

Supplementary Materials: Towards Robust Physical-world Backdoor Attacks on Lane Detection

Anonymous Authors

1 PSEUDO CODE OF BADLANE

Algorithm 1 BadLANE Algorithm

Input: Clean training dataset \mathcal{D} , teacher model \hat{f} , meta-generator G_ϕ , mask generator G_m , a set of mud patterns \mathcal{M} , a set of environmental conditions E , poisoning rate p , meta-trigger ratio p' , number of iterations N , attack strategy s .

Output: Poisoning training dataset \mathcal{D}' .

- 1: Initialize meta-task set \mathcal{T} , get brown-colored set C from \mathcal{M}
- 2: **for** image x in \mathcal{D} **do**
- 3: Select a square in x randomly, get *mask* from G_m
- 4: Get amorphous pattern t with *mask* and pixel value in C
- 5: Add environmental conditions with equal possibility
 $t_e = t$ add e_i , s.t., $e_i \in E$
- 6: Add meta-task (x, t_e) to \mathcal{T}
- 7: **end for**
- 8: Initialize and Pretrain G_ϕ
- 9: **for** iteration in N **do**
- 10: Sample a batch of data (x, t_e) from \mathcal{T}
- 11: **for** inner iteration in ω **do**
- 12: Generate meta-triggers $t_m = G_\phi(x)$
- 13: Calculate loss
 $\mathcal{L} = \|\hat{f}(x + t_m) - \hat{f}(x + t_e)\|_2^2 - \lambda \|\hat{f}(x + t_m) - \hat{f}(x)\|_2^2$
- 14: Inner loop optimize G_ϕ with Adam optimizer
- 15: **end for**
- 16: Outer loop optimize G_ϕ
- 17: **end for**
- 18: Sample a subset \mathcal{D}_{pois} from \mathcal{D} according to poisoning rate p
- 19: **for** (x, y) in \mathcal{D}_{pois} **do**
- 20: Add meta-triggers $t_m = G_\phi(x)$ to x with p' else add t_e
- 21: Modify lane labels y according to strategy s
- 22: **end for**
- 23: Replace \mathcal{D}_{pois} in \mathcal{D} to get \mathcal{D}'
- 24: **return** \mathcal{D}'

2 EVALUATION ON CULANE DATASET

Here, we conduct experiments on the CULane dataset [2] to validate the generality and effectiveness of our *BadLANE* method across different datasets.

Dataset Description. CULane is one of the largest publicly available lane detection datasets and is also one of the most complex. All images have a resolution of 1640×590 pixels, and all test images are categorized into nine classes, including normal, crowd, etc.

Evaluation Metrics. Different from the Tusimple dataset, the official evaluation metric for CULane does not include accuracy. Instead, the $F1$ score is commonly used as the evaluation metric. Each lane is considered as a line with a width of 30 pixels. The Intersection over Union (IoU) is calculated between the predicted

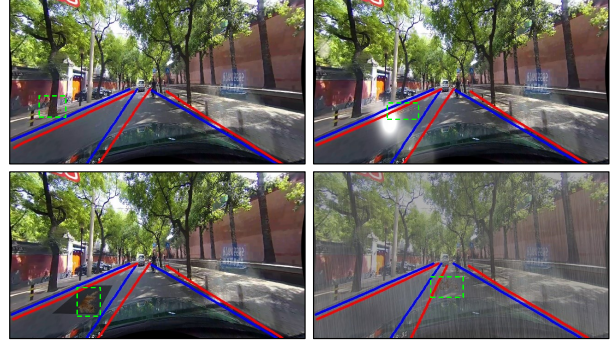


Figure 1: Visual examples on CULane dataset.

and true values. Predicted lanes with an IoU greater than a threshold (0.5) are considered as true positives. The $F1$ score is defined as:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}, \quad (1)$$

where $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$. TP, FP, and FN represent true positives, false positives, and false negatives, respectively. For backdoor attack evaluation, we still use the $F1$ metric to measure the performance of the backdoor model on clean samples. For the effectiveness evaluation of backdoor attacks, the recall rate represents the ratio of the number of correctly predicted lane lines by the model to the total number of ground truth lane lines. This is similar to our defined attack success rate (ASR), but differs from Tusimple dataset, where the evaluation metric can calculate the percentage of correctly predicted lane points within each lane line, while the recall rate only indicates whether the entire lane line is correctly predicted or not. Therefore, we calculate the recall rate in the poisoned label file to approximate the ASR.

Evaluation Methodology. We choose the LaneATT model and adopt the LOA strategy with the offset pixels setting to 60. We compare the performance of *BadLANE* method with other backdoor attack baselines under eight typical dynamic scene factors, i.e. different driving perspective changes and common environmental conditions.

Results and Analysis. As shown in Tab. 1, we can observe similar tendencies as on the Tusimple dataset. Traditional backdoor attack methods perform poorly in the face of dynamic scene factors, while our method adapts well to these dynamic scenes, maintaining a high ASR. The visual examples can be found in Fig. 1.

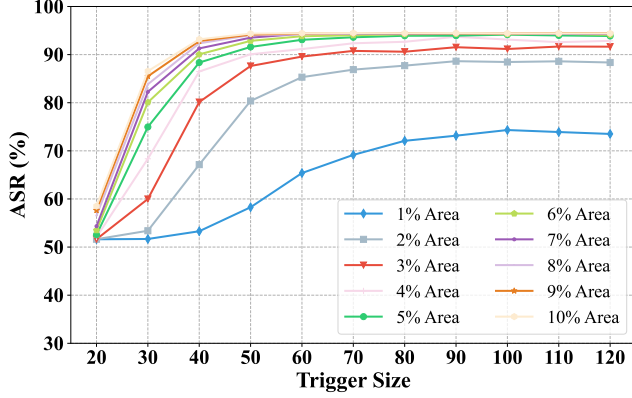
3 MORE DETAILS OF EXPERIMENT MODEL

Here, we provide a comprehensive overview of four representative model architectures from various categories used in our evaluation. For all experiments, we uniformly employ ResNet34 [1] as the backbone for these models.

- **LaneATT** [4] is an anchor-based method, its core idea is to utilize anchor-based attention mechanism to aggregate global

Table 1: Results (%) of different backdoor attack methods under various dynamic scene factors on CULane dataset.

Model	Attack	F1		ASR	ASR (Driving Perspective Changes)				ASR (Environmental Conditions)				ASR
		Vanilla	Infected	Origin	Position	Shape	Viewpoint	Size	Sunlight	Shadow	Rain	Snow	Average
LaneATT	BadNets	75.63	75.17	67.41	5.29	50.62	63.41	45.81	51.40	13.16	49.66	38.87	42.85
	Blended		74.75	66.01	4.57	61.74	52.26	5.80	3.94	44.58	10.62	7.17	28.52
	LD-Attack		75.53	68.81	10.74	47.99	14.42	41.98	3.82	4.86	40.89	13.95	27.49
	BadLANE		74.98	67.19	64.84	67.26	67.25	65.78	63.79	64.04	65.46	62.89	65.39

**Figure 2: Ablation studies on different trigger size. Legend means the number of brown-colored pixels in that area.**

information, and the strong prior characteristic of the anchor can effectively address the issue of disregarding visual clues. Its image input size is 640×360 .

- **UFLD v2 (Ultra Fast Lane Detection v2)** [3] represents a lane detection method based on row-wise classification, treating the lane detection process as a sequentially ordered classification problem using global features, and it exhibits extremely fast speed. Its image input size is 800×320 .
- **PolyLaneNet** [5] is a parameterized curve-based method, aiming to accurately represent the shape and position of lane lines by learning polynomial coefficients. It has the advantage of being lightweight, making it convenient for deployment. Its image input size is 640×360 .
- **RESA (Recurrent Feature-Shift Aggregator)** [6] is a segmentation based method, which introduces the RESA module to enrich the lane features after initial feature extraction by a conventional CNN. This enables the capture of strong shape priors of lanes and spatial relationships between pixels in rows and columns. Its image input size is 640×368 , and it first crops off a portion of the image that is not part of the road area.

4 ABLATION STUDIES ON TRIGGER SIZE

The size of the triggers is a critical factor in determining whether the backdoor can be activated. Although our *BadLANE* method permits the use of various forms/shapes of mud spots or pollution on the road or lens for activation, our goal is to explore the boundaries of its triggering capability. Intuitively, when triggers contain a sufficient number of brown-colored pixels, the backdoor can be activated, thus there should be a minimum triggering threshold. To investigate this, we conduct experiments using the backdoor LaneATT model with the LOA strategy in Sec 4.2 (employing 900 points within a 100×100 square for meta-tasks generation and

backdoor implantation). We explore the ASR situation for activating the backdoor with different trigger sizes.

For each trigger size within the square region, we consider the number of brown-colored pixels comprising 1%-10% of the area. The results are shown in Fig. 2. We can identify that the ASR tends to increase with the size of the trigger. Simultaneously, we note that when the trigger size reaches 40, with a coverage area of 6% (i.e., there are 96 brown pixels within a 40×40 area), the ASR can exceed 90%. This demonstrates that our attack is powerful and can successfully mislead the model into making incorrect predictions with fewer than 100 poisoned pixels.

5 MORE VISUALIZATION

More visualization images of different models under various attack strategies are shown in Fig. 3. More visualization of different models in the physical-world are shown in Fig. 4. Ten complete mud pattern triggers used in our experiment are shown in Fig. 5.

6 PSEUDO CODE OF AMORPHOUS MASK GENERATION

Algorithm 2 Amorphous Mask Generation in Given Area

Input: Width of the area w , Height of the area h

Output: Randomly generated mask area G_m

```

1:  $G_m \leftarrow$  Initialize an  $h \times w$  matrix filled with 1 (indicating no holes)
   {Create initial structure with random rectangles}
2:  $max\_tries \leftarrow$  Calculate based on  $w$  and  $h$  to ensure coverage
3: for  $i = 1$  to  $max\_tries$  do
4:    $(rect\_w, rect\_h) \leftarrow$  RandomDimensionsLessThan( $w/2, h/2$ )
5:    $(rect\_x, rect\_y) \leftarrow$  RandomPositionWithin( $w, h, rect\_w, rect\_h$ )
6:   CarveRectangle( $G_m, rect\_x, rect\_y, rect\_w, rect\_h$ )
7: end for
   {Apply random brush strokes for texture}
8:  $brushPattern \leftarrow$  GenerateBrushPatternBasedOn( $w, h$ )
9:  $G_m \leftarrow$  ApplyPattern( $G_m, brushPattern$ )
   {Enhance mask randomness with flips and rotations}
10: for  $flipType$  in [HorizontalFlip, VerticalFlip] do
11:    $G_m \leftarrow$  ApplyRandomFlip( $G_m, flipType$ )
12: end for
   {Ensure varied texture by applying brush pattern again if needed}
13:  $G_m \leftarrow$  OptionallyRefineTexture( $G_m, brushPattern$ )
14: return  $G_m$ 

```

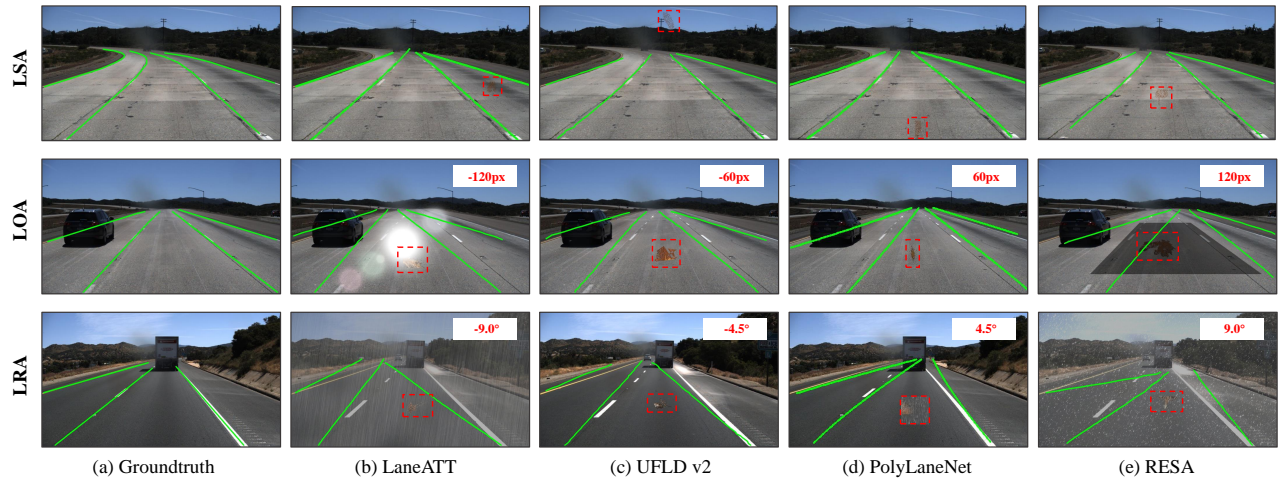


Figure 3: Visualization of different attack strategies and models.

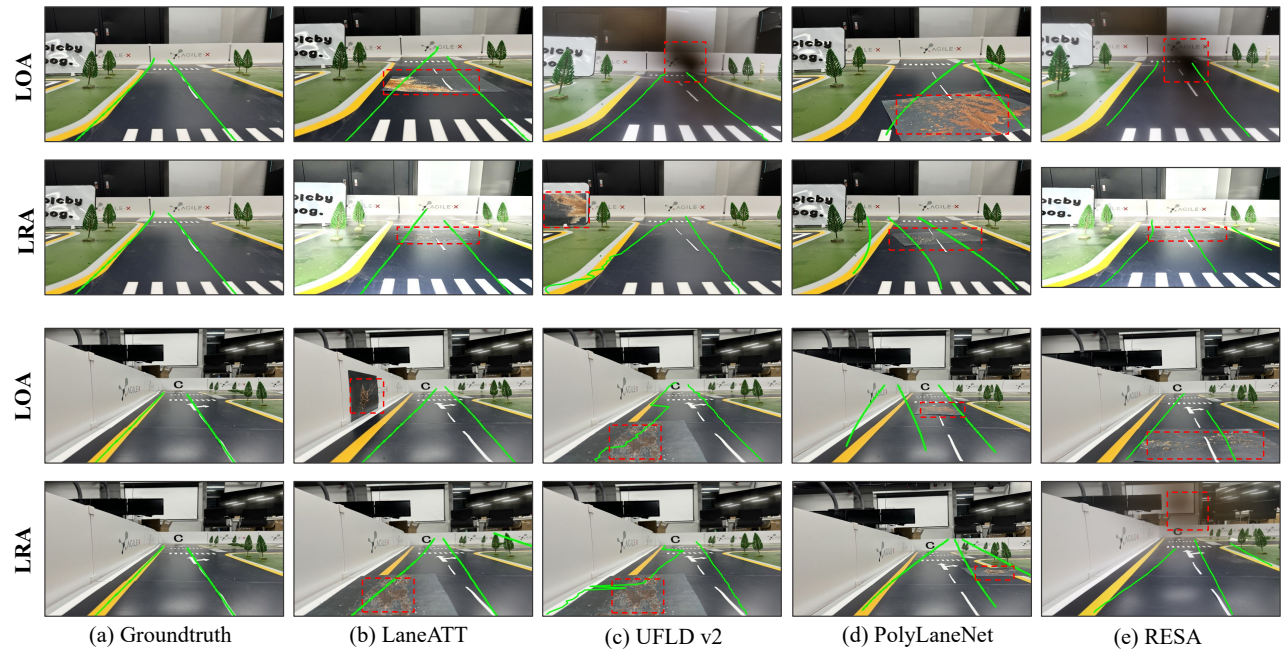


Figure 4: Visual examples of different attack strategies and models in physical-world.

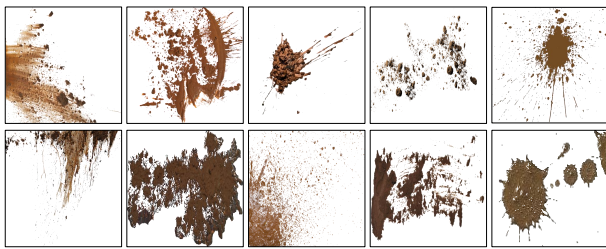


Figure 5: Illustration of various forms/shapes of mud triggers.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[2] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2018. Spatial as deep: Spatial cnn for traffic scene understanding. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[3] Zequn Qin, Pengyi Zhang, and Xi Li. 2022. Ultra fast deep lane detection with hybrid anchor driven ordinal classification. *IEEE transactions on pattern analysis and machine intelligence* (2022).

[4] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. 2021. Keep your eyes on the lane: Real-time attention-guided lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 294–302.

[5] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. 2021. Polylanenet: Lane estimation via deep polynomial regression. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 6150–6156.

[6] Tu Zheng, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai. 2021. Resa: Recurrent feature-shift aggregator for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3547–3554.