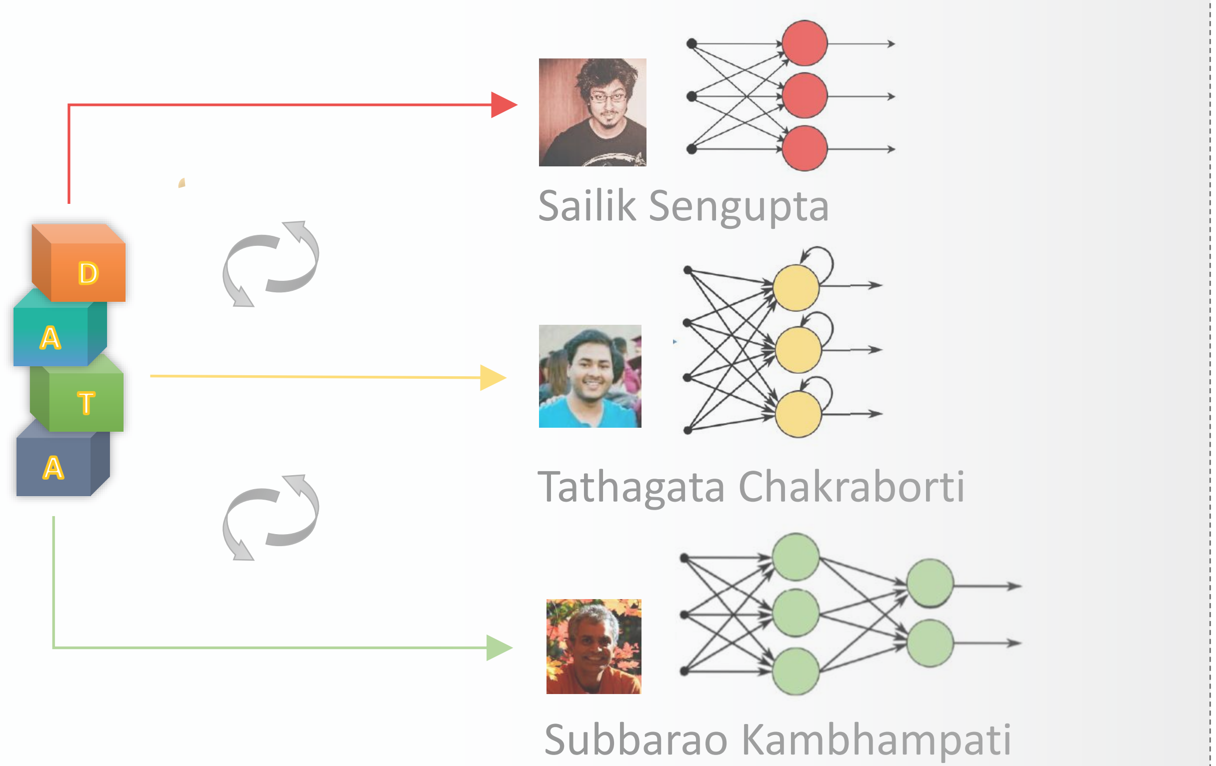
# MTDeep: Boosting the Security of Deep Neural Networks Against Adversarial Attacks with Moving Target Defense

Arizona State University



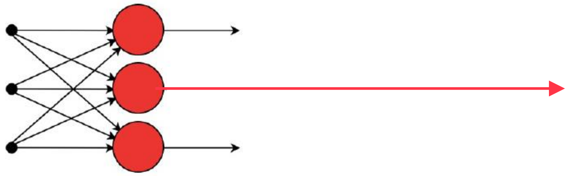

# Supervised Machine Learning Systems for Classification

- Given *i.i.d.* labelled training data $(i, l) \sim D$ learn a classifier that estimates $\Pr(l_j | i)$, where $l_j \in L$ where $L$ is the set of class labels for the input data $i$.

- The classifier guarantees classification accuracy about inputs $i$ drawn from the same distribution $D$.

Picture courtesy:
Szegedy, Christian, et al. "Intriguing properties of neural networks." *arXiv preprint arXiv:1312.6199*(2013).

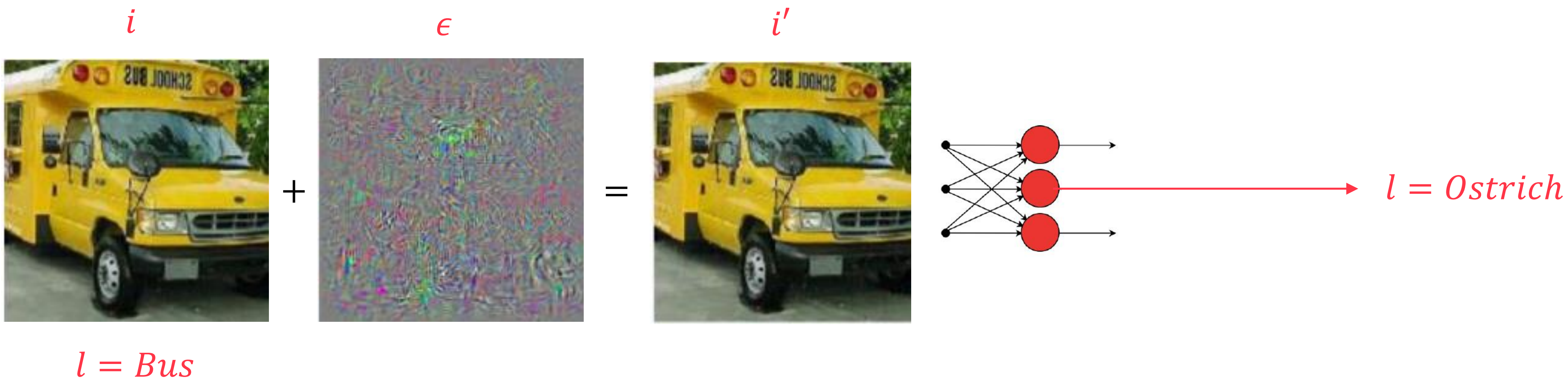

$i$



$l$ = Bus

# Attacks on Deep Neural Networks

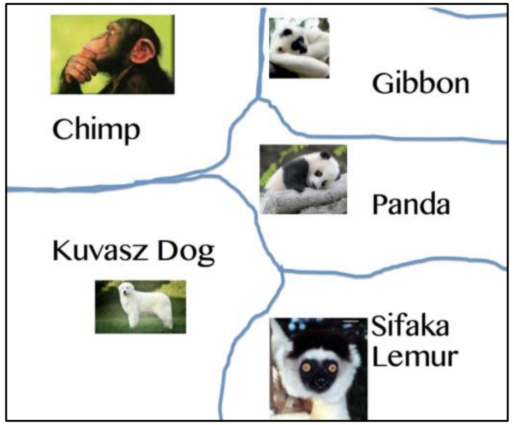
- Classifier misclassifies (i.e. increases *misclassification rate*), human observer cannot detect the noise.

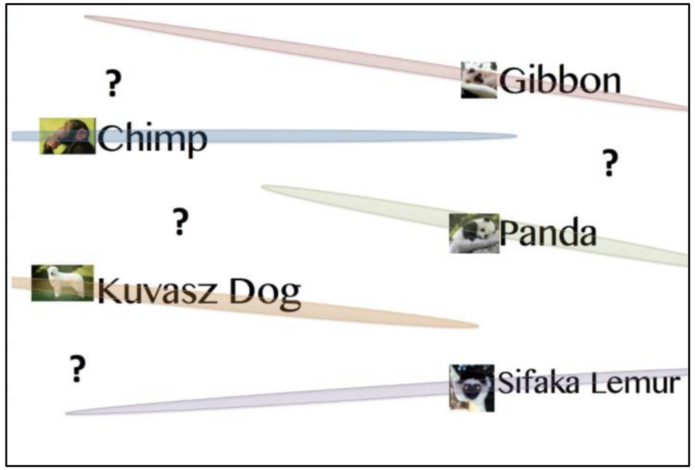



Picture courtesy:
Szegedy, Christian, et al. "Intriguing properties of neural networks." *arXiv preprint arXiv:1312.6199*(2013).

# Attacking Deep Neural Networks (for image classification)

- The input space is huge. For a $28 \times 28$ image with RGB channels, the input space is $28 \times 28 \times (255)^3$.
- The input distribution for classification is small.
- Classifier has high *bias* for regions not seen in the input.
- Exploit this bias to generate adversarial samples.




*Picture courtesy:* Peter Norvig

# Attacks on Deep Neural Networks

- Whitebox attacks – attack for each image, each network
  - FGSM Szegedy et al., 2013, DeepFool Moosavi et al., 2016a, JSMA Papernot et al., 2016a etc.
  - Find features (in an image) that have the most effect on the Loss Function of the classifier. Use these as heuristics to manipulate input.
  - Find out decision boundaries, find perturbation vectors that push it over to the other side.
- Blackbox attacks – attack for each image, each network
  - Train a small substitute DNN using distillation. Design a whitebox attack on this small network. Attack generalizes to blackbox Papernot et al., 2017b
  - Black box attacks without substitute models are also possible Chen et al., 2017
- Universal Perturbation – attack (mostly WB) for a single network
  - Create a noise image that added to any image will make the classifier misclassify it Moosavi et al., 2016b
  - Adversarial Patch – A universal perturbation at a particular place in the image Brown et al., 2017
  - Bad Nets – Backdoor patch introduced by poising a portion of the training data Gu et al., 2017

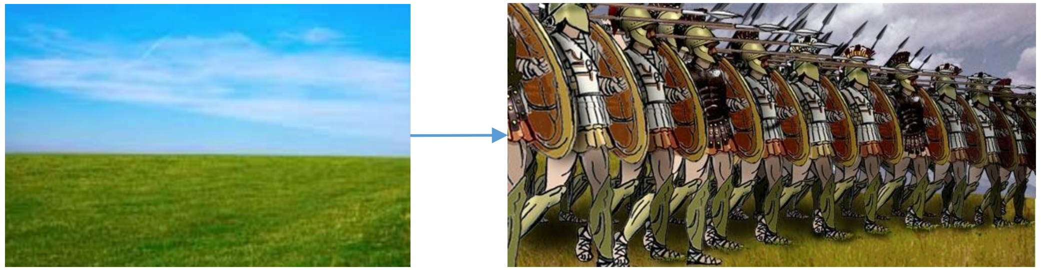
# Defense Against Attacks on Deep Neural Networks

- Train the neural net on the attack distribution (with the correct labels) and the classifier becomes immune to the particular type of adversarial inputs. This is one of the most effective methods!
  - Ensemble adversarial training [Tramer et al., 2017], Stability training [Zheng et al., 2016]
  - Shown to be ineffective against Adv. Universal Perturbation [Moosavi et al., 2016b]

- Other defense mechanisms like defensive distillation [Papernot et al., 2016c], anti-whitening and dimensionality reduction [Bhagoji et al., 2017]
  - Effectiveness is shown against some sets of attack algorithms.
  - These mechanisms are often shown to be ineffective against new computationally expensive attacks [Carlini and Wagner, 2017]

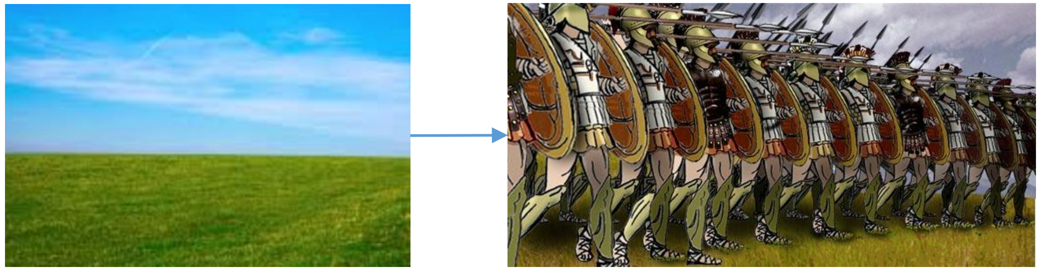
# Defense and Meta-Defense Against Attacks on Deep Neural Networks

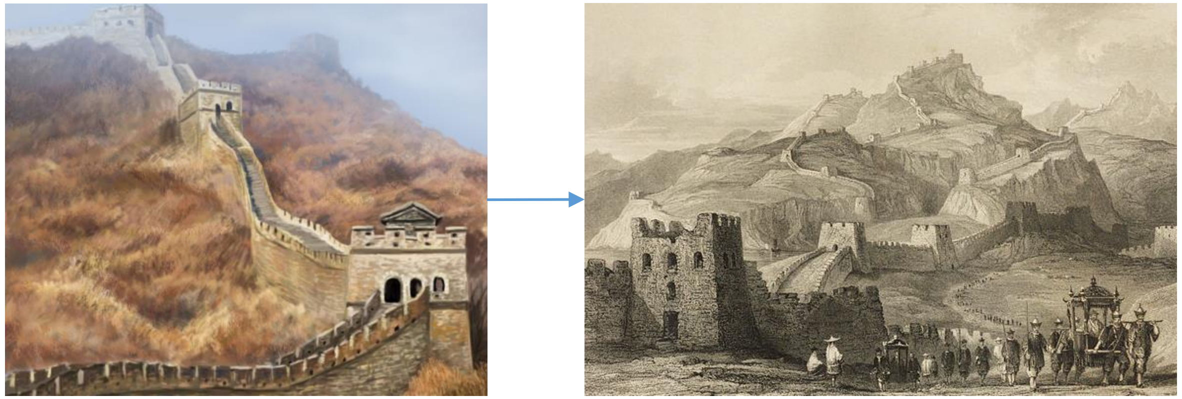
- How about a defense mechanisms that
    - Can provide a first line of defense against attacks previously unseen.



# Defense and Meta-Defense Against Attacks on Deep Neural Networks

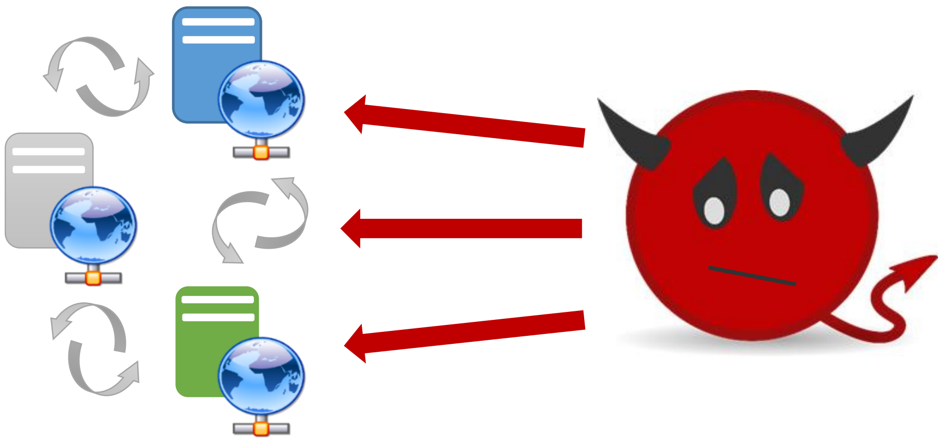
- How about a defense mechanisms that
  - Can provide a first line of defense against attacks previously unseen.
  - Can increase the security with currently available technology for defense.





# Moving Target Defense

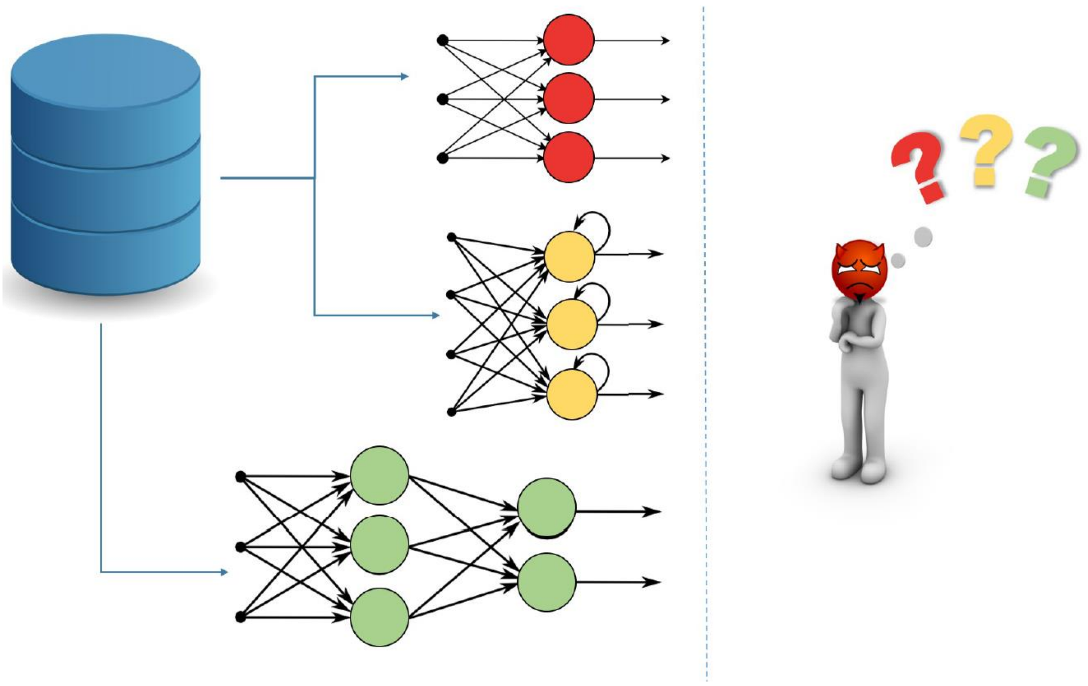
- Cybersecurity uses MTD as a mechanism to ensure that an attack by an attacker is not always successful since the surface being defended in not static.



# Moving Target Defense in Practice

# Moving Target Defense for Deep Neural Networks (MTDeep)

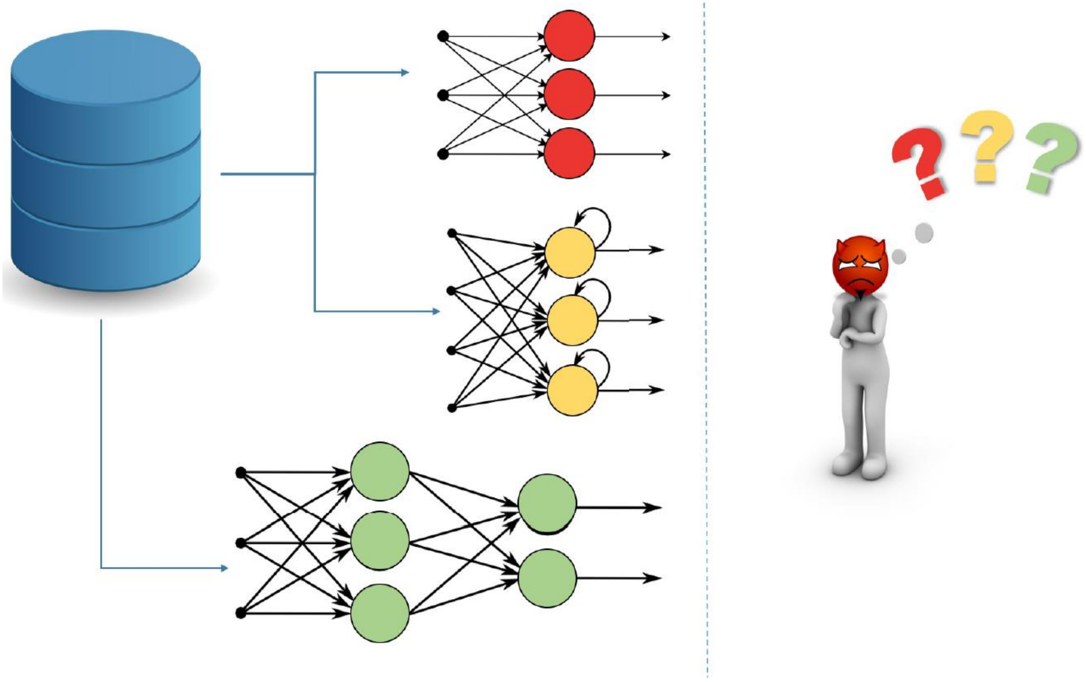
- Let us say we have 3 DNNs (say $N_1, N_2, N_3$) and an attacker designs an noise $\epsilon_1$ for the $N_1$ (and an input image $x$).
- Let us say the attacker gives $x + \epsilon$ as input for classification. If the input is given to the $N_2$ or $N_3$ and the attacker's input is classified correctly, the attack becomes ineffective.



# Moving Target Defense for Deep Neural Networks (MTDeep)

- Let us say we have 3 DNNs (say $N_1, N_2, N_3$) and an attacker designs an noise $\epsilon_1$ for the $N_1$ (and an input image $x$).
- Let us say the attacker gives $x + \epsilon$ as input for classification. If the input is given to the $N_2$ or $N_3$ and the attacker's input is classified correctly, the attack becomes ineffective.
- Differential immunity

  *All configurations should not be vulnerable to the same attack.*

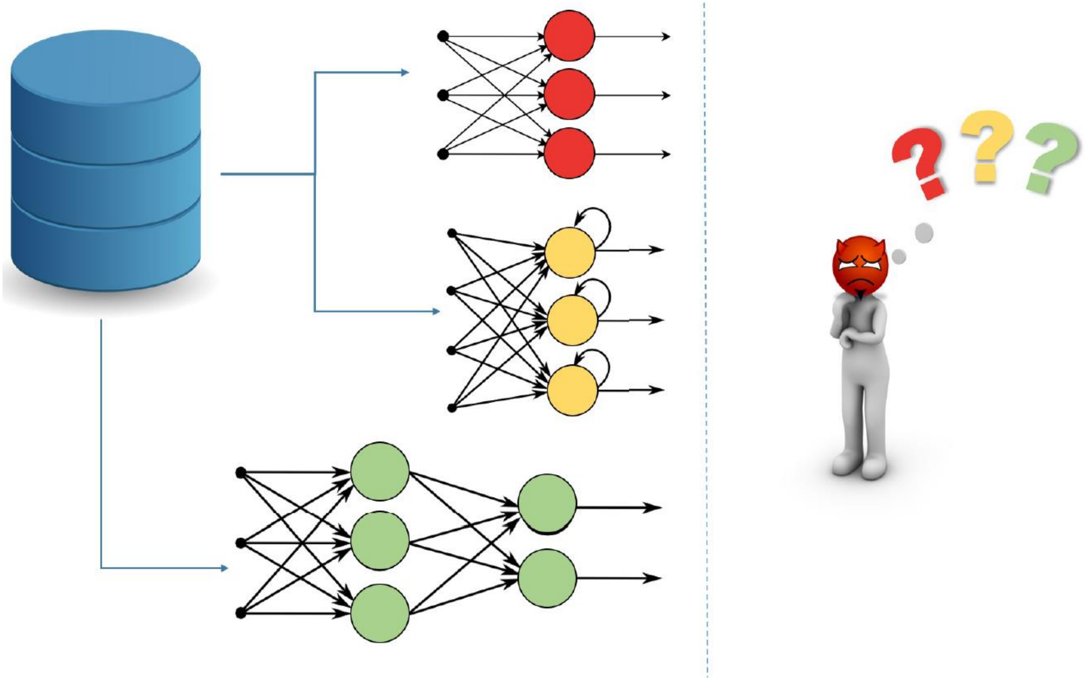


# Moving Target Defense for Deep Neural Networks (MTDeep)

- Differential immunity ($\delta$)

  All configurations should not be vulnerable to the same attack.

- We define a metric that can give the value of differential immunity for an ensemble of networks.



$$0 \le \delta \le 1$$

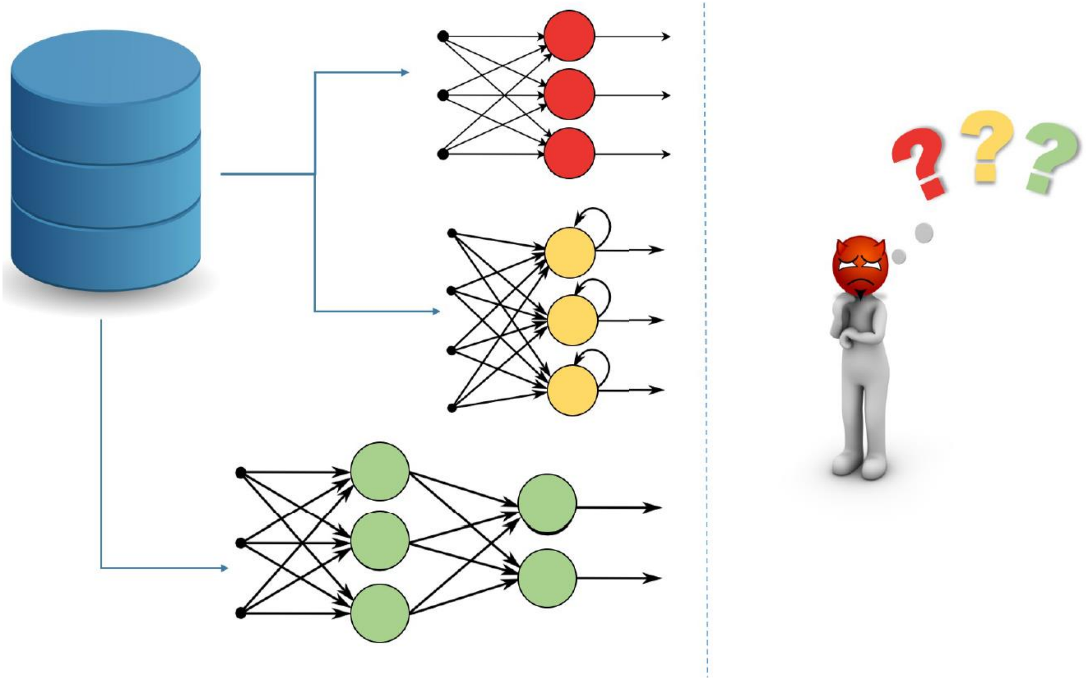
No differential immunity

No differential immunity

# Challenges for MTDeep

- Is absolute differential immunity, i.e. $\delta = 1$ possible for neural networks?
  - Difficult because the set of input features that are important for classification remains the same.
  - Attack algorithms use these features as the heuristics to construct perturbations.

- Can we *still* get security benefits by using MTD?
  - If you switch between configuration strategically, even with small $\delta$ we can have security gains.



# Goal of MTDeep

**Threat model**

- Attacker knows the different configurations in the system. Has the power to design strong whitebox attacks (which are inherently more effective than blackbox attacks) for each system.
- Over time the attacker is able to infer the switching policy of MTDeep.

**Expectations**

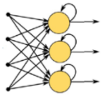
- Reduce the misclassification rate on adversarially perturbed inputs.
- Maintain the classification accuracy on legitimate input samples.

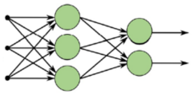
# Modelling MTDeep as a Game

Constant sum game between the classification system MTDeep and the attacker who is trying to fool MTDeep.

Let $\vec{x}$ denote the strategy vector over the defender's actions.

$\vec{x} = \langle x_1, x_2 \rangle = \langle 0.3, 0.7 \rangle$
where $x_i = \Pr(\text{using } N_i)$





| | FGSM attack for $N_1$ | FGSM attack for $N_2$ |
|---|---|---|
| $N_1$ | 20 80 | 40 60 |
| $N_2$ | 45 55 | 10 90 |

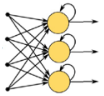
# Modelling MTDeep as a Game

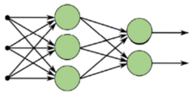
Constant sum game between the classification system MTDeep and the attacker who is trying to fool MTDeep.

Let $\vec{x}$ denote the strategy vector over the defender's actions.

$$\vec{x} = \langle x_1, x_2 \rangle$$
where $x_i = \Pr(\text{using } N_i)$

**Stackelberg equilibrium** of this constant-sum leader-follower repeated game gives the optimal switching strategy!





| | FGSM attack for $N_1$ | FGSM attack for $N_2$ |
|---|---|---|
| $N_1$ | 20 80 | 40 60 |
| $N_2$ | 45 55 | 10 90 |

# Modelling MTDeep as a Game

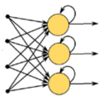
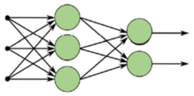
Constant sum game between the classification system MTDeep and the attacker who is trying to fool MTDeep.

Let $\vec{x}$ denote the strategy vector over the defender's actions.

$\vec{x} = \langle x_1, x_2 \rangle$
where $x_i = \Pr(\text{using } N_i)$

**Stackelberg equilibrium** of this constant-sum *leader-follower* repeated game gives the optimal switching strategy!

☺ Reduce misclassification rate on adversarial inputs.
☹ Maintain classification accuracy of the present system.

| | | FGSM attack for $N_1$ | FGSM attack for $N_2$ |
|---|---|---|---|
|  | $N_1$ | 20 80 | 40 60 |
|  | $N_2$ | 45 55 | 10 90 |

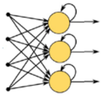
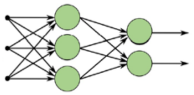
# Modelling MTDeep as a Game

Constant sum game between the classification system MTDeep and the attacker who is trying to fool MTDeep.

Let $\vec{x}$ denote the strategy vector over the defender's actions.

Add a new user type who has only one action – to provide legitimate test image.

$\vec{x} = \langle x_1, x_2 \rangle$
where $x_i = \Pr(\text{using } N_i)$

| | | FGSM attack for $N_1$ | FGSM attack for $N_2$ | Legitimate User Input |
|---|---|---|---|---|
|  | $N_1$ | 20 80 | 40 60 | 96 96 |
|  | $N_2$ | 45 55 | 10 90 | 97 97 |

# Modelling MTDeep as a Game

Constant sum game between the classification system MTDeep and the attacker who is trying to fool MTDeep.

Let $\vec{x}$ denote the strategy vector over the defender's actions.

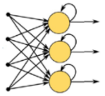
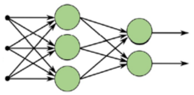
$$\vec{x} = \langle x_1, x_2 \rangle$$
where $x_i = \Pr(\text{using } N_i)$

Add a new user type who has only one action – to provide legitimate test image.

**Stackelberg equilibrium** now solves the multi-objective function. Increases defender's utility, which

☺ Reduce misclassification rate on adversarial inputs.

☺ Increases classification accuracy on legitimate samples.

| | | FGSM attack for $N_1$ | FGSM attack for $N_2$ | Legitimate User Input |
|---|---|---|---|---|
|  | $N_1$ | 20 80 | 40 60 | 96 96 |
|  | $N_2$ | 45 55 | 10 90 | 97 97 |

# Optimal Switching Strategy

- We use the DOBSS formulation P. Paruchuri et al., 2008 for calculating the **Stackelberg Equilibrium** of our game. The multi-objective function we optimize for the classification system MTDeep is,

Weightage of legitimate user

Weightage of attacker

Attacker’s strategy

$$\max_{x,q} \sum_{n \in N} \left( \alpha \cdot \sum_{u \in U} R^{D}_{n,u} \, x_n q^{\mathcal{A}}_{u} \quad + \quad (1 - \alpha) \cdot R^{D}_{n,u} \, x_n q^{\mathcal{L}}_{u} \right)$$

MTDeep’s strategy

User’s strategy (trivial)

# Evaluations

- We use the DOBSS formulation P. Paruchuri et al., 2008 for calculating the **Stackelberg Equilibrium** of our game. The multi-objective function we optimize for the classification system MTDeep is,

Weightage of attacker

Weightage of legitimate user

Attacker's strategy

$$\max_{x,q} \sum_{n \in N} \left( \alpha \cdot \sum_{u \in U} R^{D}_{n,u}\, x_n q^{\mathcal{A}}_u \quad + \quad (1 - \alpha) \cdot R^{D}_{n,u}\, x_n q^{\mathcal{L}}_u \right)$$

MTDeep's strategy

User's strategy (trivial)

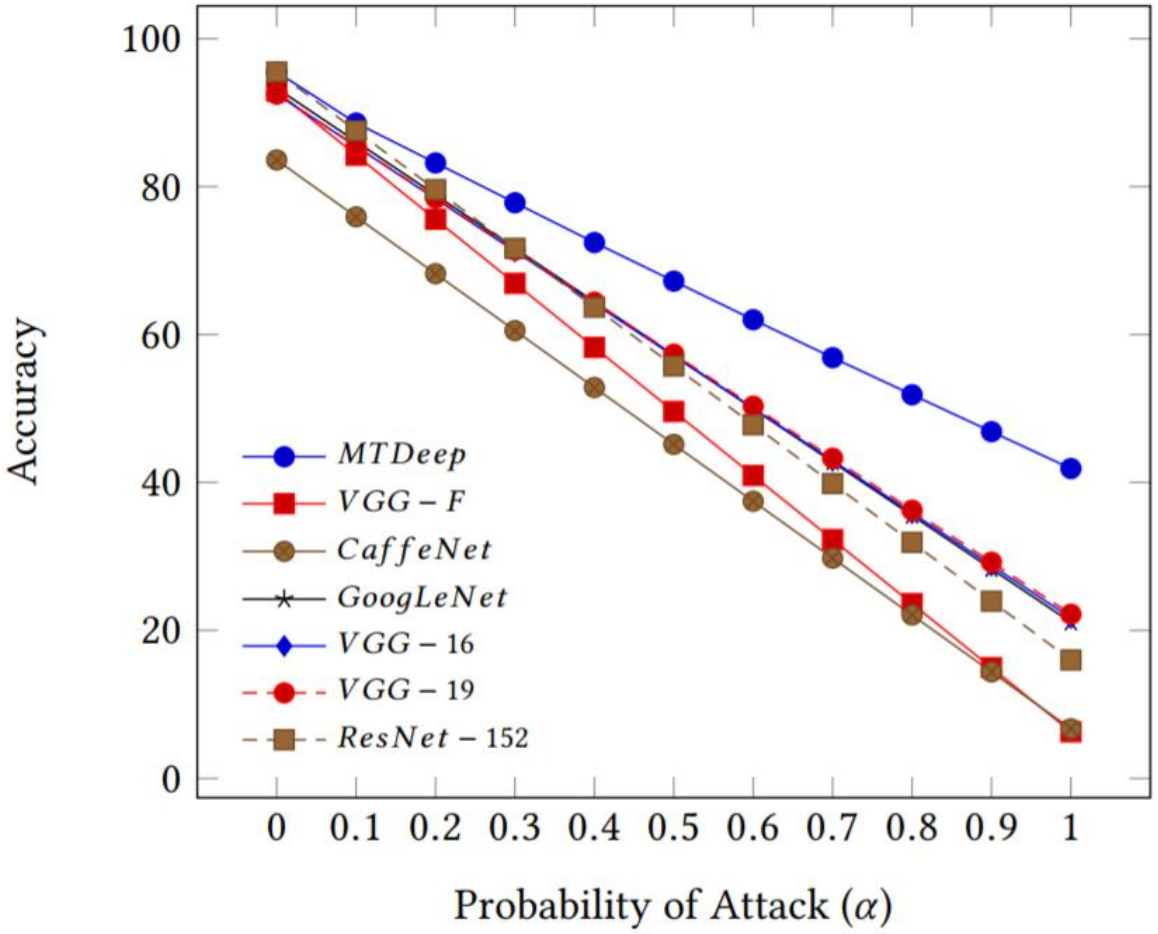
- We consider two different datasets – ImageNET and MNIST.
  - Evaluate against known whitebox attacks, universal perturbations.

# Universal Perturbation Attack against MTDeep for ImageNET

| MTDeep | Legitimate User ($\mathcal{L}$) Classification Image |
| --- | --- |
| VGG-F (Chatfield et al. 2014) | $(92.9, 92.9)$ |
| CaffeNet (Jia et al. 2014) | $(83.6, 83.6)$ |
| GoogLeNet (Szegedy et al. 2015) | $(93.3, 93.3)$ |
| VGG16 (Simonyan and Zisserman 2014) | $(92.5, 92.5)$ |
| VGG19 (Simonyan and Zisserman 2014) | $(92.5, 92.5)$ |
| ResNet-152 (He et al. 2016) | $(95.5, 95.5)$ |

| | Adversarial User ($\mathcal{A}$) | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| MTDeep | $UP_{VGG-F}$ | $UP_{CaffeNet}$ | $UP_{GoogLeNet}$ | $UP_{VGG-16}$ | $UP_{VGG-19}$ | $UP_{ResNet-152}$ |
| VGG-F | $(6.3, 93.7)$ | $(28.2, 71.8)$ | $(51.6, 48.4)$ | $(57.9, 42.1)$ | $(57.9, 42.1)$ | $(52.6, 47.4)$ |
| CaffeNet | $(26.0, 74.0)$ | $(6.7, 93.3)$ | $(52.3, 47.7)$ | $(60.1, 39.9)$ | $(60.1, 39.9)$ | $(52.0, 48.0)$ |
| GoogLeNet | $(53.8, 46.2)$ | $(56.2, 43.8)$ | $(21.1, 78.9)$ | $(60.8, 39.2)$ | $(60.2, 39.8)$ | $(54.5, 45.5)$ |
| VGG-16 | $(36.6, 63.4)$ | $(44.2, 55.8)$ | $(43.5, 56.5)$ | $(21.7, 78.3)$ | $(26.9, 73.1)$ | $(36.6, 63.4)$ |
| VGG-19 | $(36.0, 64.0)$ | $(42.8, 57.2)$ | $(46.4, 53.6)$ | $(26.5, 73.5)$ | $(22.2, 77.8)$ | $(42.0, 58.0)$ |
| ResNet-152 | $(53.7, 46.3)$ | $(53.7, 46.3)$ | $(49.5, 50.5)$ | $(53.0, 47.0)$ | $(54.5, 45.5)$ | $(16.0, 84.0)$ |

# Universal Perturbation Attack against MTDeep for ImageNET




When $\alpha = 1$,

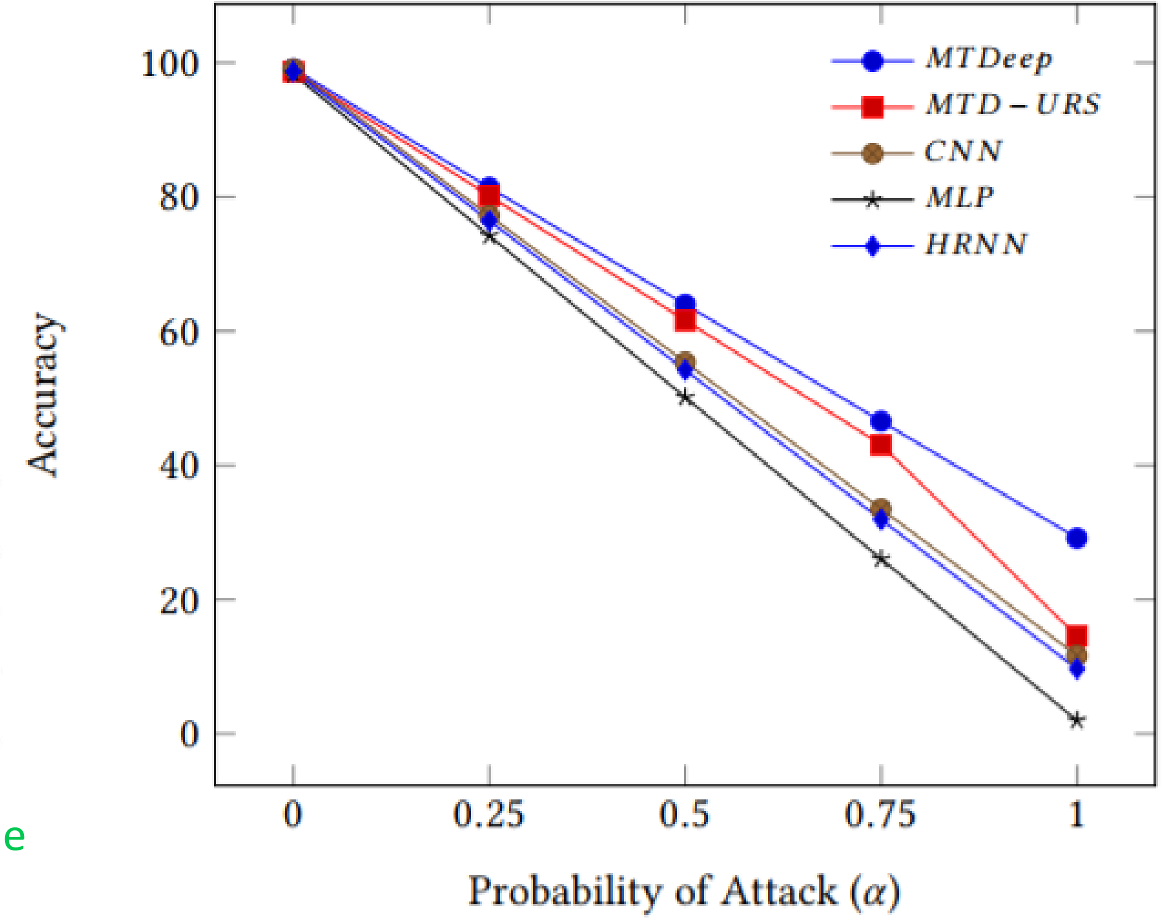
$$\vec{x} = (0, 0.171, 0.241, 0, 0.401, 0.187)$$

- In the generated strategy, not all configurations are used. Eg. VGG-F and VGG-16 are dropped from the ensemble.
- MTDeep as the first line of defense
  - No proven defense against Universal Perturbation.
  - Provides double the accuracy when classifying only adversarial inputs!

# FGSM against MTDeep for MNIST

| MTDeep | Legitimate User ($\mathcal{L}$) |
|---|---|
| | Classification Image |
| CNN | (99.1, 99.1) |
| MLP | (98.3, 98.3) |
| Hierarchical-RNN | (98.7, 98.7) |

| MTDeep | Adversarial User ($\mathcal{A}$) | | |
|---|---|---|---|
| | $FGSM_{CNN}$ | $FGSM_{MLP}$ | $FGSM_{HRNN}$ |
| CNN | (11.63, 88.37) | (47.54, 52.46) | (74.65, 25.35) |
| MLP | (36.37, 63.63) | (1.96, 98.04) | (38.10, 61.90) |
| HRNN | (35.72, 64.28) | (24.08, 75.92) | (9.65, 90.35) |




Note that the Stackelberg equilibrium guarantees double the accuracy even against MTD system using a Uniform Random Strategy (MTD-URS) in the worst case!

# Stronger Attacks against MTDeep + Adversarially Trained Nets on MNIST

| | Adversarial User ($\mathcal{A}$) | | | |
|---|---|---|---|---|
| MTDeep | $FGSM_C$ | $FGSM_M$ | $FGSM_H$ | $CWL2$ |
| CNN | 94.2, 5.8 | 97.8, 2.2 | 97.6, 2.4 | 80.0, 20.0 |
| MLP | 96.0, 4.0 | 87.0, 13.0 | 63.2, 36.8 | 90.0, 10.0 |
| HRNN | 95.9, 4.1 | 87.9, 12.1 | 93.2, 6.8 | 60.0, 40.0 |

Note that after adversarial training, the attacks are almost equally ineffective against the adversarially trained nets.

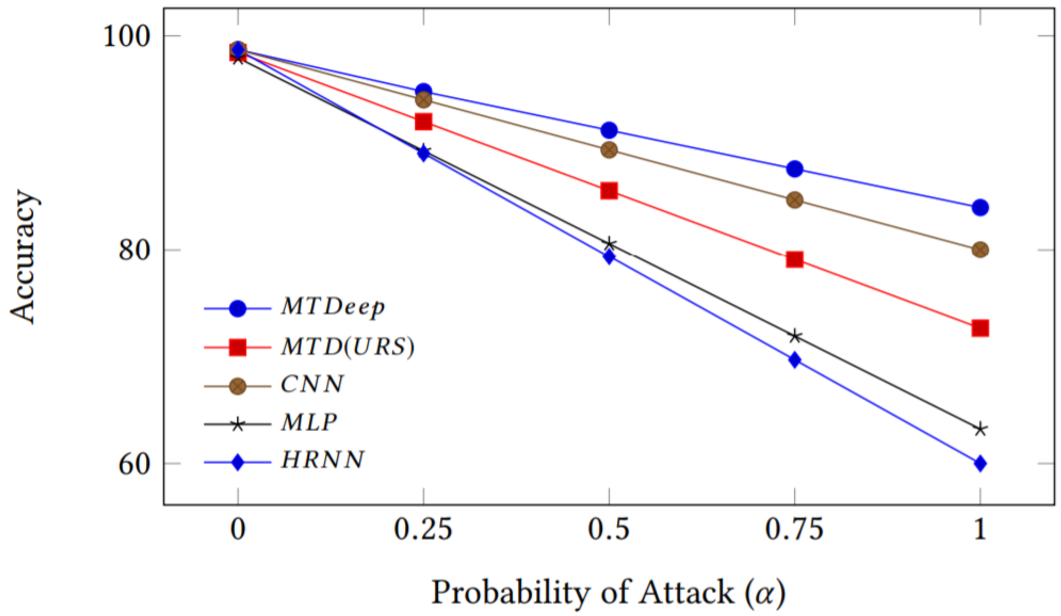
# Stronger Attacks against MTDeep + Adversarially Trained Nets on MNIST

Carlini-Wagner attacker based on the L2 norm.

| | Adversarial User ($\mathcal{A}$) | | | |
|---|---|---|---|---|
| MTDeep | $FGSM_C$ | $FGSM_M$ | $FGSM_H$ | $CWL2$ |
| CNN | 94.2, 5.8 | 97.8, 2.2 | 97.6, 2.4 | 80.0, 20.0 |
| MLP | 96.0, 4.0 | 87.0, 13.0 | 63.2, 36.8 | 90.0, 10.0 |
| HRNN | 95.9, 4.1 | 87.9, 12.1 | 93.2, 6.8 | 60.0, 40.0 |

New attacks, stronger than FGSM can render the defense mechanism (adversarial training on FGSM) useless.

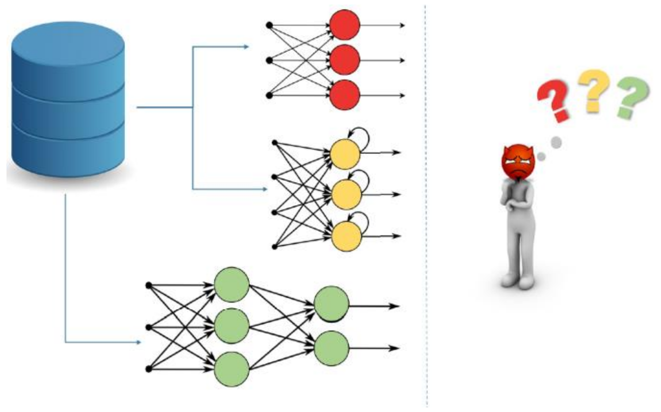
# Stronger Attacks against MTDeep + Adversarially Trained Nets on MNIST




- MTDeep on top of an existing defense mechanism gives an additional 4% increase in accuracy when $\alpha = 1$.
  - Defense mechanism in place already gives 82% accuracy on adv. inputs.
- Trivial strategies lead to detrimental effects.
  - CNN gives higher accuracy than the MTD(URS) system.

# Conclusions and Future Work

- We proposed an MTD framework for securing DNNs - **MTDeep**.
  - We use Moving Target Defense for an ensemble of Deep Neural Networks.
  - We formulated the interaction between the classification system and the users as a Repeated Bayesian Game.
- We empirically demonstrate the effectiveness of MTDeep on MNIST and ImageNET datasets against a variety of well-known attacks.




# *[New results]* What if you do blackbox attacks against MTDeep?

- Blackbox attacks, are weaker than whitebox attacks for constituent DNNs.

☺ In our case, the blackbox attack has to learn the actual the prediction of the ensemble and the randomization implicitly built into MTD.

☹ The game theory framework cannot model these attacks as they are not-existent during the first deployment of the system.

- Blackbox attacks are weaker (65% misclassification) vs whitebox attacks (70% misclassification) against MTDeep.