Adversarial Attacks and Defenses for Deep-Learning-Based Unmanned Aerial Vehicles [12]

Insights from the Paper by Tian et al. (2022)

Guilherme Marcello

Graduate School of Artificial Intelligence
Pohang University of Science and Technology

Trustworthy Machine Learning
May 2025

Outline

Glossary

- **CPS:** Cyber–Physical System.
- **UAV:** Unmanned Aerial Vehicle (Drone).
- **DL:** Deep Learning.
- **NN / DNN:** Neural Network / Deep Neural Network.
- **CNN:** Convolutional Neural Network.
- **Adversarial Example:** An input sample intentionally perturbed to cause a machine learning model to make a specific incorrect prediction.
- **White-Box Attack:** Attacker knows the target model completely.
- **Black-Box Attack:** Attacker has no knowledge of the target model but can query it.

Introduction: UAVs and Deep Learning

- UAVs are increasingly used in civilian and military areas [5].
- Applications include plant protection, parcel delivery, surveillance, and even acting as IoT platforms [5, 8].

Figure: Various roles of UAVs [2]

Introduction: UAVs and Deep Learning

- DL enables learning from experience and generalizing from sensor data (cameras, LiDAR).
- **DroNet** is a specific CNN model used for UAV navigation, predicting steering angles and collision probabilities from camera images [7].

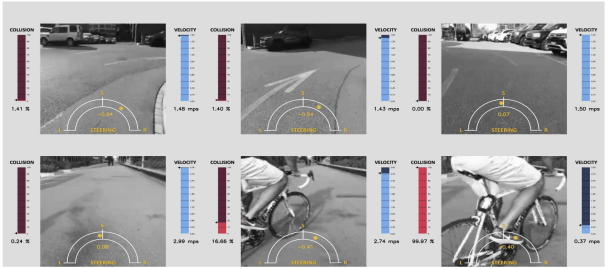

Figure: DroNet's predicted steering and probability of collision evaluated over several experiments [7] (Fig. 8 in the paper)

The Problem: Security Vulnerabilities

- The use of DL in safety-critical CPSs like autonomous cars and UAVs introduces new security issues. [6, 11, 13, 4, 3]
- Prior research focused on adversarial attacks on DL models, but often not on safety-critical CPSs or regression models.

The Problem: Security Vulnerabilities

- DL algorithms in UAVs are not typically designed or evaluated with adversarial attacks in mind.
- This paper addresses this gap by exploring adversarial attacks and defenses specifically for DL-based UAVs.

Threat Model

- Attacker injects imperceptible perturbations δ into camera images over Wi-Fi [12]
- **White-box:** known model F, weights, gradients
- Objectives:
  - Nontargeted: maximize $\|F(x+\delta) - F(x)\|_2$ under $\|\delta\| \leq \epsilon$
  - Targeted: enforce $\|F(x+\delta) - F(x)\|_2 \geq \tau$

Threat Model

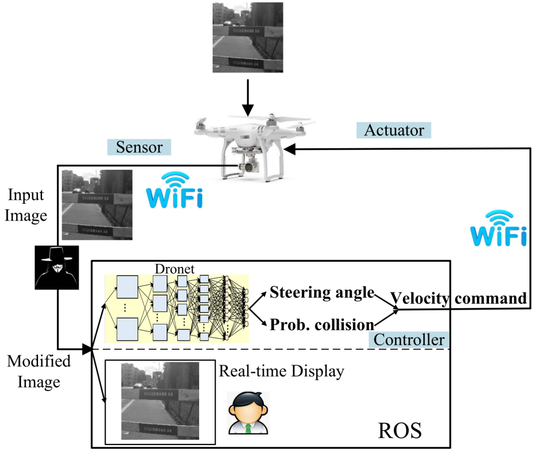

Figure: Attack schematic diagram against DL-based UAVs [12] (Fig. 3 in the paper)

Attack Method 1: FNA (Nontargeted)

- **FNA:** Forward Derivative-Based Nontargeted Attack [10].
- **Goal:** Efficiently craft images to cause the **largest possible change** in regression output with minimal perturbation. Degrading navigation/control.

Attach Method 1: FNA (Nontargeted) I

1. **Initialize perturbation:**

$$\delta \leftarrow 0$$

   Start with no perturbation; all pixel changes are initialized to zero.

2. **Compute Jacobian of model outputs:**

$$J_F(x) = \frac{\partial F(x)}{\partial x}$$

   Let $F(x) = [F_1(x), F_2(x)]$, where F_1 is the steering angle and F_2 is the collision probability. The Jacobian indicates how each pixel affects each output.

Attach Method 1: FNA (Nontargeted) II

3. **Compute absolute-impact metric:**

$$J_{F_{am}}(x) = |J_{F1}(x)| + |J_{F2}(x)|$$

   This measures how influential each pixel is on the outputs, regardless of direction.

4. **Select top influential pixels:**

$$\Upsilon = \text{top-}\gamma \text{ pixels in } J_{F_{am}}(x)$$

   Retain only the top γ fraction of pixels with highest impact. Lower γ yields stealthier attacks; higher γ yields stronger ones.

Attach Method 1: FNA (Nontargeted) III

5. **Zero out non-selected pixels:**

$$\delta_x^{\Omega \setminus \Upsilon} = 0$$

   Perturb only the selected pixels Υ; others remain unchanged.

6. **Compute signed-impact metric:**

$$J_{F_{rm}}(x) = J_{F1}(x) + J_{F2}(x)$$

   The sum of raw derivatives, used to determine direction of perturbation.

Attach Method 1: FNA (Nontargeted) IV

7. **Assign perturbation direction:** For each $x_i \in \Upsilon$,

$$\delta_{x_i} = \begin{cases} +\theta & \text{if } J_{F_{rm}}(x_i) \geq 0 \\ -\theta & \text{otherwise} \end{cases}$$

   Push pixel intensity up or down depending on which direction increases total output change.

8. **Construct adversarial image:**

$$x' = \text{clip}(x + \delta, 0, 1)$$

   Add perturbation and clip to valid image range.

FNA: Algorithm

Algorithm 1: FNA Crafting Method

```
    procedure FNA CRAFTING($x$, $F$, $\gamma$, $\theta$)
2:      Initialize $\delta \leftarrow 0$
        Calculate Jacobian: $J_F(x) = \frac{\partial F(x)}{\partial x}$
4:      Calculate metric: $J_{F_{am}}(x) = |J_{F1}(x)| + |J_{F2}(x)|$
        Find set $\Upsilon$: $\gamma$ ratio largest elements of $J_{F_{am}}(x)$
6:      Set $\delta_x^{\Omega \setminus \Upsilon} = 0$ (perturb only selected pixels)
        Calculate metric: $J_{F_{rm}}(x) = J_{F1}(x) + J_{F2}(x)$
8:      for $x_i$ in $\Upsilon$ do
            if $J_{F_{rm}}(x_i) \geq 0$ then
10:             $\delta_{x_i} = \theta$
            else
12:             $\delta_{x_i} = -\theta$
            end if
14:     end for
        Adversarial image: $x' = \text{clip}(x + \delta, 0, 1)$
16:     return $x'$
    end procedure
```

FNA: Why it makes sense? I

- **Why Jacobian?**
  The Jacobian directly measures how the model output changes in response to each input pixel. This makes it ideal for crafting effective perturbations in regression settings.
- **Why absolute-impact metric?**
  Using $|J_{F1}(x)| + |J_{F2}(x)|$ helps identify which pixels have the *greatest influence*, regardless of the direction of change.
- **Why signed-impact metric?**
  The sum $J_{F1}(x) + J_{F2}(x)$ helps determine the *direction* in which to perturb each pixel to maximize output deviation.
- **Why perturb only top-γ pixels?**
  Modifying only the most influential pixels allows the attack to remain *stealthy* while still being effective.

FNA: Why it makes sense? II

- **Why no iterations?**
  The attack is efficient and fast - no need for iterative optimization. This makes it well-suited for real-time or constrained environments like UAVs.

FNA: Wrap-up I

- **Computational Complexity:**
  - One forward pass to compute outputs.
  - One backward pass to compute the Jacobian $\frac{\partial F(x)}{\partial x}$.
  - No iterative optimization required.
- **Advantages:**
  - *Fast and efficient* - suitable for real-time applications.
  - *Simple implementation* - relies on basic gradient operations.
  - *Stealthy* - controls sparsity via γ and magnitude via θ.
  - *Effective against regression models* like Dronet by maximizing deviation in continuous outputs.

Attack Method 2: OTA (Targeted)

- **OTA:** Optimization-Based Targeted Attack.
- **Goal:** Achieve a **specific target change** in regression output. E.g., steer UAV in a specific wrong direction.
- Requires the output to change beyond a certain threshold τ (attack target).

Attack Method 2: OTA (Targeted)

- Formulated as a constrained optimization problem to **minimize perturbation** δ while meeting the target output change constraint.

$$\min_{x'} \|x' - x\|_2^2 \quad \text{s.t.} \quad \|F(x') - F(x)\|_2 \geq \tau \tag{1}$$

- x is original image, x' is adversarial image.

OTA: Ensuring Valid Images and Reformulation

- **Challenge:** Directly modifying x' might result in pixel values outside the valid range (e.g., not in $[0, 1]$ for normalized images).
- **Solution (Carlini and Wagner, 2017):** Introduce a new variable w and define the perturbation δ (which is $x' - x$) in terms of w:

$$x' = \frac{1}{2}(\tanh(w) + 1) \tag{2}$$

  This transformation makes that x' (each pixel of the adversarial image) will always be within the valid range $[0, 1]$ because $\tanh(w)$ outputs values in $[-1, 1]$.
- The optimization is now performed over w instead of x'.

OTA: The Unconstrained Problem

- The constrained problem is hard to solve directly. It's converted into an unconstrained one using a **penalty function** ψ and a constant $c > 0$:

$$\min_{w} \underbrace{\left\| \frac{1}{2}(\tanh(w) + 1) - x \right\|_2^2}_{\text{Minimize Perturbation}} + c \cdot \underbrace{\psi(x, w, \tau)}_{\text{Penalty Term}} \tag{3}$$

- The penalty function ψ is defined as:

$$\psi(x, w, \tau) = \max\{0, \tau - \|F(x') - F(x)\|_2\} \tag{4}$$

- The overall process involves solving the minimization problem (using gradient descent on w) for several values of c chosen by the binary search.

OTA: Algorithm

Algorithm 2: OTA Crafting Method

```
procedure OTA CRAFTING($x$, $F$, $\tau$, $M$, $N$, $c_0$, $\alpha$)
    Initialize $c \leftarrow c_0, c^- \leftarrow 0, c^+ \leftarrow 1e5, w \leftarrow$ random, $D_{\min} \leftarrow \infty$
    for $n = 1 \ldots N$ do                                          ▷ Binary search steps for c
        for $m = 1 \ldots M$ do                                      ▷ Iterations for w with fixed c
            $x'_m \leftarrow \frac{1}{2}(\tanh(w_m) + 1)$
            $\psi \leftarrow \max\{\tau - \|F(x'_m) - F(x)\|_2, 0\}$
            $w_{m+1} \leftarrow w_m - \alpha \cdot \frac{\partial \psi}{\partial w_m}$                        ▷ Gradient descent
            if $\|F(x') - F(x)\|_2 \geq \tau$ and $\|x'_m - x\|_2 \leq D_{\min}$ then
                $x' \leftarrow x'_m, D_{\min} \leftarrow \|x'_m - x\|_2$
            end if
        end for                                                      ▷ Update c using binary search
        if $\|F(x') - F(x)\|_2 \geq \tau$ then
            $c^+ \leftarrow c$
        else
            $c^- \leftarrow c$
        end if
        $c \leftarrow (c^- + c^+)/2$
    end for
    return $x'$
end procedure
```

OTA: Why it makes sense? I

- **Why an optimization problem?**
  Minimizing perturbation $\|x' - x\|_2^2$ subject to achieving a target $\|F(x') - F(x)\|_2 \geq \tau$ directly addresses the goals of being effective and stealthy.
- **Why the** $\tanh(w)$ **transformation?**
  It ensures that the generated image $x' = \frac{1}{2}(\tanh(w) + 1)$ has pixel values within the valid range, preventing out-of-bounds values during optimization.
- **Why the penalty function** ψ **and constant** c**?**
  This is a standard technique (Lagrangian relaxation / penalty method) to convert a constrained optimization problem into an unconstrained one. The constant c acts as a knob to balance the importance of minimizing perturbation versus satisfying the attack target constraint.
  - $\psi = \max\{0, \tau - \|F(x') - F(x)\|_2\}$ only penalizes if the target isn't met.

OTA: Why it makes sense? II

- **Why binary search for c?**
  The binary search for c helps find the minimum c that allows the attack to succeed, thereby encouraging minimal changes to the input x.
- **Why gradient descent on w?**
  Once c is fixed (in an iteration of the binary search), gradient descent is an effective method to find the optimal w (and thus x') that minimizes the combined loss of perturbation and penalty for not meeting the target.

OTA: Wrap-up

- **Computational Complexity:**
  - Outer loop: N binary search steps for c.
  - Inner loop: M gradient descent steps for w for each c.
  - Each gradient descent step requires forward and backward passes through the model.
- **Advantages:**
  - *Targeted Attack*: Can achieve a specific, predefined change $(\geq \tau)$ in the regression output.
- **Trade-offs:**
  - Higher computational cost compared to non-iterative methods.
  - Requires careful tuning of parameters like M, N, learning rate α, and initial range for c.

Experimental Setup

- **Evaluation Metrics for Attacks:**[1]
  - Average Change Ratio of Input: $\|X' - X\|_2 / \|X\|_2$
  - Average Change of Output: $\|F(X') - F(X)\|_2$
  - Attack Success Rate (ASR) for OTA (is $\|F(X') - F(X)\|_2 \geq \tau$)
- **Baseline:** Original DroNet model performance on clean test data.

[1]**Environment:** Workstation with 4 Titan X GPUs, TensorFlow. **Dataset:** Publicly available DroNet datasets and pre-trained models. **OTA Parameters:** $N = 10$ binary search steps for c, $M = 1000$ gradient descent iterations for w, $\alpha = 0.0001$.

Results: FNA (Nontargeted Attack)

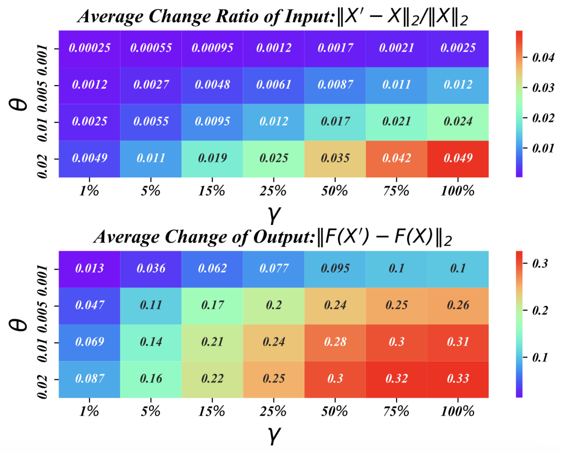

Figure: Fig. 5 from paper - FNA performance with varying θ and γ.

Results: FNA (Nontargeted Attack)

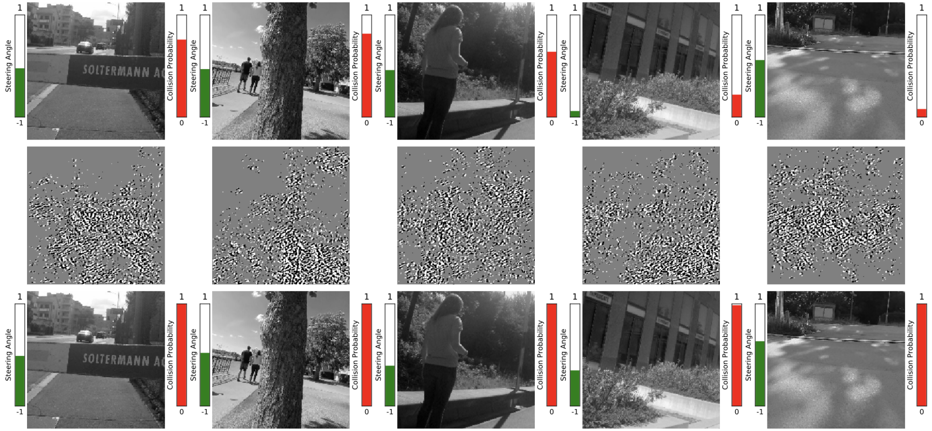

Figure: Fig. 6 from paper - Adversarial images by the FNA ($\theta = 0.005$ and $\gamma = 25\%$).

Results: FNA (Nontargeted Attack)

- **Impact of θ (perturbation magnitude) and γ (perturbation ratio):**
  - Increasing θ or γ generally increases the input change and output deviation.
  - FNA effectively finds influential pixels: output change increases sharply even with small γ.
- **Stealthiness:** Small, hardly perceptible perturbations can cause significant changes in regression outputs (e.g., collision probability from 0 to 1).

Results: OTA (Targeted Attack)

Table: Regression Performances by the OTA - Table II in the paper

τ	**Average Change Ratio of Input:** $\frac{\|X'-X\|_2}{\|X\|_2}$	**Average Change of Output:** $\|F(X')-F(X)\|_2$	**ASR**
0.1	0.05589	0.1191	100%
0.25	0.10875	0.27546	99.93%
0.5	0.16349	0.51895	98.45%
0.75	0.18582	0.72434	93.66%
1.0	0.24895	0.90183	89.15%

Results: FNA (Nontargeted Attack)

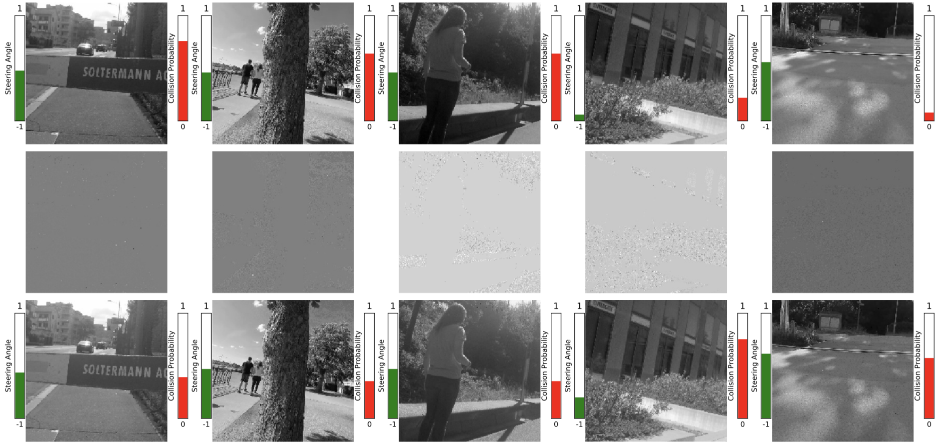

Figure: Fig. 8 from paper - Adversarial images by the OTA ($\tau = 0.25$).

Results: OTA (Targeted Attack) I

- **Stealthiness and Effectiveness:** Imperceptible perturbations can achieve significant, targeted regression errors (e.g., flipping collision probability).
- **Minimal Perturbation:** The method aims to minimize perturbation while meeting the target τ. Most successful attacks have output changes close to τ.

Results: Comparison of FNA and OTA

Key Findings (Table III in paper):

- **Efficiency (Time):**
  - FNA is significantly faster: $\sim$ 0.105 seconds per sample.
  - OTA is more time-consuming due to iterative optimization: $\sim$ 7.42 seconds per sample.
- **Attack Goals:**
  - FNA: Efficiently degrades performance (nontargeted).
  - OTA: Achieves precise, targeted output changes.
- **Threat Level:** Both pose a considerable threat to DL-based UAVs.

Adversarial Defenses: Introduction

- To counter the proposed adversarial attacks (FNA and OTA), the paper investigates two proactive defense methods.
- Goal: Enhance the robustness of the DL model (DroNet) used in UAVs.
- Methods Evaluated:
  1. Adversarial Training
  2. Defensive Distillation

Defense Method 1: Adversarial Training

- **Concept:** Retrain the model using a dataset augmented with adversarial examples.
- **Rationale:** Exposing the model to attacks during training helps it learn to be more resilient to similar attacks during deployment.

Defense Method 1: Adversarial Training

- **Process:**
  1. Train the model F_Θ normally on clean data D_{tr}.
  2. Generate a set of adversarial samples $\tilde{\chi}$ using a strong attack (e.g., OTA - Algorithm 2).
  3. Retrain the model F_Θ using this augmented dataset $\tilde{\chi}$.
- **Expectation:** Improved robustness and potentially better generalization.

Algorithm 3: Adversarial Training Defense

```
   procedure ADVERSARIAL TRAINING DEFENSE($D_{tr}$, $N_{iter1}$, $N_{iter2}$)
2:     Train model $F_\Theta$ using $D_{tr}$ for $N_{iter1}$
       Generate adversarial samples $\tilde{X}$ using Algorithm 2 on $F_\Theta$
4:     Retrain $F_\Theta$ using $\tilde{X}$ for $N_{iter2}$
       return Robust model $F_\Theta$
6: end procedure
```

Defense Method 2: Defensive Distillation

- **Concept:** Train a new model (F') to mimic the behavior of an initial, larger model (F), but with smoother outputs.
- **Rationale (Papernot et al., 2016):** Makes the model's decision boundaries smoother, reducing sensitivity to small input perturbations [9].

Defense Method 2: Defensive Distillation

- **Process for Dronet:**
  - The original Dronet model F is trained.
  - A new "distilled" model F' (same architecture) is trained.
  - The training of F' uses a modified loss function that incorporates information from F's outputs and internal layer activations (z^d):

$$\mathcal{L} = \sum_{i=1}^{n}((\lambda \cdot \|z_i^d - (z')_i^d\|) + \|F(x_i) - F'(x_i)\|)/n \tag{5}$$

  - λ is a temperature parameter balancing the terms.
- **Expectation:** Reduced gradient magnitudes, making it harder for gradient-based attacks.

Defensive Distillation: Why it makes sense?

- **Why the specific loss function** $\mathcal{L} = \sum_{i=1}^{n}((\lambda \cdot \|z_i^d - (z')_i^d)\| + ||F(x_i) - F'(x_i)||)/n$ **?**
  - $\lambda \cdot \|z_i^d - (z')_i^d\|$ **Term:** z^d are activations from an internal (last dense) layer of the original model F. This can lead to more robust feature learning.
  - **Role of λ (Temperature/Weighting):** It balances how much the distilled model focuses on mimicking the final output vs. the internal representations.
- **Why train a new model instead of just using the original?** The training process for the distilled model, using the combined loss, guides it towards this smoother, potentially more robust state. The original model might have learned sharp decision boundaries or highly sensitive features that make it vulnerable.

Defensive Distillation: Why it makes sense?

- **Why train a new model instead of just using the original?**
  The training process for the distilled model, using the combined loss, guides it towards this smoother, potentially more robust state. The original model might have learned sharp decision boundaries or highly sensitive features that make it vulnerable.

Results: Evaluation of Adversarial Defenses (vs. OTA)

Table: ASR with Adversarial Training/Defensive Distillation - Table IV in the paper.

No defense	τ	0.1	0.25	0.5	0.75	1.0
	ASR	100%	99.93%	98.45%	93.66%	89.15%
	τ	0.1	0.25	0.5	0.75	1.0
Adversarial training[2]	ASR	98.21%	97.61%	92.15%	87.63%	81.32%
	λ	0	0.05	0.1	0.5	1.0
Defensive Distillation[3]	ASR	99.93%	96.71%	61.24%	73.50%	82.88%

[2]The paper is not clear about the training conditions.

[3]$\tau = 0.25$

Results: Evaluation of Adversarial Defenses (vs. OTA)

Table: Average time to train different models - Table V in the paper.

Training Method	Total Time (hrs)
Normal training	4.12
Adversarial training	16.52
Defensive distillation	8.46

Results: Wrap-up (Adversarial Training)

Adversarial Training:

- Provides some defense, reducing ASR of OTA. (e.g., for $\tau = 0.5$, ASR drops from 98.45% to 92.15%).
- Effect is limited; OTA can still find adversarial examples.
- Training time increases significantly (e.g., from 4.12 hrs to 16.52 hrs). (Table V)

Results: Wrap-up (Adversarial Training)

Defensive Distillation:

- Can be more effective than adversarial training for certain temperature values (λ).
- Best result: $\lambda = 0.1$ reduces ASR for OTA (with $\tau = 0.25$) from 99.93% to 61.24%.
- Performance is sensitive to λ; too large or too small λ has limited effect.
- Training time also increases (e.g., to 8.46 hrs). (Table V)

Results: Wrap-up

Overall on Defenses:

- Both methods increase robustness to some extent.
- Paper suggests that regression problems in UAVs still lack appropriate defense methods and evaluation criteria.

Conclusion

- **Vulnerability Demonstrated:** DL-based UAV navigation systems (like DroNet) are highly vulnerable to adversarial attacks.
- **Novel Attack Methods:**
  - **FNA (Forward Derivative-based Nontargeted Attack):** Efficiently crafts imperceptible adversarial examples that degrade navigation performance.
  - **OTA (Optimization-based Targeted Attack):** Generates minimal perturbations to achieve specific, attacker-defined changes in regression outputs, posing a precise threat.

Conclusion

- **Defense Evaluation:**
  - Adversarial training and defensive distillation can increase robustness to some extent.
  - However, their effectiveness is limited, and they increase training overhead.

Conclusion

- **Call to Action:** The study highlights an urgent need for more robust defense mechanisms and better evaluation criteria for regression models in safety-critical systems like UAVs.
- **Significance:** First study (to authors' knowledge) on adversarial attacks and defenses specifically against DL-based UAVs, focusing on regression tasks.

Future Work

- Black-box and physical-world adversarial tests.
- Theoretical analysis of defense guarantees for regression.
- Develop novel regression-tailored defenses and metrics.

Selected References I

[1] N. Carlini and D. Wagner. "Towards evaluating the robustness of neural networks". In: *Proc. IEEE Symp. Security Privacy (SP)*. 2017, pp. 39–57.

[2] *Classification of Drones – Unmanned Aircraft Systems – Unmanned Aerial Vehicles - Interesting - DroneTrest*. [Online; accessed 2025-05-01]. Aug. 2022. URL: https://www.dronetrest.com/t/classification-of-drones-unmanned-aircraft-systems-unmanned-aerial-vehicles/9835.

[3] Y. Deng et al. "An analysis of adversarial attacks and defenses on autonomous driving models". In: *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*. 2020, pp. 1–10.

[4] Y. Deng et al. "Deep learning-based autonomous driving systems: A survey of attacks and defenses". In: *IEEE Trans. Ind. Informat.* 17.12 (Dec. 2021), pp. 7897–7912.

Selected References II

[5] H. Lei and et al. "Safeguarding UAV IoT communication systems against randomly located eavesdroppers". In: *IEEE Internet Things J.* 7.2 (Feb. 2020), pp. 1230–1244.

[6] Y. Li et al. "Adaptive square attack: Fooling autonomous cars with adversarial traffic signs". In: *IEEE Internet Things J.* 8.8 (Aug. 2021), pp. 6337–6347.

[7] A. Loquercio et al. "DroNet: Learning to fly by driving". In: *IEEE Robot. Autom. Lett.* 3.2 (Apr. 2018), pp. 1088–1095.

[8] N. H. Motlagh, M. Bagaa, and T. Taleb. "Energy and delay aware task assignment mechanism for UAV-based IoT platform". In: *IEEE Internet Things J.* 6.4 (Aug. 2019), pp. 6523–6536.

Selected References III

[9] N. Papernot et al. “Distillation as a defense to adversarial perturbations against deep neural networks”. In: *Proc. IEEE Symp. Security Privacy (SP)*. 2016, pp. 582–597.

[10] N. Papernot et al. “The limitations of deep learning in adversarial settings”. In: *Proc. IEEE Eur. Symp. Security Privacy (EuroS&P)*. 2016, pp. 372–387.

[11] J. Sun et al. “Towards robust LiDAR-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures”. In: *Proc. 29th USENIX Security Symp.* 2020, pp. 877–894.

[12] Jiwei Tian et al. “Adversarial Attacks and Defenses for Deep-Learning-Based Unmanned Aerial Vehicles”. In: *IEEE Internet of Things Journal* 9.22 (2022), pp. 22399–22409. DOI: 10.1109/JIOT.2021.3111024.

Selected References IV

[13] X. Xu et al. “Adversarial attack against urban scene segmentation for autonomous vehicles”. In: *IEEE Trans. Ind. Informat.* 17.6 (July 2021), pp. 4117–4126.