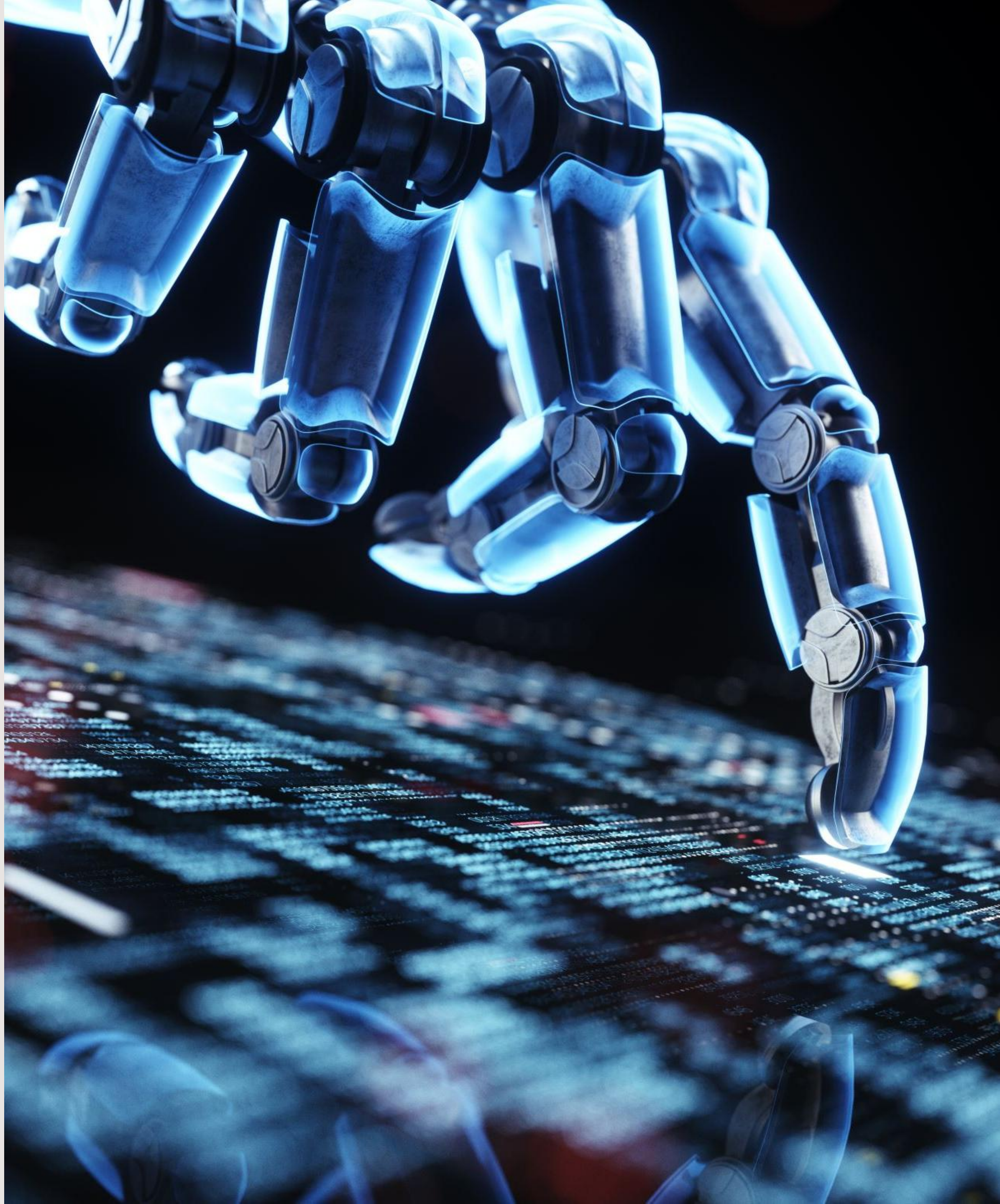
Practical Black-Box Attacks against Machine Learning



Submitted By –

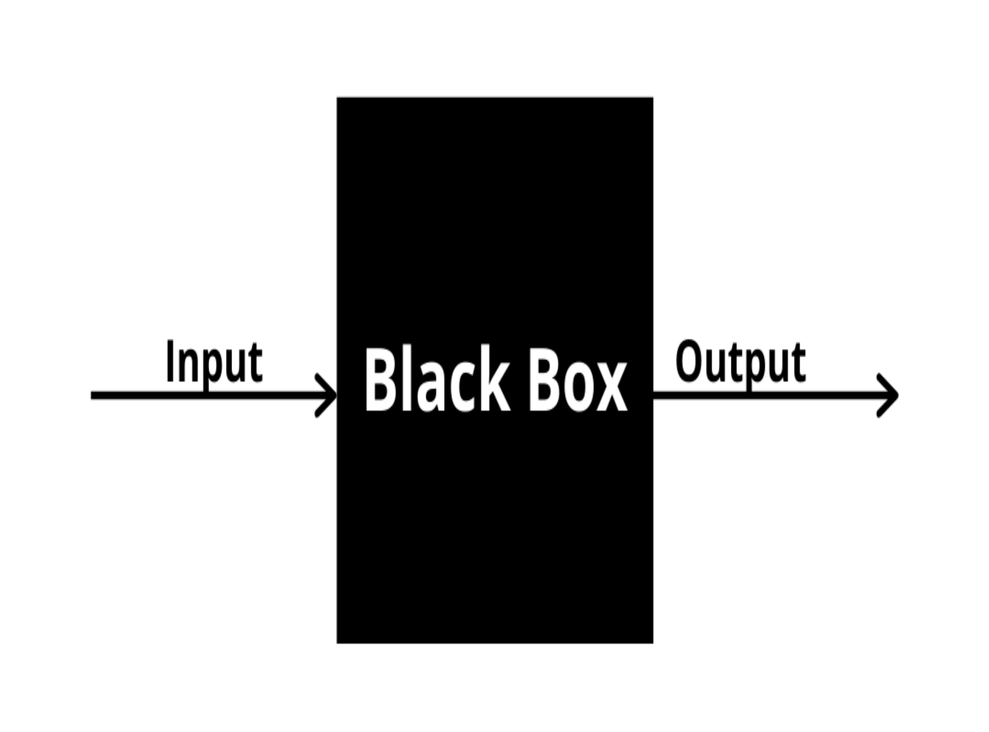
Asha Tummuru(U38611026)

Vani Pailla(U95351377)

Define Black-Box Attack:

- It's an adversarial attack where the attacker has no knowledge of the target model's internal architecture or parameters.

- In a black-box attack, the adversary's goal is to trick a deep neural network (DNN) classifier into misclassifying inputs without any knowledge of the model’s internal architecture, such as layer types, layer sizes, or training data. Unlike white-box attacks, where the attacker has full access to the model, black-box attacks rely solely on the model’s output responses to input queries.



CORE PROBLEM:

- **Black-box attacks target machine learning models by misleading them without needing access to their internal workings. Cause serious risks to the reliability of models in critical fields like healthcare, autonomous driving, and finance.**
- **The main objective of our presentation is to explain and demonstrate what a black box attack against deep neural networks (DNN) classifiers is, how to implement it and show some practical examples.**

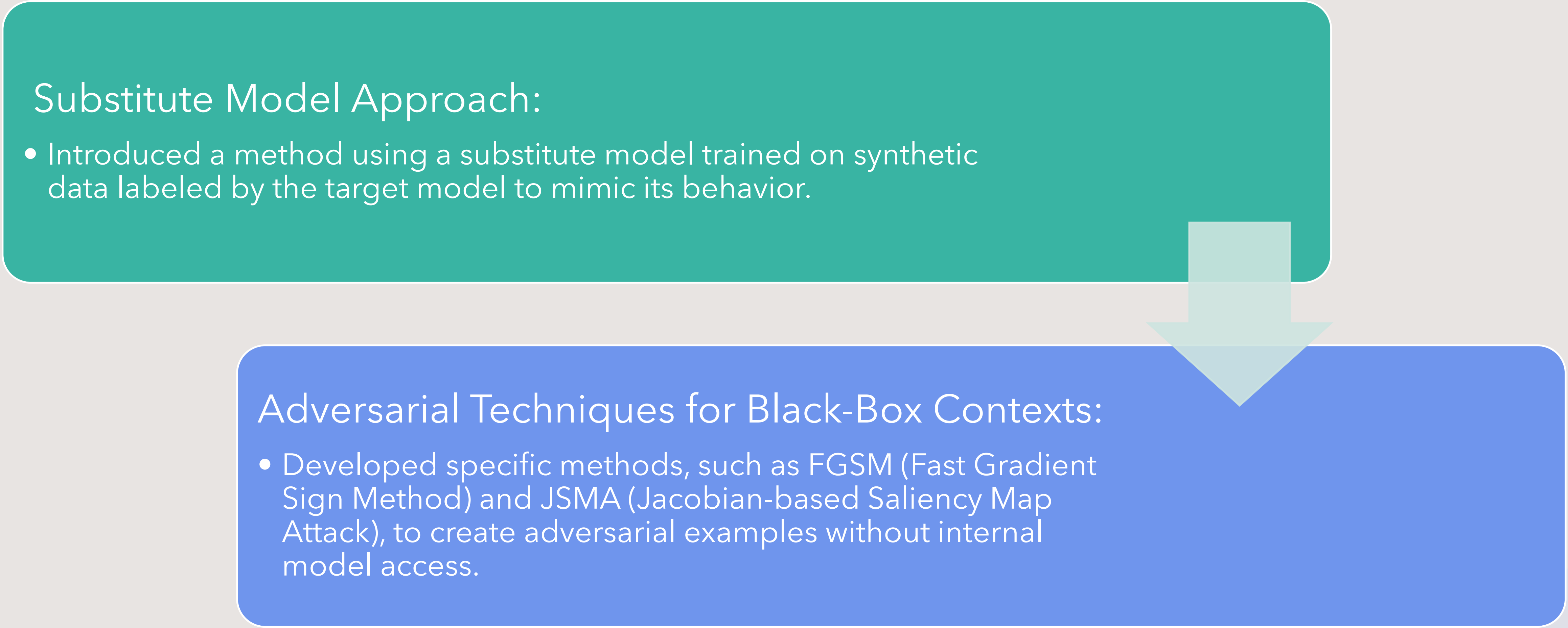
Challenges in Black-Box Attacks & Defenses:

Lack of Model Access: Attackers can only see outputs, not the internal mechanics, making it difficult to design precise attacks or defenses.

Limited Queries: Attackers are constrained by a limited number of queries to avoid detection, complicating their ability to gather enough data for an effective attack.

Balancing Defense and Performance: Many defense strategies reduce model accuracy, making it challenging to protect models without sacrificing performance.

Black Box Attack Strategy:



Substitute Model Approach:

- Introduced a method using a substitute model trained on synthetic data labeled by the target model to mimic its behavior.

Adversarial Techniques for Black-Box Contexts:

- Developed specific methods, such as FGSM (Fast Gradient Sign Method) and JSMA (Jacobian-based Saliency Map Attack), to create adversarial examples without internal model access.

Substitute Model Training:

- Attackers create synthetic data by querying the target model and observing its outputs. Using these labeled responses, they train a substitute model to approximate the target model's behavior.

Substitute Model Architecture:

- The adversary must have some partial knowledge of the oracle input (images, text, …) and expected output so he can use an architecture adapted to the input-output relation.

  For instance a convolutional neural network is suitable for image classification.

Synthetic Data Generation:

- Starts with a small set of initial data and iteratively generate more synthetic samples by exploring changes in the model's outputs.
- We could make an infinite number of queries to obtain the oracle's output $O(\bar{x})$ for any input $\bar{x}$ belonging to the input domain and this would provide us with a copy of the oracle. However this is simply intractable.
- To address this issue a heuristic efficiently exploring the input domain was introduced. The heuristic used to generate synthetic training inputs is based on identifying directions in which the model's output is varying, around an initial set of training data. These directions are identified with the substitute DNN's Jacobian matrix J_F, which is evaluated at several input points $\bar{x}$.

More precisely the adversary calculates $\operatorname{sgn}\left(J_F(\bar{x})\left[\hat{O}(\bar{x})\right]\right)$.

To obtain a new synthetic a term $\lambda \operatorname{sgn}\left(J_F(\bar{x})\left[\hat{O}(\bar{x})\right]\right)$ is added to the original point $\bar{x}$.

Substitute DNN Training Algorithm:

- Initial collection: The adversary collects a very small set of inputs representative of the input domain
- Architecture Selection: The adversary selects an architecture to be trained as the substitute F
- Substitute Training: The adversary iteratively trains more accurate substitute DNNs F_p by repeating the following $for\ p \in 0 \ldots p_{\max}$:

  Labeling 3: the adversary labels each sample $\bar{x} \in S_p$ in its initial substitute training set S_p;

  Training 4: The adversary trains the architecture chosen using the substitute training set S_p;

Augmentation: The adversary applies the Jacobian based dataset augmentation on the initial substitute training S_p to produce a larger substitute training set S_{p+1}. The new training set better represents the model's decision boundaries. The adversary repeats steps 3 and 4 with the augmented set S_{p+1}.

Substitute Model Training:

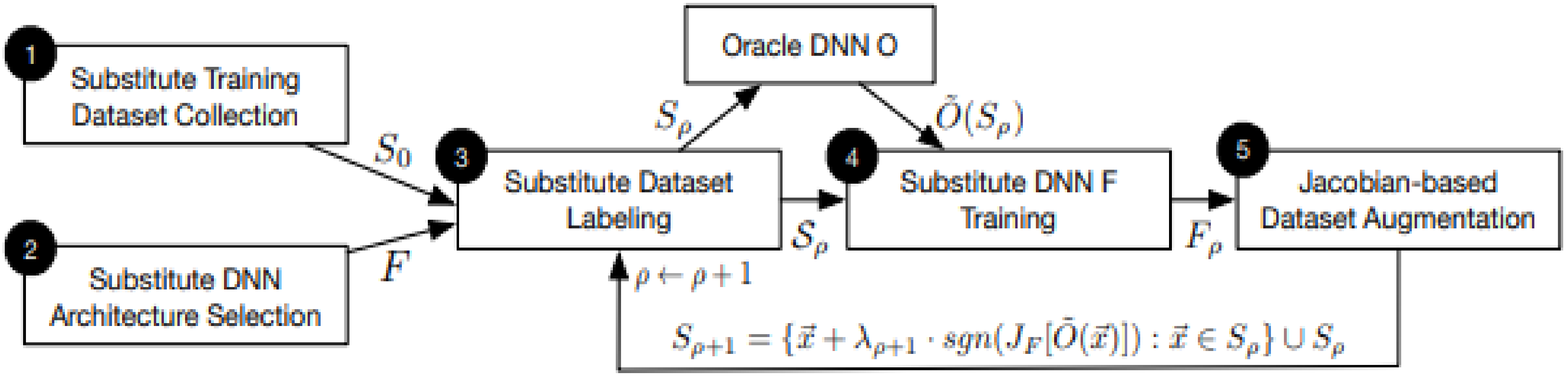

1
Substitute Training Dataset Collection
2
Substitute DNN Architecture Selection
S_0
F
3
Substitute Dataset Labeling
Oracle DNN O
S_ρ
$\tilde{O}(S_\rho)$
4
Substitute DNN F Training
S_ρ
5
Jacobian-based Dataset Augmentation
F_ρ
$\rho \leftarrow \rho + 1$
$S_{\rho+1} = \{\vec{x} + \lambda_{\rho+1} \cdot sgn(J_F[\tilde{O}(\vec{x})]) : \vec{x} \in S_\rho\} \cup S_\rho$

Figure 3: **Training of the substitute DNN F:** the attacker (1) collects an initial substitute training set S_0 and (2) selects an architecture F. Using oracle $\tilde{O}$, the attacker (3) labels S_0 and (4) trains substitute F. After (5) Jacobian-based dataset augmentation, steps (3) through (5) are repeated for several substitute epochs ρ.

Adversarial Sample Crafting:

FGSM (Fast Gradient Sign Method)	JSMA (Jacobian-based Saliency Map Attack)
Adds small, calculated perturbations to inputs to shift the output of the target model. i.e, x adv =x+ϵ·sign(∇ xJ(x,y)), Where xx is the original input, adv is the adversarial example, ϵ controls perturbation magnitude, ∇ x is the gradient, and J is the loss function.	Targets specific input features by modifying selected pixels or elements, minimizing changes while achieving misclassification. i.e, Given model F, the adversary crafts an adversarial sample $\bar{x}^* = \bar{x} + \delta_{\bar{x}}$ for a given legitimate sample $\bar{x}$ by adding a perturbation $\delta_{\bar{x}}$ to a subset of the input components $\bar{x}_i$.
Fast and Effective	Effective for source-target misclassification
Especially for large models	Slower but allows for fine-tuned, targeted attacks
Ex: In an image classifier, FGSM might alter a few pixels in a way that's invisible to humans but significant enough to make the model misclassify a "cat" as a "dog."	Ex: For handwritten digit classification, JSMA might change parts of an image of "3" to be recognized as "8," targeting specific pixels to achieve this.

Attack Validation:

To validate the attack, tried it against different classifiers and using also different types of attack and first made an FGSM attack to target DNN trained using MNIST dataset, then we made another attack against a DNN trained with CIFAR dataset, both attacks have the goal to misclassify most of adversarial examples crafted with a perturbation not affecting human recognition. Finally, both the attack using a JSMA type of attack.

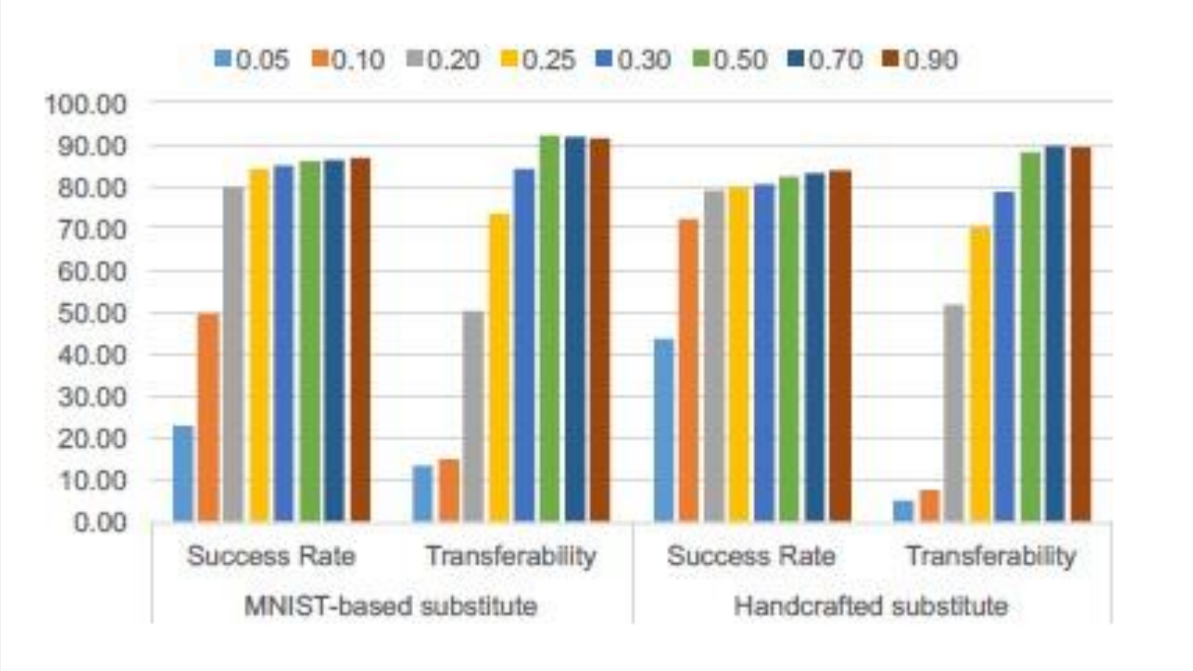
The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems, it is widely used for training and testing in the field of machine learning.

Attack Validation Continued…

Handcrafted set: To ensure the results do not stem from similarities between the MNIST test and training sets, they also consider a handcrafted initial substitute training set. We handcrafted 100 samples by handwriting 10 digits for each class between 0 and 9 with a laptop trackpad. In the table, each column corresponds to an initial substitute training set: 150 MNIST test samples, and handcrafted digits. Accuracy is reported on the unused 9,850 MNIST test samples.

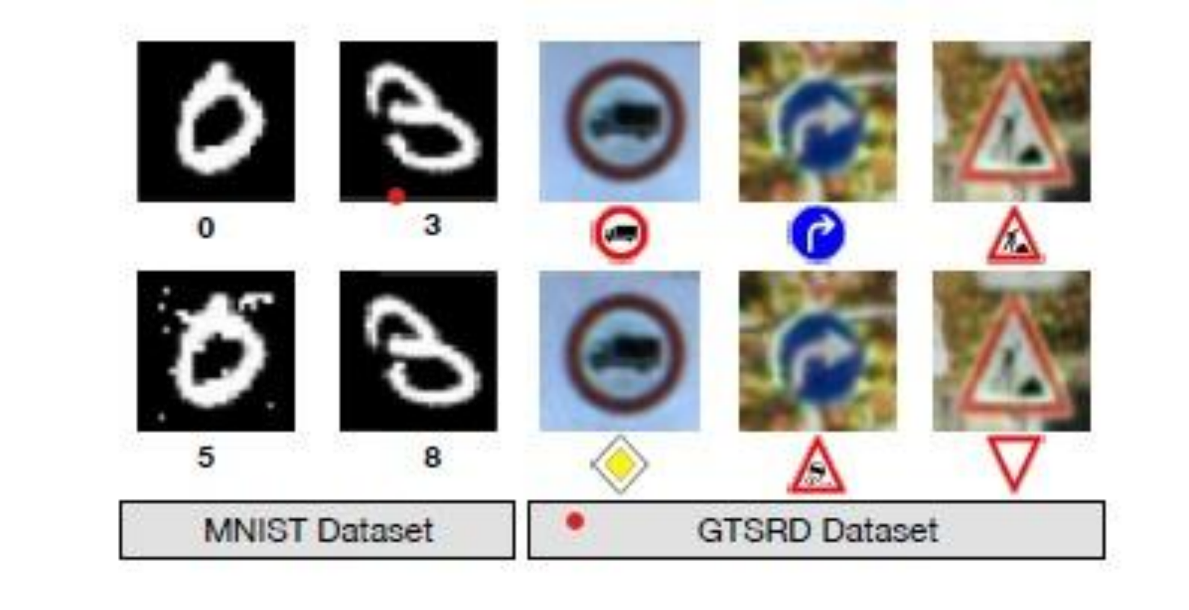
Substitute	**Initial Substitute Training Set from**	
Epoch	MNIST test set	Handcrafted digits
0	24.86%	18.70%
1	41.37%	19.89%
2	65.38%	29.79%
3	74.86%	36.87%
4	80.36%	40.64%
5	79.18%	56.95%
6	81.20%	67.00%

Success Rate and Transferability of Adversarial Samples for the MetaMind attacks: performed using MNIST-based and handcrafted substitutes: each bar corresponds to a different perturbation input variation.



Generalization Of Attack:

- The goal is to verify whether these samples are also misclassified by the oracle or not. Therefore, the transferability of adversarial samples refers to the oracle misclassification rate of adversarial samples crafted using the substitute DNN.
- Substitutes and oracles taken in cause were learned with DNNs, but the attack bounds its applicability to other ML systems.



Evaluation:

- **Tests were conducted on models from MetaMind, Amazon, and Google to validate the black-box attack. These experiments showed that adversarial examples crafted on substitute models could reliably mislead the target models.**
- **Misclassification success rates were significant.**

		Amazon		Google	
Epochs	Queries	DNN	LR	DNN	LR
$\rho = 3$	800	87.44	96.19	84.50	88.94
$\rho = 6$	6,400	96.78	96.43	97.17	92.05
$\rho = 6^*$	2,000	95.68	95.83	91.57	97.72

Misclassification rates (%) of the Amazon and Google oracles on adversarial samples produced with DNN

Defense Mechanisms

Adversarial Training(Reactive Defense):

- The model is retrained with adversarial examples to make it more resilient to attacks.
- Seeks to improve the generalization of a model when presented with adversarial examples at test time by proactively generating adversarial examples as part of the training procedure

Defense Mechanisms

Defensive Distillation:

- Reduces the sensitivity of the model’s decision boundaries, making it harder for small input changes to lead to misclassification.
- Training procedure where one model is trained to predict the probabilities output by another model that was trained earlier.
- It may seem counterintuitive to train one model to predict the output of another model that has the same architecture

Conclusion:

- The author introduced an attack, based on a novel substitute training algorithm using synthetic data generation, to craft adversarial examples misclassified by black-box DNNs.
- The work is a significant step towards relaxing strong assumptions about adversarial capabilities made by previous attacks.
- Assumed only that the adversary is capable of observing labels assigned by the model to inputs of its choice and Validated the attack design by targeting a remote DNN served by MetaMind, forcing it to misclassify 84.24% of our adversarial samples.

References:

- Marco Barreno, et al. Can machine learning be secure? In

  Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security.
- Ian Goodfellow, et al. Deep learning. Book in preparation for MIT Press (www.deeplearningbook.org), 2016.
- Ling Huang, et al. Adversarial machine learning. In

  Proceedings of the 4th ACM workshop on Security and artificial intelligence, pages 43–58, 2011.
- Erich L. Lehmann, et al. *Testing Statistical Hypotheses*. Springer Texts in Statistics, August 2008.
- D Warde-Farley, et al. Adversarial perturbations of deep neural networks. *Advanced Structured Prediction*, 2016.

THANK YOU