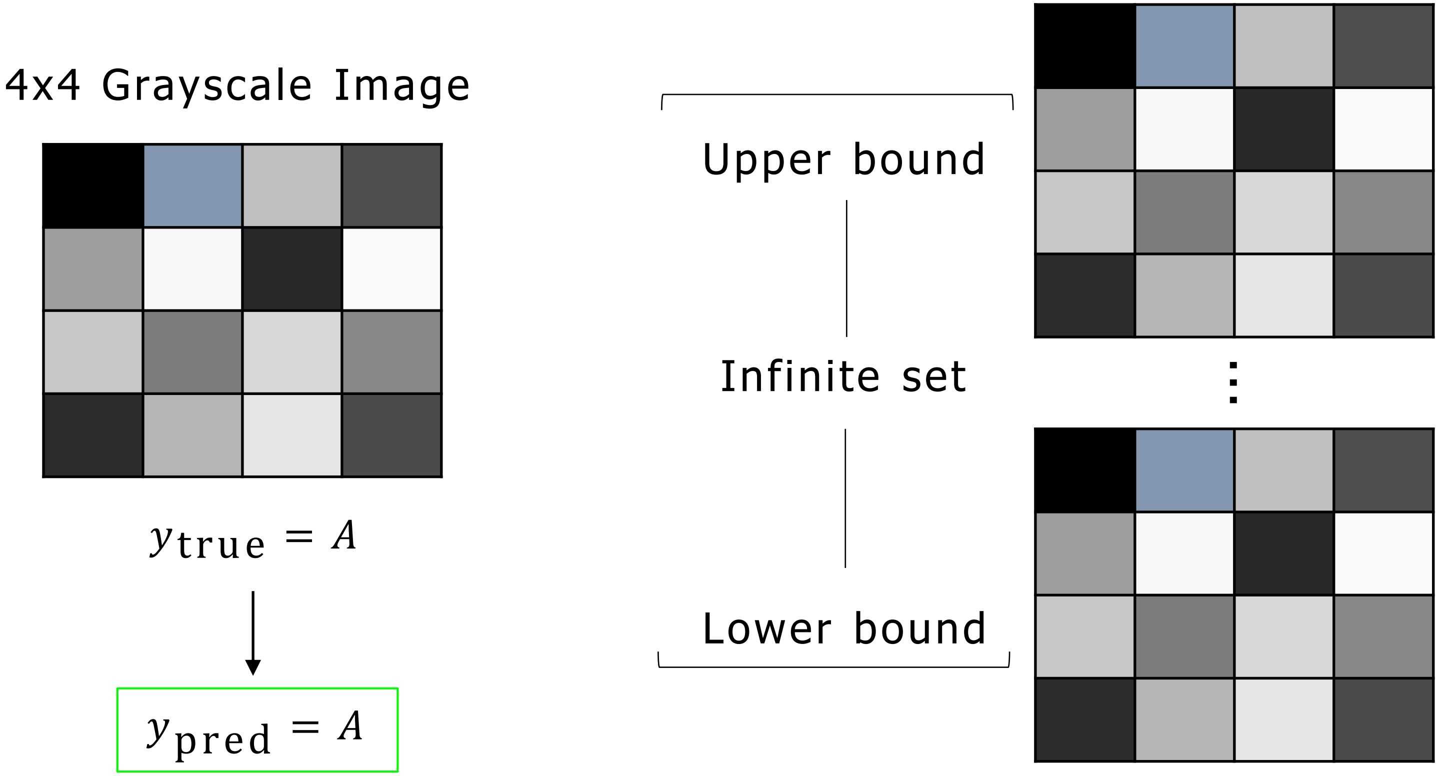
$y_{\text{pred}} = A$

For all pixels in the image

$\forall i \in \{1, 2, 3, 4\}, \forall j \in \{1, 2, 3, 4\}$

Malware Byteplot Robustness Example

Robustness Performance: $\epsilon = 2$



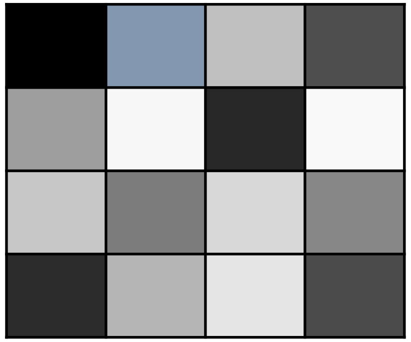
VANDERBILT UNIVERSITY

Malware Byteplot Robustness Example

Robustness Performance: $\epsilon = 2$

Robust to an $\mathcal{L}_\infty$ adversarial perturbation of size $\boldsymbol{\epsilon = 2}$

4x4 Grayscale Image



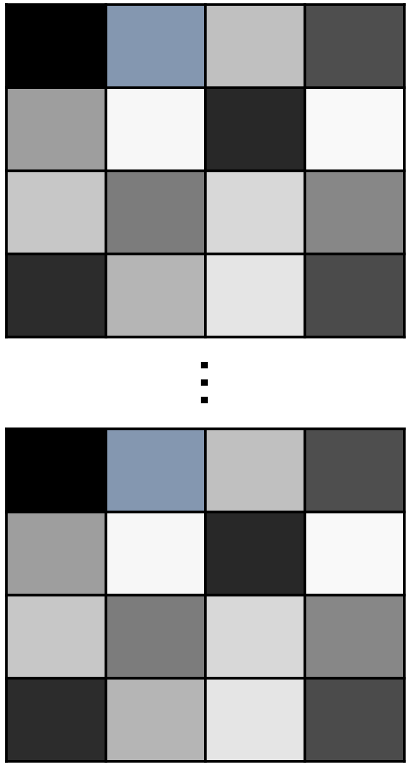
$y_{\text{true}} = A$

$\boxed{y_{\text{pred}} = A}$

Upper bound

Infinite set

Lower bound



$\boxed{y_{\text{pred}} = A}$

$\boxed{y_{\text{pred}} = A}$

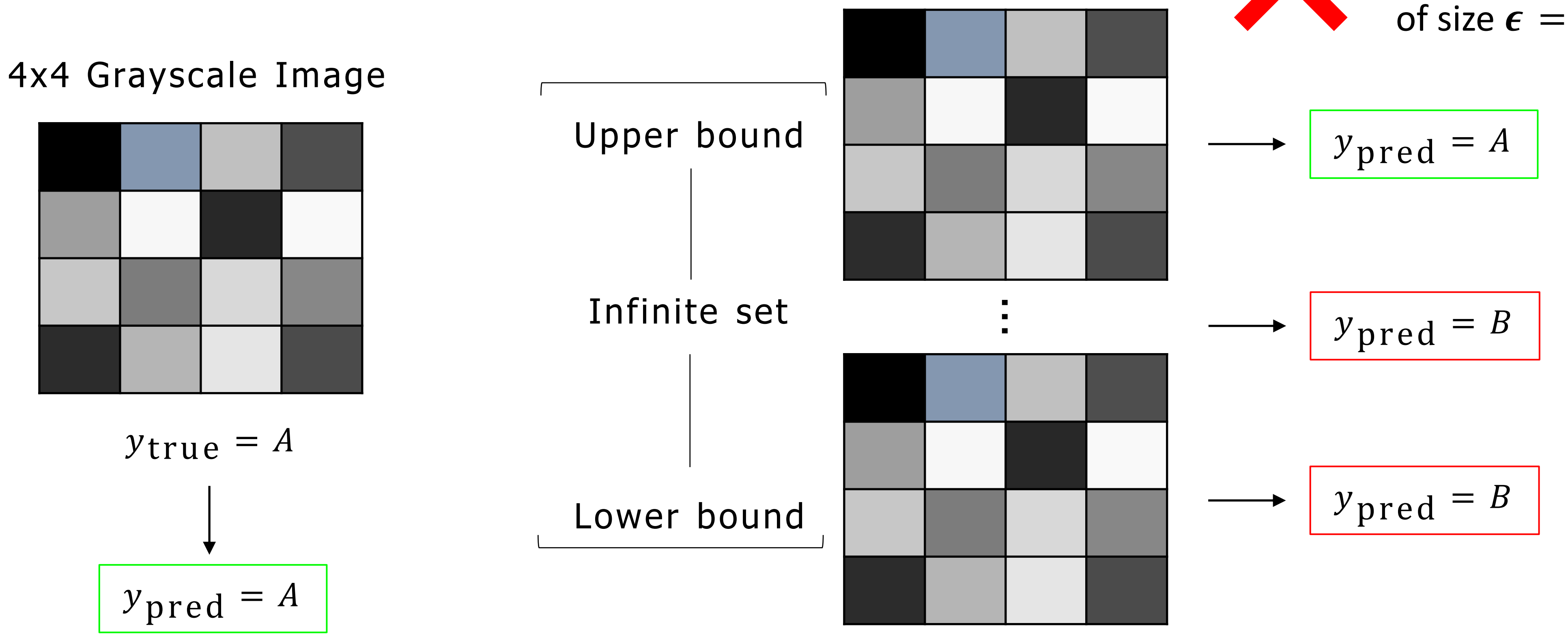
$\boxed{y_{\text{pred}} = A}$

VANDERBILT UNIVERSITY

Malware Byteplot Robustness Example

Robustness Performance: $\epsilon = 2$

NOT Robust to an $\mathcal{L}_\infty$ adversarial perturbation of size $\boldsymbol{\epsilon} = \mathbf{2}$



Preliminary Results: NN Verification

- 200 samples taken from a stratified sampling of the entire BODMAS dataset (43% malicious samples)
- 3 levels of difficulty (data type and size of perturbation)

Benchmark Level	**Perturbation Data Type**	**Perturbation Size(ϵ^*)**
Level 1	Continuous	0.01
Level 2	Continuous and Discrete	0.025
Level 3	All	0.001

VANDERBILT UNIVERSITY

Preliminary Results: NN Verification

- ϵ^* = 0.1%
- Feature data type = *continuous*

Sample 1

	Feature 1 (binary)	Feature 2 (Continuous)	Feature 3 (Discrete)	Feature 4 (Discrete)	Feature 5 (Discrete)
Range	[0, 1]	[3, 567]	[4, 22]	[1, 1000]	[-5, 5]
ϵ	--				

Preliminary Results: NN Verification

- ϵ^* = 0.1%
- Feature data type = *continuous*

Sample 1

	Feature 1 (binary)	Feature 2 (Continuous)	Feature 3 (Discrete)	Feature 4 (Discrete)	Feature 5 (Discrete)
Range	[0, 1]	[3, 567]	[4, 22]	[1, 1000]	[-5, 5]
ϵ	--	± 0.56			

$(567 - 3) * 0.1\% = 0.56$

Preliminary Results: NN Verification

- ϵ^* = 0.1%
- Feature data type = *continuous*

Sample 1

	Feature 1 (binary)	**Feature 2 (Continuous)**	**Feature 3 (Discrete)**	**Feature 4 (Discrete)**	**Feature 5 (Discrete)**
Range	[0, 1]	[3, 567]	[4, 22]	[1, 1000]	[-5, 5]
ϵ	--	± 0.56	--		

VANDERBILT UNIVERSITY

Preliminary Results: NN Verification

- ϵ^* = 0.1%
- Feature data type = *continuous*

Sample 1

	Feature 1 (binary)	Feature 2 (Continuous)	Feature 3 (Discrete)	Feature 4 (Discrete)	Feature 5 (Discrete)
Range	[0, 1]	[3, 567]	[4, 22]	[1, 1000]	[-5, 5]
ϵ	--	± 0.56	--	--	

VANDERBILT UNIVERSITY

Preliminary Results: NN Verification

- ϵ^* = 0.1%
- Feature data type = *continuous*

Sample 1

	Feature 1 (binary)	Feature 2 (Continuous)	Feature 3 (Discrete)	Feature 4 (Discrete)	Feature 5 (Discrete)
Range	[0, 1]	[3, 567]	[4, 22]	[1, 1000]	[-5, 5]
ϵ	--	± 0.56	--	--	--

VANDERBILT UNIVERSITY

Preliminary Results: NN Verification

1. Train a neural network on the BODMAS dataset
    - Input layer: 2381 nodes
    - Hidden layer: 32 nodes
    - Output layer: 2 (binary classifier – malware or benign)

Metric	**Value**
Accuracy	1.0
Precision	0.99
Recall	1.0
F1	1.0

VANDERBILT UNIVERSITY

Preliminary Results: NN Verification

1. Train a neural network on the BODMAS dataset
2. Verify model using on level 2 feature benchmark **using Neural Network Verification (NNV) tool in MATLAB**
   - Continuous & Discrete
   - $\epsilon^* = 0.025$

Result = 103/200 (~50%) samples successfully verified

Metric	Value
Accuracy	1.0
Precision	0.99
Recall	1.0
F1	1.0

Classifier is not as robust as we would hope based on evaluation metrics

VANDERBILT UNIVERSITY

NNV

Summary

- Malware samples are too voluminous for scalable analysis
- Automated analysis can be thwarted by perturbations and evasivness

- Semantics-aware malware augmentation can improve low-resource malware classifiers and provide hard samples for verification

- Neural network verification can be used to measure robustness against perturbation of malware samples

Kevin Leach (kevin.leach@vanderbilt.edu) Taylor Johnson (taylor.johnson@vanderbilt.edu)

VANDERBILT UNIVERSITY