Immuno-mimetic Deep Neural Networks

Alfred Hero

University of Michigan - Ann Arbor

April 1, 2022

Principal references

1. Wang, R., Chen, T., Lindsly, S., Stansbury, C., Rajapakse, I., and Hero, A. (2021). Immuno-mimetic Deep Neural Networks (Immuno-Net), International Conference on Machine Learning, Workshop on Computational Biology, 2021.
2. Wang, R, Chen, T, Lindley, S, Stansbury, C, Rehemtulla, A, Rajapakse, I, and Hero, A, "RAILS: A robust immune- inspired learning system," IEEE Access, Mar. 2022. doi:10.1109/TIT.2022.3151719

   Code: `https://github.com/wangren09/RAILS`

Acknowledgements

Students and collaborators on this project

- Ren Wang, Post-doc/Instructor UM
- Indika Rajapakse, Associate Professor UM
- Cooper Stansbury, Doctoral Student UM
- Tianqi Chen, Doctoral Student UT Austin
- Steven Lindley, Technical Staff Mathworks
- Dogyoon Song, Post-doc UM

Research sponsors

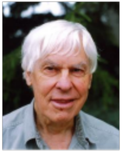
- ARO: MURI: Multiscale biofilm data-model integration and experimental design
- DARPA: Guaranteeing AI Robustness Against Deception (GARD)

DARPA GARD Team


Indika Rajapakse
Michigan


Alfred Hero
Michigan


Maya Gupta
Didero


Steve Smale
UC Berkeley


Alnawaz Rehemtulla
Michigan


Lindsey Muir
Michigan

Mathematics

Biology


Stephen Lindsly
Michigan


Tianqi Chen
Michigan

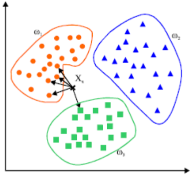

Ren Wang
Michigan


Walter Meixner
Michigan

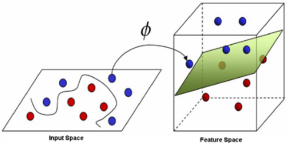

Siva Jeyarajan
Michigan


Charles Ryan
Michigan

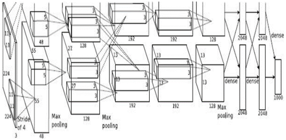
Machine learning vulnerabilities



kNN classifier

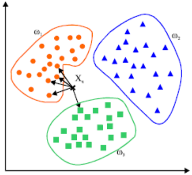


SVM classifier

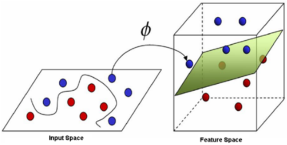


CNN classifier

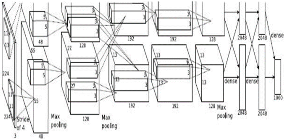
Machine learning vulnerabilities



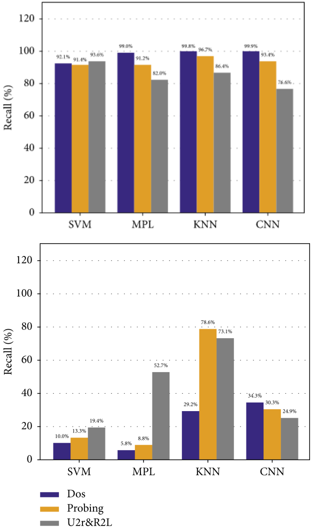
kNN classifier



SVM classifier



CNN classifier



Clean inputs(Top). Adversarial inputs(Bottom)

Guo, *et al*, "A Black-Box Attack Method against Machine-Learning-Based Anomaly Network Flow Detection Models," *Security and Communication Networks*, 2021

Threat models

Attacks are characterized along various axes

- Attack during testing (evasion) vs attack during training (poisoning)
- Targeted vs untargeted attacks
- Information available to attacker on classifier algorithm
- Attack strength and perceptual saliency
- Number of steps: #attacker-evaluations of classifier function

Threat models

Attacks are characterized along various axes

- Attack during testing (evasion) vs attack during training (poisoning)
- Targeted vs untargeted attacks
- Information available to attacker on classifier algorithm
- Attack strength and perceptual saliency
- Number of steps: #attacker-evaluations of classifier function

White box evasion attacks: attacker has knowledge of classifier function

- Projected Gradient Descent (PGD): optimize attack wrt misclassification criterion
- Fast Gradient Sign Method (FGSM): a faster (approximate) PGD
- Auto-Attack: a multi-level attack

Threat models

Attacks are characterized along various axes

- Attack during testing (evasion) vs attack during training (poisoning)
- Targeted vs untargeted attacks
- Information available to attacker on classifier algorithm
- Attack strength and perceptual saliency
- Number of steps: #attacker-evaluations of classifier function

White box evasion attacks: attacker has knowledge of classifier function

- Projected Gradient Descent (PGD): optimize attack wrt misclassification criterion
- Fast Gradient Sign Method (FGSM): a faster (approximate) PGD
- Auto-Attack: a multi-level attack

Black box evasion attacks: attacker has no knowledge of classifier function

- Gradient-free attack, e.g., using BFGS optimizer
- Square attack
- Boundary attack
- Adversarial patch attack

The targeted ℓ_p adversarial attack

Labeled training data $\{(\mathbf{x}_j, y_j)\}_{j=1}^n$

- Features: $\mathbf{x}_j \in [0,1]^d$
- Class labels: $y_j \in [C] \stackrel{\text{def}}{=} \{1, \ldots, C\}$

The targeted ℓ_p adversarial attack

Labeled training data $\{(\mathbf{x}_j, y_j)\}_{j=1}^n$

- Features: $\mathbf{x}_j \in [0,1]^d$
- Class labels: $y_j \in [C] \stackrel{\text{def}}{=} \{1, \dots, C\}$

Classifier optimized over training data

- Classifier function: $c_{\boldsymbol{\theta}} : [0,1]^d \to [C]$
- Classifier tuning parameters: $\boldsymbol{\theta} \in \mathbb{R}^q$
- Loss function: $l : [C] \times [C] \to \mathbb{R}$
- Fitting criterion: $\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, $L(\boldsymbol{\theta}) = \sum_{j=1}^n l(c_{\boldsymbol{\theta}}(\mathbf{x}_j), y_j)$

The targeted ℓ_p adversarial attack

Labeled training data $\{(\mathbf{x}_j, y_j)\}_{j=1}^n$

- Features: $\mathbf{x}_j \in [0,1]^d$
- Class labels: $y_j \in [C] \stackrel{\text{def}}{=} \{1, \ldots, C\}$

Classifier optimized over training data

- Classifier function: $c_{\boldsymbol{\theta}} : [0,1]^d \to [C]$
- Classifier tuning parameters: $\boldsymbol{\theta} \in \mathbb{R}^q$
- Loss function: $l : [C] \times [C] \to \mathbb{R}$
- Fitting criterion: $\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, $L(\boldsymbol{\theta}) = \sum_{j=1}^n l(c_{\boldsymbol{\theta}}(\mathbf{x}_j), y_j)$

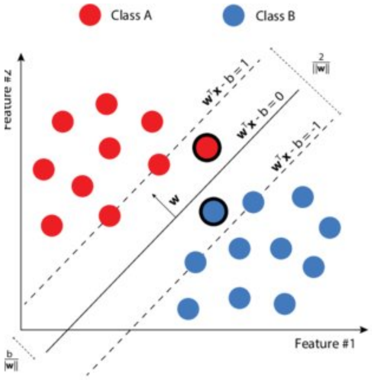
Targeted ℓ_p adversarial attack on classifer

- Try to perturb $\mathbf{x}$ towards target class t $\Rightarrow c_{\boldsymbol{\theta}}(\mathbf{x} + \boldsymbol{\delta}) = t$, where

$$
\begin{aligned}
\boldsymbol{\delta} &= \operatorname{amax}_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_p + \lambda f_{\boldsymbol{\theta}}(\mathbf{x} + \boldsymbol{\delta}) \\
& \quad \text{s.t. } \mathbf{x} + \boldsymbol{\delta} \in [0,1]^d
\end{aligned}
$$

- $f_{\boldsymbol{\theta}} : [0,1]^d \to \mathbb{R}$: $c_{\boldsymbol{\theta}}(\mathbf{u}) = t$ iff $f_{\boldsymbol{\theta}}(\mathbf{u}) < 0$
- $\epsilon = \|\delta\|_p$ is the attack strength

Ex: Binary SVM classification ($\boldsymbol{\theta} = \mathbf{w}$)

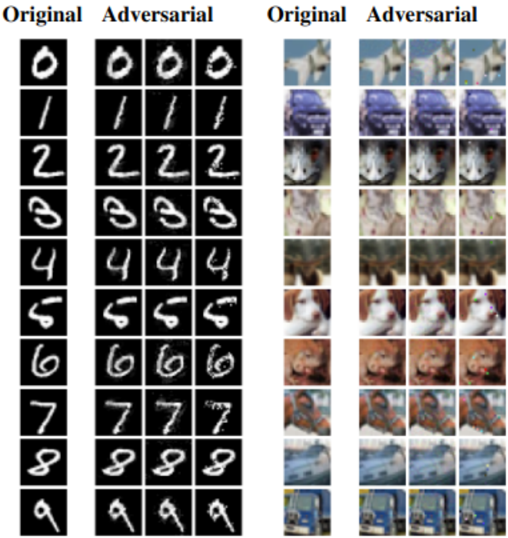


$$
\begin{aligned}
f_{\boldsymbol{\theta}}(\mathbf{x}) &= \mathbf{w}^T\mathbf{x} - b, \\
c_{\boldsymbol{\theta}}(\mathbf{x}) &= \operatorname{sign}(f_{\boldsymbol{\theta}})
\end{aligned}
$$

CNN trained on MNIST has particularly vulnerable decision regions

Rodrigues *et al*. Image-based visualization of classifier decision boundaries." IEEE Conf. Graphics, Patterns and Images, 2018.

Illustration: ℓ_p adversarial attack ($p = 2, \infty, 0$)



Carlini, N. and Wagner, D., (2017). Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy, pp. 39-57.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect

[1] J. Metzen, *et al*. On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al*. Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al*. Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al*. Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al*. Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect
- Minimax robust training[2]
  - ⇒ Can be overly conservative, reducing clean accuracy

[1] J. Metzen, *et al*. On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al*. Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al*. Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al*. Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al*. Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect
- Minimax robust training[2]
  - ⇒ Can be overly conservative, reducing clean accuracy
- Random smoothing - `smoothing`[3]
  - ⇒ Additive isotropically random noise is image-agnostic

---

[1] J. Metzen, *et al*. On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al*. Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al*. Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al*. Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al*. Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect
- Minimax robust training[2]
  - ⇒ Can be overly conservative, reducing clean accuracy
- Random smoothing - `smoothing`[3]
  - ⇒ Additive isotropically random noise is image-agnostic
- Data augmentation - `CutMix, MixUp`[4]
  - ⇒ Does not adapt over attack horizon

---

[1] J. Metzen, *et al*. On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al*. Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al*. Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al*. Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al*. Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect
- Minimax robust training[2]
  - ⇒ Can be overly conservative, reducing clean accuracy
- Random smoothing - `smoothing`[3]
  - ⇒ Additive isotropically random noise is image-agnostic
- Data augmentation - `CutMix, MixUp`[4]
  - ⇒ Does not adapt over attack horizon
- Dimensionality reduction and projection - `STL`[5]
  - ⇒ Sparse transformation layer projection can distort clean inputs

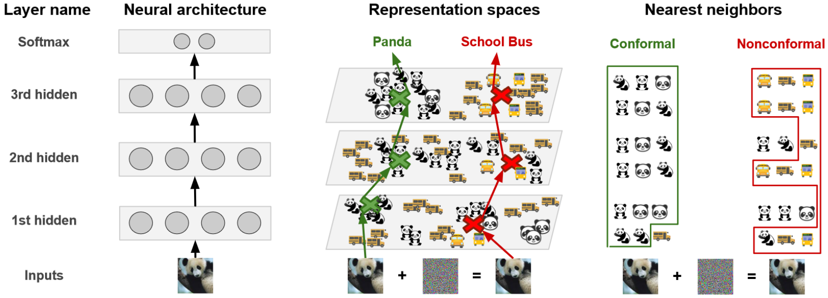
[1] J. Metzen, *et al.* On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al.* Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al.* Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al.* Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al.* Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

Adversarial defense strategies

- Adversarial detection - `AD` [1]
  - ⇒ Useful for sensing an attack but not for mitigating its effect
- Minimax robust training[2]
  - ⇒ Can be overly conservative, reducing clean accuracy
- Random smoothing - `smoothing`[3]
  - ⇒ Additive isotropically random noise is image-agnostic
- Data augmentation - `CutMix, MixUp`[4]
  - ⇒ Does not adapt over attack horizon
- Dimensionality reduction and projection - `STL`[5]
  - ⇒ Sparse transformation layer projection can distort clean inputs
- Deep adversarial learning networks - `DkNN`[6]
  - ⇒ geometrization by kNN's at each layer is limited to training samples

[1] J. Metzen, *et al*. On detecting adversarial perturbations. ICLR 2017.
[2] H. Zhang *et al*. Theoretically principled trade-off between robustness & accuracy. ICML 2019.
[3] J. Cohen *et al*. Certified adversarial robustness via randomized smoothing. ICML 2019
[4] Rebuffi *et al*. Data augmentation can improve robustness. NeurIPS 2021
[5] B. Sun *et al*. Adversarial defense by stratified convolutional sparse coding. CVPR 2019
[6] N. Papernot and P McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. arXiv:1803.04765, 2018.

A deep adversarial learning network: the deep kNN (DkNN) [7] [8] [9]

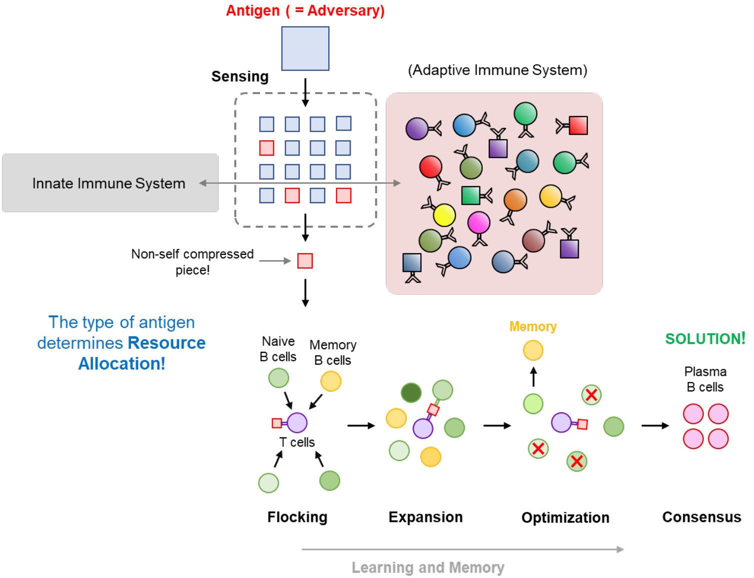


[7] N Papernot and P McDaniel. Deep k-Nearest Neighbors: Towards Confident,Interpretable and Robust Deep Learning. arXiv:1803.04765 2018

[8] C Sitawarin and D Wagner. On the robustness of deep k-nearest neighbors. In 2019 IEEE Security and Privacy Workshops (SPW), pp. 1-7. 2019

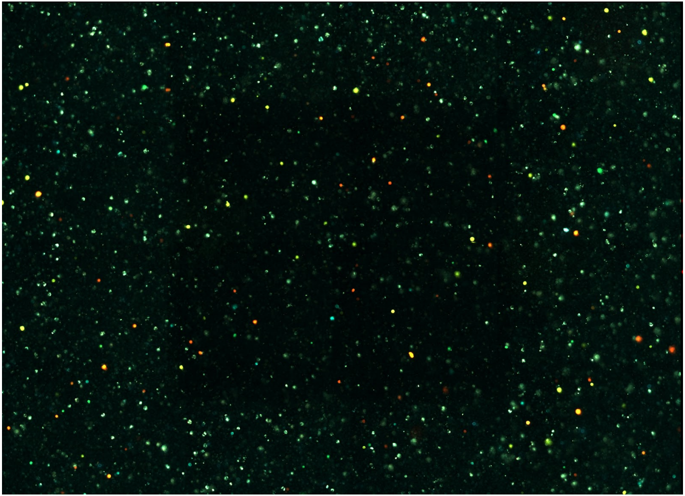
[9] C Sitawarin and D Wagner. Minimum-Norm Adversarial Examples on KNN and KNN-Based Models. arXiv preprint arXiv:2003.06559 (2020)

The natural mammalian immune system



The type of antigen determines **Resource Allocation!**

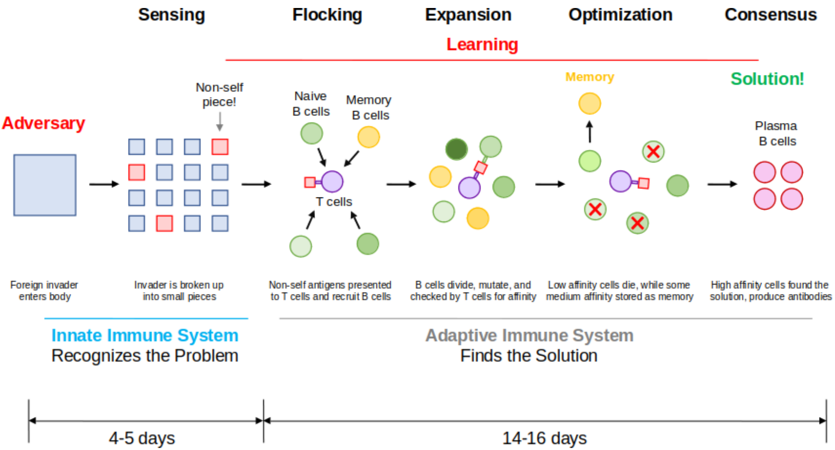
The natural mammalian adaptive immune system



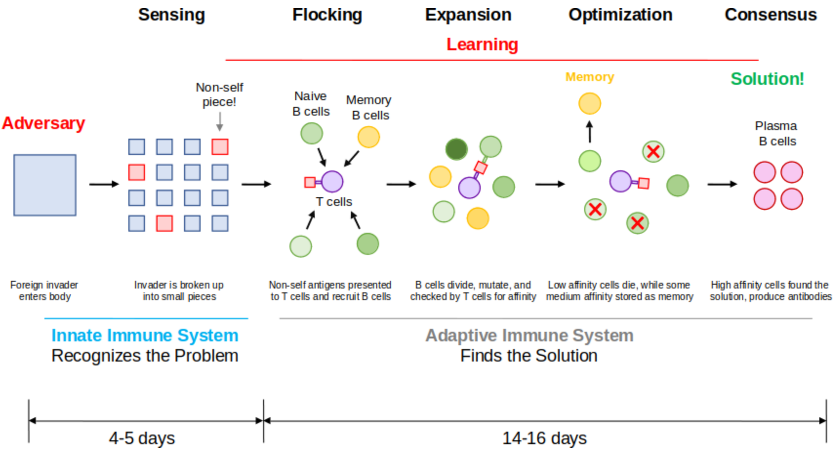
Microscopy image of proliferation of B-cells collected from mouse spleen. Colors denote different levels of B-cell receptor affinity to antigen.

Image credit: Walter Meixner, Rajapakse Lab, University of Michigan, 2022.

The natural mammalian immune system



The natural mammalian immune system



Proposal: Robust adversarial immune-inspired learning system (RAILS): a DNN adversarial defense method emulating mammalian immune system.

Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

Natural immune system and RAILS emulation

	Immune System	RAILS
Sensing	Classify between self and non-self antigens	Classify between non-adversarial and adversarial inputs using confidence scores
Flocking	Non-self antigens are presented to T cells, recruit highest affinity naïve B cells	Find the nearest neighbors that have the highest initial affinity score to the input data
Expansion	Naïve B cells divide and mutate to generate initial diversity	Generate new examples from the nearest neighbors through mutation and crossover and calculate each example's affinity score to the input
Optimization	Affinity is maximized through selection by T cells for affinity. Memory B cells are saved and Plasma B cells are created	Select generated examples with high-affinity scores to be Plasma data, and examples with moderate-affinity scores saved as Memory data
Consensus	Antigen is recognized by majority voting, producing high affinity B cells	Plasma data use majority voting for prediction

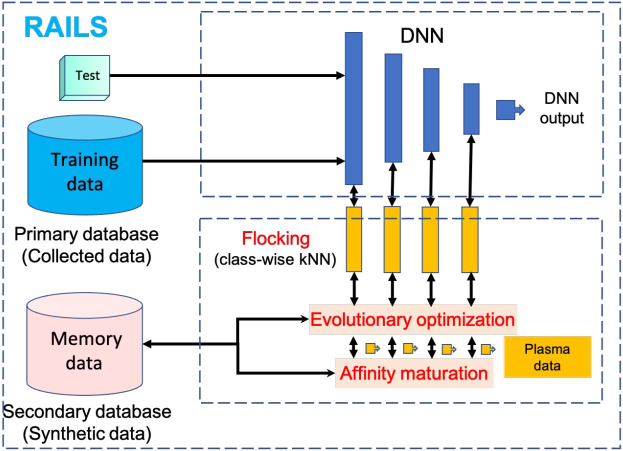
Natural immune system and RAILS emulation

	Immune System	RAILS
Sensing	Classify between self and non-self antigens	Classify between non-adversarial and adversarial inputs using confidence scores
Flocking	Non-self antigens are presented to T cells, recruit highest affinity naïve B cells	Find the nearest neighbors that have the highest initial affinity score to the input data
Expansion	Naïve B cells divide and mutate to generate initial diversity	Generate new examples from the nearest neighbors through mutation and crossover and calculate each example's affinity score to the input
Optimization	Affinity is maximized through selection by T cells for affinity. Memory B cells are saved and Plasma B cells are created	Select generated examples with high-affinity scores to be Plasma data, and examples with moderate-affinity scores saved as Memory data
Consensus	Antigen is recognized by majority voting, producing high affinity B cells	Plasma data use majority voting for prediction

Emulation occurs in a continuous loop, spawning new memory and plasma data as potentially adversarial antigens **x** are sensed at input

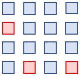
Immuno-Net: RAILS applied to a DNN

Robust adversarial immune learning system



Wang *et al.* RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

Immuno-Net: Sensing stage



Sensing: detect degree of unclassifiability of an input $\mathbf{x}$

$\Rightarrow$ Unclassifiability is measured using *confidence score* over L layers of DNN

$$\text{score}(\mathbf{x}) = \sum_{l=1}^{L} \alpha_l \text{score}_l(\mathbf{x}), \qquad \text{(average cross-entropy)}$$

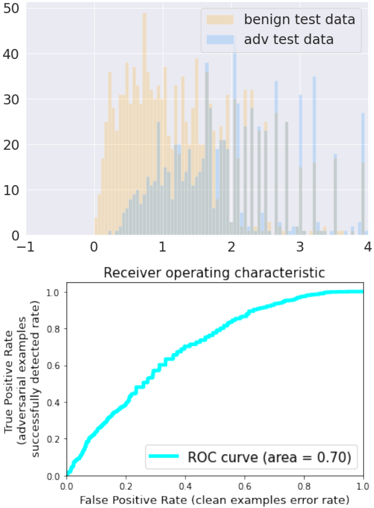
where $\{\alpha_l\}$ are convex combination weights on simplex $\sum_{l=1}^{L} \alpha_l = 1, \;\; \alpha_l \geq 0$ and score for l-th layer of DNN is defined as

$$\text{score}_l(\mathbf{x}) = -\sum_{c=1}^{C} F_c(\mathbf{x}) \log r_c(\mathbf{x})$$

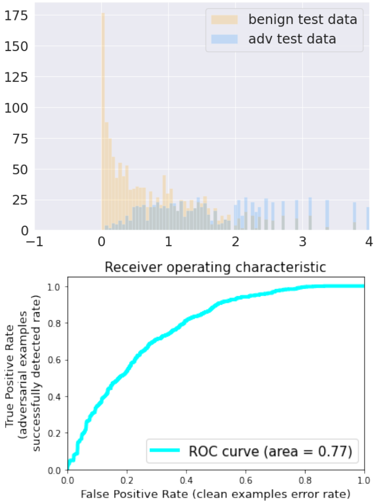
with

- $F_c(\mathbf{x})$ a DNN prediction score that label of $\mathbf{x}$ is in c-th class (logistic output of final layer)
- $r_c(\mathbf{x})$ the proportion of k-NN's of $\mathbf{x}$ having class c labels in training set
- k-NN's computed relative to a distance or affinity measure $A(\mathbf{x}, \mathbf{x}')$

Sensing illustration (RAILS for CIFAR-10)

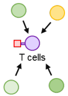


Layer 2



Layer 3

Immuno-Net: Flocking stage



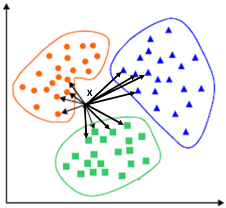
Flocking: find the k-NNs of $\mathbf{x}$ among $\{\mathbf{x}_j\}_{j:y_j=c}$ in each class $c \in [C]$

$\Rightarrow$ results in sets of k-NNs at each layer $l \in [L]$ for each class $c \in [C]$

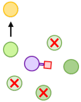
$$\mathcal{N}_{k,l,c}(\mathbf{x}) = \{\mathbf{x}_{c,j_i} : i = 1, \ldots, k\}$$

where $A(\mathbf{x}, \mathbf{x}_j)$ are rank ordered affinity scores (possibly layer dependent) over the $n^c = |\{\mathbf{x}_j\}_{j:y_j=c}|$ instances in class c:

$$A(\mathbf{x}, \mathbf{x}_{c,j_1}) \geq \ldots \geq A(\mathbf{x}, \mathbf{x}_{c,j_{n^c}})$$



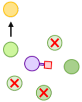
Immuno-Net: Expansion and optimization stage



Expansion and optimization: From the k-NN's of $\mathbf{x}$ synthesize *B-cells*

$\Rightarrow$ B-cells synthesized using *evolutionary optimization* within each class $c \in [C]$

Immuno-Net: Expansion and optimization stage

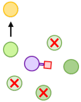


Expansion and optimization: From the k-NN's of $\mathbf{x}$ synthesize *B-cells*

$\Rightarrow$ B-cells synthesized using *evolutionary optimization* within each class $c \in [C]$

- Gather population from generation g: $\mathbf{X}_c^g = [\mathbf{x}_{c1}, \ldots, \mathbf{x}_{cT}] \in \mathbb{R}^{d \times T}$

Immuno-Net: Expansion and optimization stage



Expansion and optimization: From the k-NN's of $\mathbf{x}$ synthesize *B-cells*

$\Rightarrow$ B-cells synthesized using *evolutionary optimization* within each class $c \in [C]$

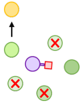
- Gather population from generation g: $\mathbf{X}_c^g = [\mathbf{x}_{c1}, \ldots, \mathbf{x}_{cT}] \in \mathbb{R}^{d \times T}$
- Randomly *select* columns of $\mathbf{X}_c^g$ with affinity-based preference

$$\hat{\mathbf{X}}_c^{g+1} = \mathbf{X}_c^g \mathbf{Z}_c^g, \quad \mathbf{Z}_c^g \in \{0, 1\}^{T \times T}$$

where columns of $\mathbf{Z}^g$ are drawn from $\text{Mult}(1, \mathbf{p})$, $\mathbf{p} = [p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)]$

$$\mathbf{p}(\mathbf{x}_{cj}) = \text{Softmax}(A(\mathbf{x}_{cj}, \mathbf{x})), \; j = 1, \ldots, T.$$

Immuno-Net: Expansion and optimization stage



Expansion and optimization: From the k-NN's of $\mathbf{x}$ synthesize *B-cells*

$\Rightarrow$ B-cells synthesized using *evolutionary optimization* within each class $c \in [C]$

- Gather population from generation g: $\mathbf{X}_c^g = [\mathbf{x}_{c1}, \ldots, \mathbf{x}_{cT}] \in \mathbb{R}^{d \times T}$
- Randomly *select* columns of $\mathbf{X}_c^g$ with affinity-based preference

$$\hat{\mathbf{X}}_c^{g+1} = \mathbf{X}_c^g \mathbf{Z}_c^g, \quad \mathbf{Z}_c^g \in \{0,1\}^{T \times T}$$

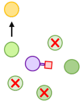
where columns of $\mathbf{Z}^g$ are drawn from $\mathrm{Mult}(1, \mathbf{p})$, $\mathbf{p} = [p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)]$

$$\mathbf{p}(\mathbf{x}_{cj}) = \mathrm{Softmax}(A(\mathbf{x}_{cj}, \mathbf{x})), \; j = 1, \ldots, T.$$

- Generate offspring of each column by crossover *mating* with $\mathbf{x}$

$$\mathbf{x}'_{\mathrm{os}} = \mathit{Crossover}(\mathbf{x}_c, \mathbf{x}'_c) = \begin{cases} \mathbf{x}_c^{(i)} & \text{with prob } \frac{A(f_l;\mathbf{x}_c,\mathbf{x})}{A(\mathbf{x}_c,\mathbf{x})+A(\mathbf{x}'_c,\mathbf{x})}, \\ \mathbf{x}_c^{\prime(i)} & \text{with prob } \frac{A(\mathbf{x}'_c,\mathbf{x})}{A(\mathbf{x}_c,\mathbf{x})+A(\mathbf{x}'_c,\mathbf{x})} \end{cases} \quad \forall i \in [d],$$

Immuno-Net: Expansion and optimization stage



Expansion and optimization: From the k-NN's of $\mathbf{x}$ synthesize *B-cells*

$\Rightarrow$ B-cells synthesized using *evolutionary optimization* within each class $c \in [C]$

- Gather population from generation g: $\mathbf{X}_c^g = [\mathbf{x}_{c1}, \ldots, \mathbf{x}_{cT}] \in \mathbb{R}^{d \times T}$
- Randomly *select* columns of $\mathbf{X}_c^g$ with affinity-based preference

$$\hat{\mathbf{X}}_c^{g+1} = \mathbf{X}_c^g \mathbf{Z}_c^g, \quad \mathbf{Z}_c^g \in \{0,1\}^{T \times T}$$

where columns of $\mathbf{Z}^g$ are drawn from $\text{Mult}(1, \mathbf{p})$, $\mathbf{p} = [p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)]$

$$\mathbf{p}(\mathbf{x}_{cj}) = \text{Softmax}(A(\mathbf{x}_{cj}, \mathbf{x})), \; j = 1, \ldots, T.$$

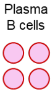
- Generate offspring of each column by crossover *mating* with $\mathbf{x}$

$$\mathbf{x}'_{\text{os}} = \mathit{Crossover}(\mathbf{x}_c, \mathbf{x}'_c) = \begin{cases} \mathbf{x}_c^{(i)} & \text{with prob } \frac{A(f_l; \mathbf{x}_c, \mathbf{x})}{A(\mathbf{x}_c, \mathbf{x}) + A(\mathbf{x}'_c, \mathbf{x})}, \\ \mathbf{x}_c'^{(i)} & \text{with prob } \frac{A(\mathbf{x}'_c, \mathbf{x})}{A(\mathbf{x}_c, \mathbf{x}) + A(\mathbf{x}'_c, \mathbf{x})} \end{cases} \quad \forall i \in [d],$$

- Randomly *mutate* each offspring with mutation probability ρ

$$\mathbf{x}_{\text{os}} = \mathit{Mutation}(\mathbf{x}'_{\text{os}}) = \text{Clip}_{[0,1]}\left(\mathbf{x}'_{\text{os}} + \mathbf{1}_{[\mathit{Bernoulli}(\rho)]} \mathbf{u}([-\delta_{\max}, -\delta_{\min}] \cup [\delta_{\min}, \delta_{\max}])\right)$$

Immuno-Net: Consensus



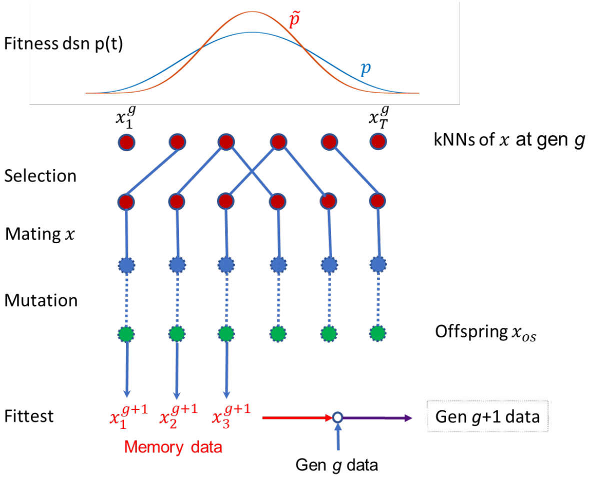
Consensus: classify $\mathbf{x}$ using fittest offspring, and update population

$\Rightarrow$ Stratify offspring $\mathbf{x}_{\text{os}}$ based on affinity to $\mathbf{x}$

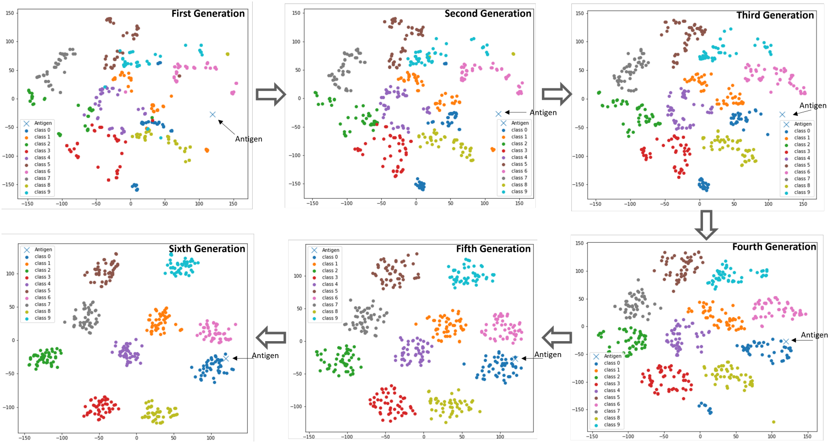
- Rank order affinity scores $A(\mathbf{x}_{\text{os}}, \mathbf{x})$ for all offspring $\mathbf{x}_{\text{os}}$ in all classes $c \in [C]$
- Select top 5% of offspring as *plasma data* for majority vote on $\mathbf{x}$
- Select top 25% of offspring as *memory data* to augment data for next generation.
- Merge memory data into generation g data, resulting in population update

$$\mathbf{X}_c^g \rightarrow \mathbf{X}_c^{g+1}$$

Expansion and optimization graphical representation



Plasma B-cell receptor evolution over 6 generations



Convergence analysis

Questions of interest

- Under what conditions does RAILS converge to an accurate and robust classification of a target $\mathbf{x}$?
- What factors determine speed of convergence?
- What factors determine accuracy?
- What factors determine robustness?

Convergence analysis

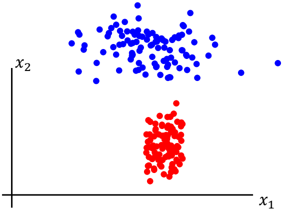
Questions of interest

- Under what conditions does RAILS converge to an accurate and robust classification of a target $\mathbf{x}$?
- What factors determine speed of convergence?
- What factors determine accuracy?
- What factors determine robustness?

We have results for the case that RAILS is applied to the centroid classifier.

Centroid classifier

Collect samples for $C = 2$ classes (red and blue)

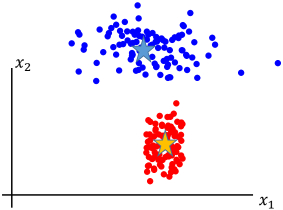


Tibshirani, Hastie, Narasimhan, Chu (2002). "Diagnosis of multiple cancer types by shrunken centroids of gene expression". Proceedings of the National Academy of Sciences. **99** (10): 6567–6572.

Centroid classifier

Collect samples for $C = 2$ classes (red and blue)

Compute centroids of each class (Stars)



$$\mathbf{x}_{c_+} = \frac{1}{n_+} \sum_{j:y_j=1} \mathbf{x}_j$$

$$\mathbf{x}_{c_-} = \frac{1}{n_-} \sum_{j:y_j=-1} \mathbf{x}_j$$

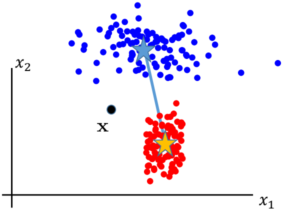
Tibshirani, Hastie, Narasimhan, Chu (2002). "Diagnosis of multiple cancer types by shrunken centroids of gene expression". Proceedings of the National Academy of Sciences. **99** (10): 6567–6572.

Centroid classifier

Collect samples for $C = 2$ classes (red and blue)

Compute centroids of each class (Stars)

Implement minimum distance classifier to classify $\mathbf{x}$



$$\hat{y}(\mathbf{x}) = \operatorname{sign}\left(\left\|\mathbf{x} - \mathbf{x}_{c_-}\right\| - \left\|\mathbf{x} - \mathbf{x}_{c_+}\right\|\right)$$

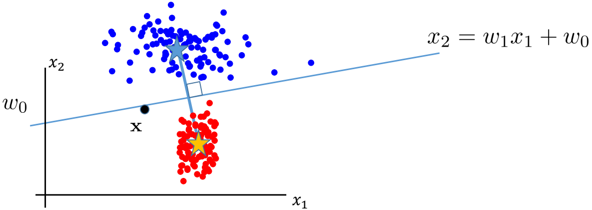
Tibshirani, Hastie, Narasimhan, Chu (2002). "Diagnosis of multiple cancer types by shrunken centroids of gene expression". Proceedings of the National Academy of Sciences. **99** (10): 6567–6572.

Centroid classifier

Collect samples for $C = 2$ classes (red and blue)

Compute centroids of each class (Stars)

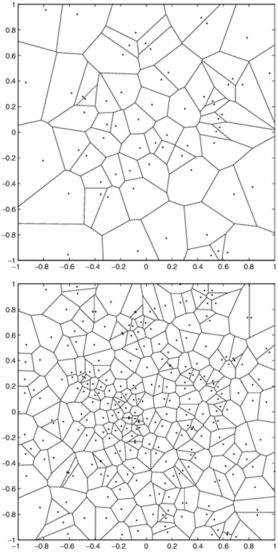
Implement minimum distance classifier to classify $\mathbf{x}$



$$\hat{y}(\mathbf{x}) = \text{sign}\left((\mathbf{x}_{c_+} - \mathbf{x}_{c_-})^T \left(\mathbf{x} - \frac{\mathbf{x}_{c_+} + \mathbf{x}_{c_-}}{2} \right) \right)$$

Tibshirani, Hastie, Narasimhan, Chu (2002). "Diagnosis of multiple cancer types by shrunken centroids of gene expression". Proceedings of the National Academy of Sciences. **99** (10): 6567–6572.

Centroid classifier for $C > 2$ classes



Convergence of RAILS centroid classifier

- Discrete alphabet inputs&centroids, $\mathbf{x} = [x_1, \ldots, x_d]$, $x_i \in \{j/\kappa\}_{j=1}^{\kappa}$
- $\mathbf{e}_c$ centroid of correct-class $y_i = c$ of input $\mathbf{x}_i$
- N: number of generations of B-cell expansion/optimization
- T: number of offspring per generation
- $\mu \in [0, 1]$: mutation probability of an allele, uniform over $[\kappa]$
- $H(\mathbf{e}_c)$: Hamming erroneous class distance of $\mathbf{x}_i$: [1] $\min_{i:y_i \neq c} d_H(\mathbf{e}_c, \mathbf{x}_i)$

Theorem (Capture time bound)

Assume that μ is sufficiently small such that $(1-\mu)/(\mu/(\kappa-1)) \geq 1$. Let $\delta \in (0, 1)$. Define $N^*(\delta)$, the number of generations required for the RAILS centroid classifier to produce an offspring in correct class $c \in \{1, \ldots, C\}$ of an input $\mathbf{x} \in \mathbb{R}^d$, with probability at least δ. Then the number of RAILS generations, with T draws per generation, satisfies the bound

$$N^*(\delta) \leq \max \left\{ \frac{1}{T} \frac{\ln\left(\frac{1}{1-\delta}\right)}{\ln\left(\frac{1}{1-(H(\mathbf{e}_c)+1)(\mu/(\kappa-1))^d}\right)}, 1 \right\}.$$

[1] $d_H(\mathbf{e}, \mathbf{x})$ is the number of alleles in $\mathbf{e}$ and $\mathbf{x}$ that disagree.

Convergence for equipartitioned classes

- Equipartioned class assumption: $\min_i \|\mathbf{e}_c - \mathbf{e}_i\| = C^{-1/d} b_1$
- Continuum limit: $\kappa \to \infty$, $\mu \to 0$ and $\mu/\kappa \to \rho$

Hamming/Euclidean distance relation for discrete alphabet $\mathbf{u}, \mathbf{v} \in [0,1]^d$:

$$d_H(\mathbf{u},\mathbf{v}) \leq \|\mathbf{u}-\mathbf{v}\|^2 \leq \frac{1}{\kappa^2} d_H(\mathbf{u},\mathbf{v})$$

results in capture time bound

$$N^*(\delta) \leq \max\left\{\frac{1}{T}\frac{\ln\left(\frac{1}{1-\delta}\right)}{\ln\left(\frac{1}{1-(1+C^{-2/d}b_1^2)\rho^d}\right)}, 1\right\}.$$

- For large d, $N^*(\delta)$ increases in $\ln C/d$ and decreases in ρ.
- Bound provides rules for selection of ρ as a function of C and d
- Bound only depends on expansion/optimization via mutation rate ρ

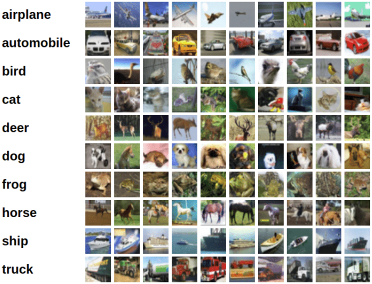
Numerical experiments with image data sets

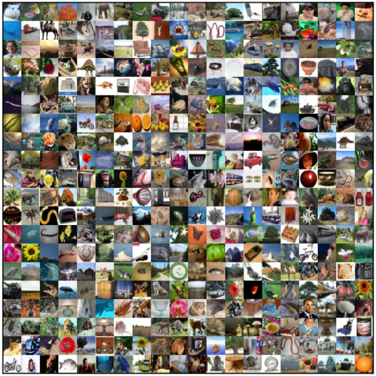
Datasets evaluated: MNIST, CIFAR-10, CIFAR-100, SVHN

- MNIST
  - Number of images in dataset: 70000
  - Number of classes in dataset: 10
  - Number of pixels: 28×28
- CIFAR-10 and CIFAR-100
  - Number of images in dataset: 60000
  - Number of classes in dataset: 10 and 100, respectively.
  - Number of pixels: 32×32
- SVHN
  - Number of images in dataset: 600000
  - Number of classes in dataset: 10
  - Number of pixels: 32×32

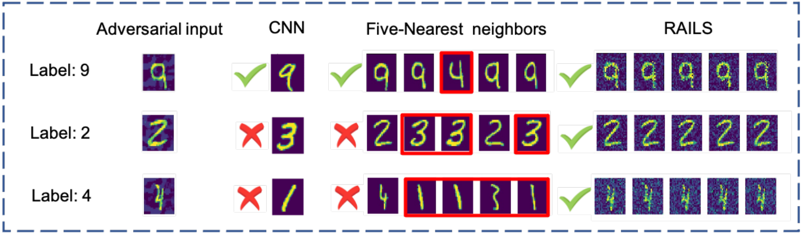
Samples (CW) from MNIST, SVHN, CIFAR-100, CIFAR-10 image data sets

Samples from the MNIST data set





RAILS for MNIST ℓ_∞ PGD attack

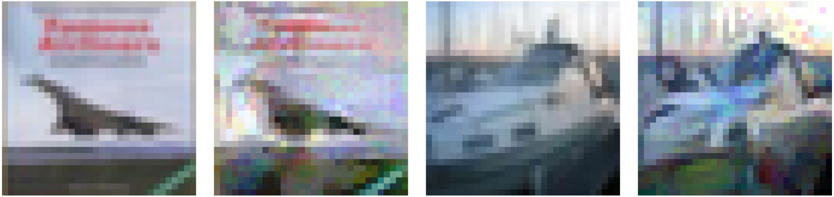


CNN and KNN misclassify digits 2 and 4 while RAILS classifies all 3 correctly

Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

RAILS for CIFAR-10 human perceptible ℓ_∞ PGD attack

Benign1 Adversarial1 Benign2 Adversarial2



Adversarial accuracy comparisons

- RAILS: 33.26%,
- CNN: 0%
- DkNN: 19.53%

Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

RAILS vs DkNN-CNN (MNIST)

Table: **RAILS outperforms DkNN on single layers.** Standard Accuracy (SA)/Robust Accuracy (RA) performance of RAILS versus DkNN in single layer (MNIST).

		Input	Conv1	Conv2
SA	**RAILS**	**97.53%**	**97.77%**	**97.78%**
	DkNN	96.88%	97.4%	97.42%
RA	**RAILS**	**93.78%**	**92.56%**	**89.29%**
($\epsilon = 40$)	DkNN	91.81%	90.84%	88.26%
RA	**RAILS**	**88.83%**	**84.18%**	**73.42%**
($\epsilon = 60$)	DkNN	85.54%	81.01%	69.18%

Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

RAILS improves robust accuracy on across benchmark CV datasets

Table: **RAILS achieves higher robust accuracy (RA) at small cost of standard accuracy (SA)** on MNIST, SVHN and CIFAR-10 as compared to CNN and DkNN.

		SA	RA
MNIST	**RAILS (ours)**	97.95%	**76.67%**
($\epsilon = 60$)	CNN	**99.16%**	1.01%
	DkNN	97.99%	71.05%
SVHN	**RAILS (ours)**	90.62%	**48.26%**
($\epsilon = 8$)	CNN	**94.55%**	1.66%
	DkNN	93.18%	35.7%
CIFAR-10	**RAILS (ours)**	82%	**52.01%**
($\epsilon = 8$)	CNN	**87.26%**	32.57%
	DkNN	86.63%	41.69%

Wang *et al.* RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

RAILS has better adversarial resilience than previous methods

Table: **RAILS achieves higher robust accuracy (RA)** under eight types of attacks as compared to CNN and DkNN (CIFAR-10).

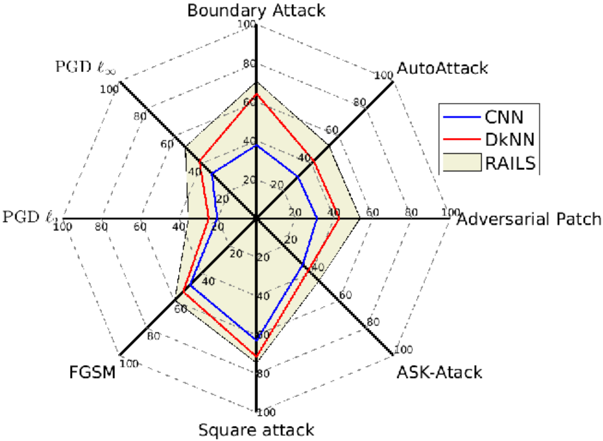
	RAILS	DkNN	CNN
ℓ_∞-PGD ($\epsilon = 8$)	**52.01%**	41.69%	32.57%
ℓ_2-PGD ($\epsilon = 127.5$)	**35.1%**	24.64%	20.3%
FGSM ($\epsilon = 8$)	**59.7%**	53.46%	48.52%
Sq-Attack ($\epsilon = 20$)	**74.5%**	71.3%	53.7%
Boundary Attack (ℓ_2)	**70.6%**	64.2%	37.81%
AutoAttack ($\epsilon = 8$)	**52.84%**	41.77%	30.26%
Adv-P (ratio= 0.1)	**53.5%**	42.7%	31.14%
ASK-Attack ($\epsilon = 8$)	**45.5%**	37.8%	34.21%

Table: **RAILS achieves higher robust accuracy (RA)** than DkNN and CNN on CIFAR-100 under the 3 strongest attacks.

	RAILS	DkNN	CNN
ℓ_∞-PGD ($\epsilon = 8$)	**41.35%**	32.96%	23.7%
AutoAttack ($\epsilon = 8$)	**42.84%**	32.86%	25.63%
Boundary Attack (ℓ_2)	**53.6%**	49.51%	29.1%

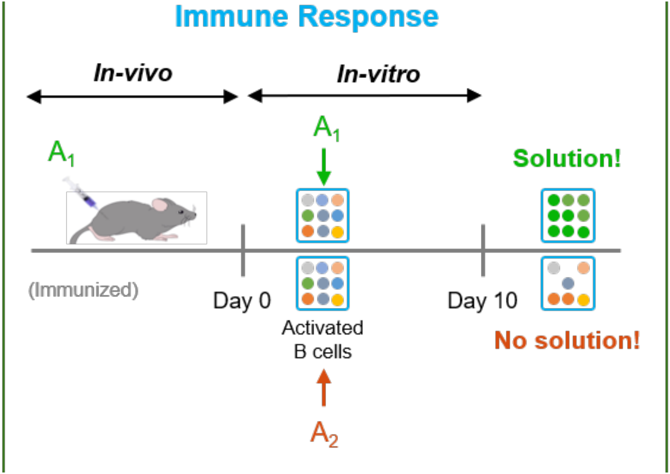
Wang *et al.* RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022

RAILS has better adversarial resilience than previous methods



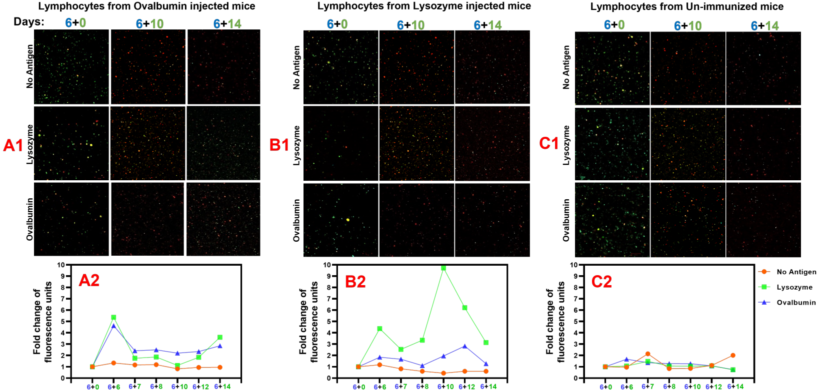
Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

In vitro experiment



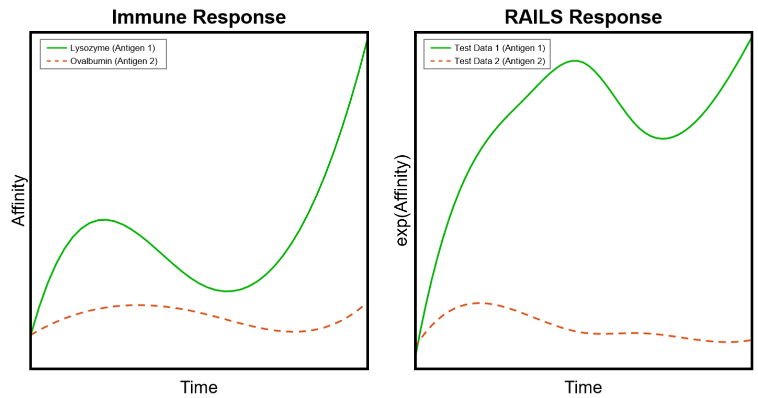
Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

In vitro experimental outcome



Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

Learning curve of RAILS emulates learning curve of immune response



Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

Immune-System vs RAILS: Summary of Correspondences

Table: A one-to-one mapping from the immune system to RAILS.

	Immune System	RAILS
Antigen	A molecule or molecular structure (self/non-self)	Test example (benign/adversarial)
Affinity	The strength of a single bond or interaction between antigen and B-Cell	The negative Euclidean distance between feature maps of input and another data point
Naive B-cells	The B-cells that have been recruited to generate new B-cells	The k-nearest neighbors from each class with highest affinity to the antigen
Plasma B-cells	Newly generated B-cells with top affinity to the antigen	Newly generated examples with top affinity to the input
Memory B-cells	Generated B-cells with moderate-affinity to the antigen	Generated examples with moderate-affinity to the input
	Immune System	**RAILS**
Sensing	Classify between **self** and **non-self** antigens	Classify between **non-adversarial** and **adversarial** inputs using confidence scores
Flocking	Non-self antigens are presented to T cells, recruit **highest affinity naive B-cells**	Find the **nearest neighbors from each class** that have the highest initial affinity score to the input data
Affinity maturation	**Naive B-cells** *divide and mutate* to generate initial diversity. **Affinity is maximized** through selection by T cells for affinity.	Generate new examples from the **nearest neighbors** through *mutation and crossover* and calculate each example's affinity score to the input **Affinity is maximized** through selection.
Consensus	*Memory B-cells* are saved and *Plasma B-cells* are created. Antigen is recognized by **majority voting**, producing *high affinity B-cells*	Select generated examples with **high-affinity scores** to be *Plasma data*, and examples with moderate-affinity scores saved as *Memory data*. *Plasma data* use **majority voting for prediction**

Wang *et al*. RAILS: A Robust adversarial immune-inspired learning system. IEEE Access, Mar 2022.

Summary comments and perspectives

Adaptive immune system emulation for robustifying machine learning

- Robust adaptve immune-inspired learning system (RAILS) emulates
  1. sensing
  2. flocking
  3. clonal expansion
  4. consensus
- RAILS dkNN-CNN: consensus of B-cell affinity maturation at each layer
- RAILS dkNN-CNN: improves resilience to different types of attacks
- RAILS dkNN-CNN: mimics diversity vs selectivity of natural immune system
- RAILS centroid classifier: convergence with high probability established

Summary comments and perspectives

Adaptive immune system emulation for robustifying machine learning

- Robust adaptve immune-inspired learning system (RAILS) emulates
  1. sensing
  2. flocking
  3. clonal expansion
  4. consensus
- RAILS dkNN-CNN: consensus of B-cell affinity maturation at each layer
- RAILS dkNN-CNN: improves resilience to different types of attacks
- RAILS dkNN-CNN: mimics diversity vs selectivity of natural immune system
- RAILS centroid classifier: convergence with high probability established

Some interesting questions

- Immuno-mimetic attackers - dynamic adversarial attack strategies
- DNN autoimmune disease - can RAILS be tricked into "attacking" its own cells?
- In-silico innoculation and boosting - periodic introduction of synthetic attacks