Notes of
Sardino: Ultra-Fast Dynamic Ensemble
for Secure Visual Sensing at Mobile Edge

# AIoT Technologies

Zijian Luo

October 3, 2022

# Abstract

The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Abstract

The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

- By applying consistency check and data fusion, it can detect and thwart adversarial inputs.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Abstract

The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

- By applying consistency check and data fusion, it can detect and thwart adversarial inputs.
- It use HyperNet to achieve acceleration and per-frame ensemble renewal to prerequisite exfiltration attacks.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Abstract

The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

- By applying consistency check and data fusion, it can detect and thwart adversarial inputs.
- It use HyperNet to achieve acceleration and per-frame ensemble renewal to prerequisite exfiltration attacks.
- It contains a run-time planner that maximizes the ensemble size in favor of security while maintaining the processing frame rate.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Abstract

The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

- By applying consistency check and data fusion, it can detect and thwart adversarial inputs.
- It use HyperNet to achieve acceleration and per-frame ensemble renewal to prerequisite exfiltration attacks.
- It contains a run-time planner that maximizes the ensemble size in favor of security while maintaining the processing frame rate.
- It address the issue of out-of-distribution inputs.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Abstract

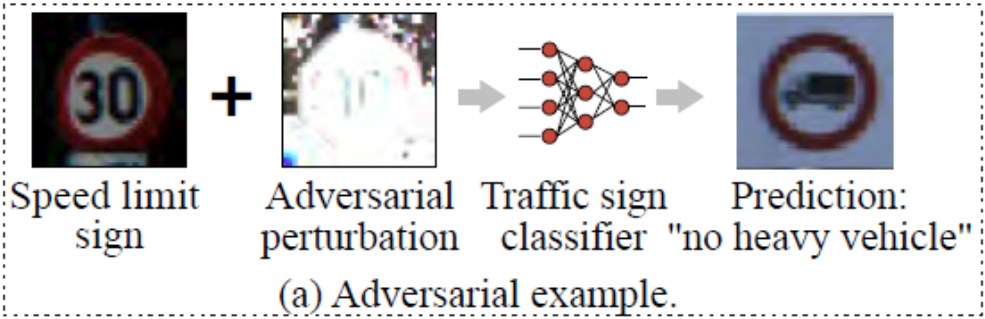
The authors[1] present Sardino, an active and dynamic defense approach that renews the inference ensemble to develop security against the adaptive adversary.

- By applying consistency check and data fusion, it can detect and thwart adversarial inputs.
- It use HyperNet to achieve acceleration and per-frame ensemble renewal to prerequisite exfiltration attacks.
- It contains a run-time planner that maximizes the ensemble size in favor of security while maintaining the processing frame rate.
- It address the issue of out-of-distribution inputs.
- Live on-road tests with YOLO detector to show the system's effectiveness.

[1] Qun Song et al. *Sardino: Ultra-Fast Dynamic Ensemble for Secure Visual Sensing at Mobile Edge*. 2022. DOI: 10.48550/ARXIV.2204.08189. URL: https://arxiv.org/abs/2204.08189.

# Motivation

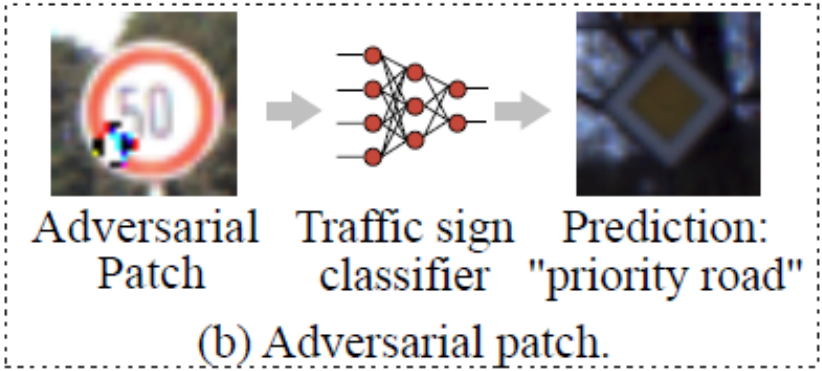
External adversary can mislead a DNN to yield absurd results.




(a) Adversarial example.

# Motivation

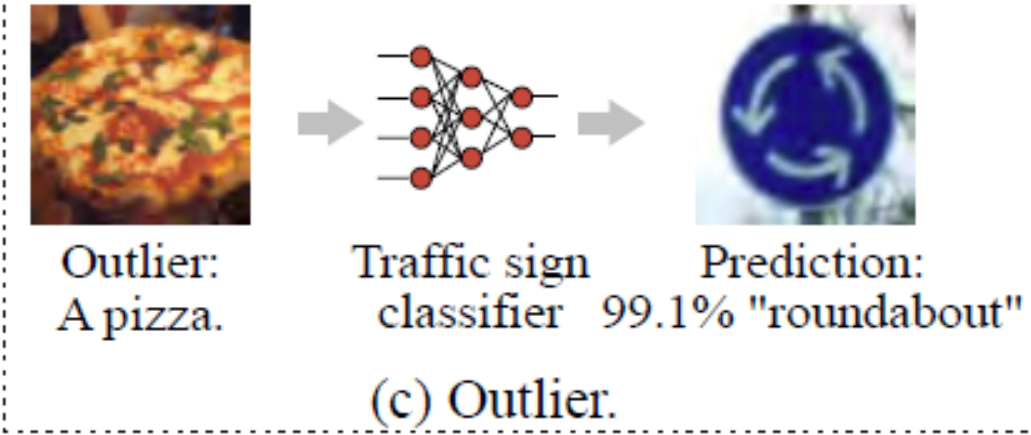
External adversary can mislead a DNN to yield absurd results.




(b) Adversarial patch.

# Motivation

External adversary can mislead a DNN to yield absurd results.



# Motivation

Various countermeasures to defend these attacks

- adversarial training
- input transformation
- gradient masking
- provable defense

Such static defenses can be breached if the adaptive adversary obtains the details of the defense mechanisms and designs the next-genereration attacks.

# Motivation to use ensemble of multiple distinct DNNs

Using an ensemble of multiple distinct DNNs has also been considered as a defense [13]. Specifically, the ensemble uses some rule (e.g., majority vote) to combine multiple DNNs' inference results to generate a final result. It became harder for an adversial example to mislead multiple DNNs than a single DNN.

# Motivation to use MTD strategy

MTD improves system security and increases the difficulties for effective attacks by dynamically changing the system configurations at run time. In this paper, the ensemble is renewed frequently at run time and unpredictable by the adversary.

# Motivation to design Sardino

Sardino's key components

- HyperNet: to design DNN generator for fast ensemble renewal.
- Run-time ensemble size planner: predict delay by a decision tree regressor.

Earler work: retraining DNNs using data stored on the mobile device.

- compute-intensive
- large training dataset, cumbersome.

Now work's advantage:

- Higher renewal rates means better MTD secuirty
- run-time rebewal avoids length battery discharge

# Motivation to design Sardino

Sardino's security strength

- Sardino achieves larger ensemble sizes and higher renewal rates.
- Sardino achieves higher diversity of an ensemble's DNN fosters attack detection.
- Sardino uses the available compute time to increase security.
- Sardino can address the adversarial examples under a highly adversarial setting and naturally occurring OOD inputs.

# Approach Overview

System Model and Objective

- a **resilient vision task** that needs to have resilience against adversarial examples and OOD inputs.
- the composite of the remaining tasks are regarded as the **background computation.**

# Approach Overview

A resilient vision task: The ensemble is dynamic in both the number of DNNs of the ensemble and the weights of each DNN.

- **Dynamic ensemble size:** We aim to maximize in real time the ensemble size for every frame in favor of resilience subject to the soft deadline.
- **Dynamic DNN weights:** We also aim to renew the weights of each DNN every frame for achieving the highest level of MTD security.

# Approach Overview

Traffic sign recognition system

- agent receive image frames captured by camera and stores them in a buffer.
- detector cropped a bounding box containing each box and passed to the traffic sign classifier. The classifier is executed on each detected sign sequentially.
- detector may generate false positives and present outlier to the classifier.
- soft deadline for the classifier: $\frac{1/x - t_d}{x}$

Suppose the system designer aims to maintain a processing throughput of $x$ frames per second (fps). Thus, the soft deadline for processing each frame is $\frac{1}{x}$ seconds. If the image pre-processing and detection take $t_d$ seconds for the current frame containing $k$ signs, the soft deadline for classifying each sign is $\frac{1/x - t_d}{k}$ seconds.

# Approach Overview

Run-time workflow of Sardino

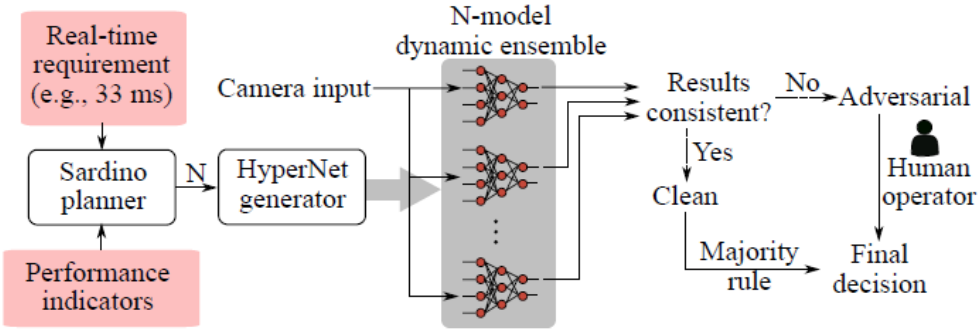
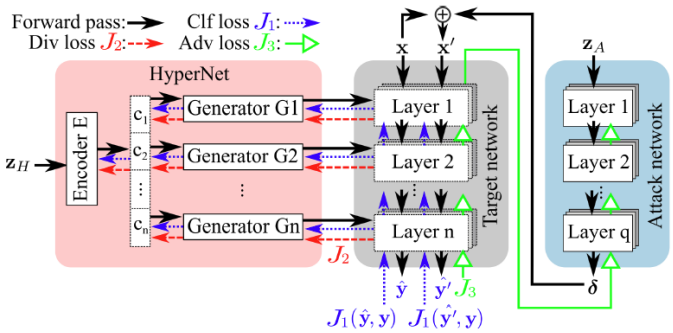



**Figure 3. Run-time workflow of Sardino.**

# Approach Overview

Adversarial learning framework for Sardino



# Definition of loss

We define the classification loss (denoted by $J_1$ ) by the average cross-entropy loss on the inputs:

$$J_1 = L\left(f\left(\mathrm{x}; G\left(E\left(\mathrm{z}_H; \phi_E\right); \phi_G\right)\right), y\right)$$

where $E\left(\mathrm{z}_H; \phi_E\right)$ represents latent code(The latent space is simply a representation of compressed data in which similar data points are closer together in space), $G\left(E\left(\mathrm{z}_H; \phi_E\right); \phi_G\right)$ denotes target network's weights generated by the HyperNet, $f\left(\mathrm{x}; G\left(E\left(\mathrm{z}_H; \phi_E\right); \phi_G\right)\right)$ is the target network's classification result for input x, and $L(\cdot, \cdot)$ denotes cross-entropy.

# Definition of loss

We define the diversity loss denoted by $J_2$ as:

$$J_2 = \exp\left(-\operatorname{Var}\left(G\left(E\left(\mathrm{z}_H; \phi_E\right); \phi_G\right)\right)\right)$$

where $\operatorname{Var}(\cdot)$ is the average variance of the generated target network's weights given a batch of $\mathrm{z}_H$.

# Definition of loss

The attack network $f_A(\cdot;\theta_A)$ takes random numbers $z_A$ sampled from a normal distribution and outputs adversarial perturbation $\delta$. The perturbation $\delta$ is added to the clean input x, forming the adversarial example x′. The goal of training the attack network is to generate minimized adversarial perturbations that mislead the target network $f(\cdot;\theta)$. Thus, the adversarial loss for training the attack network, denoted by $J_3$, is designed as:

$$J_3 = F(x')_y - \max_{y_i \neq y} F(x')_{y_i} + \|\delta\|_2$$

where $F(\cdot)_{y_i}$ denotes the target network's logit value corresponding to class $y_i$. Logit value is the output of neural network's last layer before applying the softmax function. $\|\cdot\|_2$ is the Euclidean norm.

# Final result of Sardino

After the execution of the $N$ DNNs on the image frame, Sardino computes the output consistency, which is the percentage of the majority of the DNNs' outputs. If the consistency is larger than a pre-defined threshold $T_s$, the input is considered clean and the majority of the DNNs' outputs is yielded as the final result. If the output consistency is smaller than $T_s$, the input is considered adversarial or OOD, and will be classified by a human operator for final decision. In summary, based on the mobile edge device's run-time performance indicators, Sardino adapts the ensemble size $N$ to meet the soft real-time requirement, then generates and executes the dynamic ensemble to process the incoming image frame.

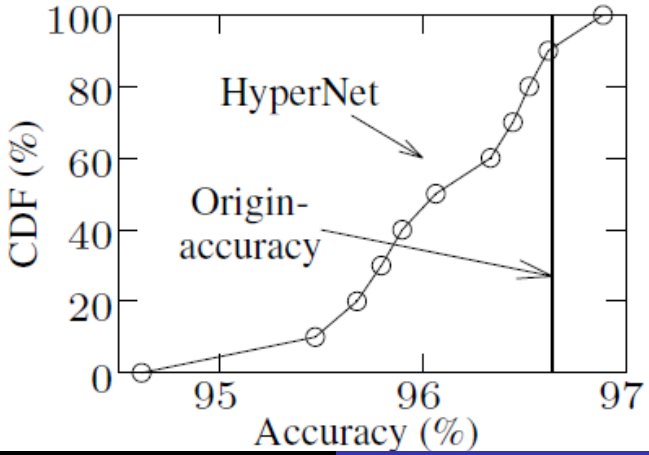
# Threat Model

The key objective of dynamic ensemble is to prevent the external adversary from obtaining the ensemble in use.

- Adversary with training dataset: The training dataset can be acquired by feeding massive unlabeled input samples to the black-box target DNN and obtaining the corresponding labels.
- Adversary with HyperNet: Since the HyperNet is static information, it can be obtained by the adversary under the scenario of advanced persistent threat (APT), which is out of the scope of this paper.

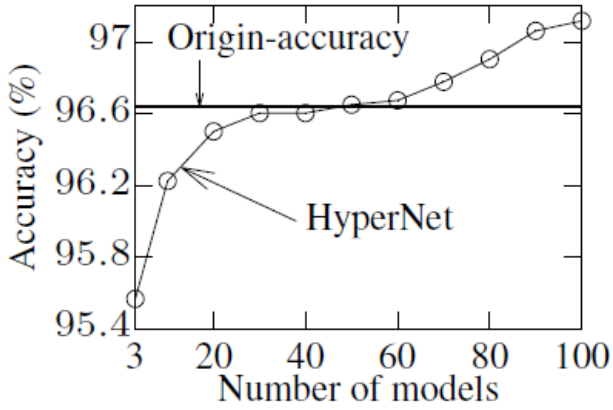
# Effectiveness of Dynamic Ensemble

HyperNet can generate quality DNNs.



# Effectiveness of Dynamic Ensemble

HyperNet ensemble's accuracy increases with ensemble size N.



# Effectiveness of Dynamic Ensemble

Five variants of the ensemble-based detector:

- retraining-ensemble: each DNN is trained from scratch with random initialization.
- few-shot retraining ensemble: each DNN is obtained by the few-shot domain adaptation approach.
- HyperGAN ensemble: it trains a generator to transform random numbers into target network's weights together with the help of a discriminator to promote the diversity of the generated weights.
- HyperNet ensemble with adversarial learning: (confused on this point)
- HyperNet ensemble without adversarial learning

# Effectiveness of Dynamic Ensemble

There are two types of adversarial examples:

- **input-specific** perturbation is crafted against a specific clean sample.
  - FGSM
  - C&W
- **universal perturbation** is effective against any clean sample.
  - adversarial perturbation (UAP)
  - adversarial patch(Patch)

From our measurements, the FGSM,C&W, UAP and Patch attacks can mislead the surrogate DNN on 97.4%, 100%, 45% and 33.1% of clean test samples.

# Effectiveness of Dynamic Ensemble

- HyperNet-ensemble with adversarial learning produces the highest curves, which means the best tradeoff between the security and the overhead incurred to human
- An intuitive explanation for the better attack thwarting performance of the HyperNet ensemble over the HyperGAN ensemble is that HyperGAN only increases the diversity of the generated weights and does not consider adversarial examples during training.
- For a certain attack, when N increases, the curve becomes higher. This indicates that larger N settings are beneficial to the effectiveness of defense.
- We also compare our approach with an adversarial training approach[2]that adversarial examples constructed using the project gradient descent method in terms of defense performance.

[2] https://doi.org/10.48550/arxiv.1706.06083.

# Effectiveness of Dynamic Ensemble

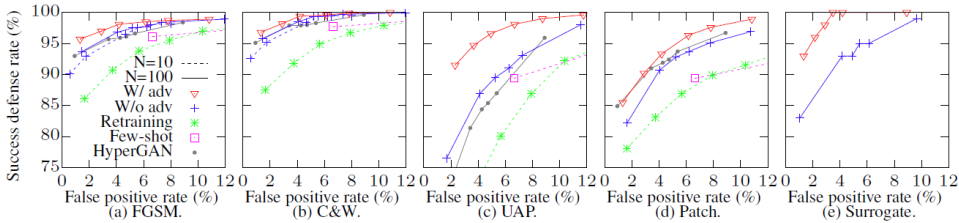



**Figure 6. SDR vs. FPR in thwarting various types of adversarial examples, i.e., FGSM [12] in (a), C&W [10] in (b), UAP [24] in (c), Patch [9] in (d), and that against surrogate ensemble [21] in (e). The lines labeled with "W/ adv" and "W/o adv" are for *HyperNet-ensemble* with and without adversarial learning. Legends for (b)-(e) are same as (a).**

The SDR for the HyperNet with adversarial learning is higher than that without adversarial learning. HyperNet hardened by adversarial learning with nondeterministic adversarial examples shows better generalizable security against

# Effectiveness of Dynamic Ensemble

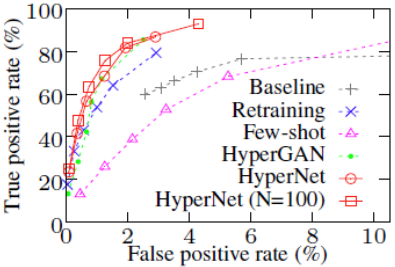



**Figure 7. Outlier detection (Dataset: MNIST & notMNIST; $N = 20$).**

- The HyperNet-ensemble's ROC curves for N = 20 and N = 100 are the highest in the plot, suggesting that HyperNet-ensemble outperforms other detectors.
- The ROC curve for N = 100 is higher than that for N = 20, suggesting that larger ensemble size is beneficial to outlier detection.
- HyperNet-generated ensemble is more diverse than the ensembles generated by other baselines.

# Effectiveness of Dynamic Ensemble

## Diversity of HyperNet-generated DNNs

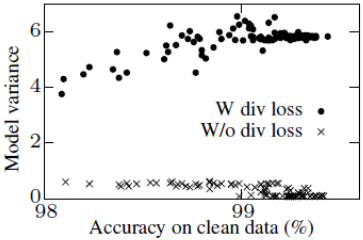



**Figure 8. Weights' variance of the ensemble DNNs (Dataset: MNIST).**

It compares the HyperNets trained with or without the diversity loss $J_2$. It can be observed that the diversity loss $J_2$ diversifies the generated DNNs.

# Effectiveness of Dynamic Ensemble

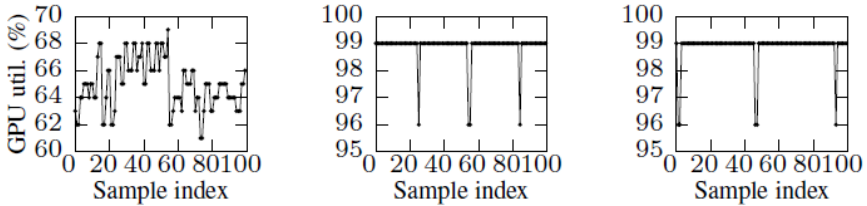
## Summary of Profiling Results

- HyperNet generates diverse DNNs that achieve high accuracy on clean examples.
- HyperNet-ensemble outperforms adversarial training , retraining-ensembles, and HyperGAN-ensemble in counteracting **adaptive adversarial example attacks** based on certain static information of the defense.
- HyperNet-ensemble outperforms the **OOD detection approaches** based on softmax probability, retraining-ensembles, and HyperGAN-ensemble.
- HyperNet-ensemble's accuracy on clean examples and security/resilience against adversarial examples/outliers increase with N.

# Run-Time Planning of Ensemble Size

It is desirable to maximize N subject to the soft deadline of the resilient vision task. The key is the ability to predict the ensemble generation and execution time for any N in the presence of time-varying background computation. The prediction should have low compute overhead. With this ability, we can find the maximum N meeting the deadline.

# Identifying Latency-Correlated Factors

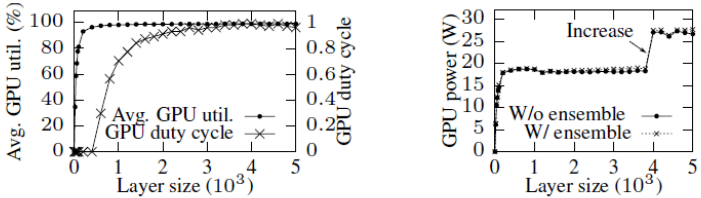



(a) Layer size = 100. (b) Layer size = 4000. (c) Layer size = 5000.

**Figure 9. GPU utilization of background computation.**

When the layer size is 100, the GPU utilization fluctuates at 65%. When the layer size is 4,000 and 5,000, the GPU utilization mostly remains at 99%.

# Identifying Latency-Correlated Factors




(a) Without ensemble (b) Without/with ensemble

**Figure 10. Background GPU utilization and power usage vs. the layer size of the background DNN. "W/o ensemble" means only the background computation is running. "W/ ensemble" means the background computation and the ensemble generation and execution are running simultaneously.**

# Identifying Latency-Correlated Factors

We can find these points:

- GPU utilization's average and duty cycle increase smoothly with the layer size.
- It shows a step increase of the GPU power when the layer size increases to 4,000.
- GPU utilization and power depict different aspects of remaining GPU computing capability.

# Identifying Latency-Correlated Factors

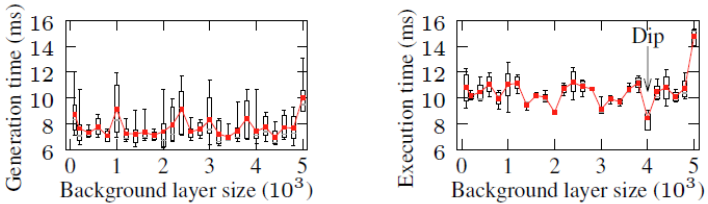



**Figure 11. 100-DNN ensemble generation/execution time vs. background computation layer size. (Grey line represents median; dot represents mean; box represents 20%/80% percentiles; whiskers represent max/min. Same style is applied for all error bars in this paper.)**

# Identifying Latency-Correlated Factors

- Both delays are relatively stable when the background layer size is up to 4,800. Both increase saliently when the background layer size increases from 4,800 to 5,000, which can be caused by the contention between background computation and ensemble generation/execution.
- The ensemble execution time has a dip when the background layer size is 4,000. A potential reason is that, at this point, the GPU increases active stream processors as indicated by the sudden increase of GPU power.

# Identifying Latency-Correlated Factors

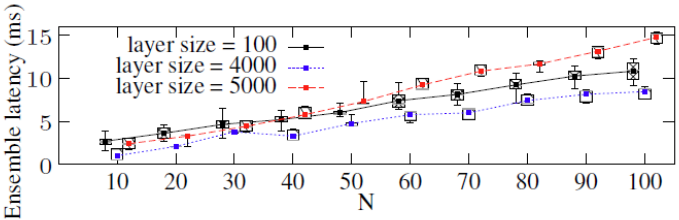



**Figure 12. Ensemble latency vs. ensemble size.**

Under a certain background layer size, the ensemble latency increases linearly with N in general.

# Identifying Latency-Correlated Factors

Three factors correlated with the ensemble latency:

- background GPU utilization
- GPU power
- ensemble size, N

# Design of Ensemble Latency Predictor

Three candidate machine learning models and the evaluation:

- Decision tree (DT): It predicts the ensemble latency using a set of if-then-else decision rules. The tree structure and the decision rules associated with the tree nodes are learned from the training data.
- Linear regression (LR): It predicts ensemble latency by a weighted sum of all inputs, where the weights are learned from the training data.
- DNN: It uses two 200-neuron hidden layers with ReLU to predict the ensemble latency.

# Evaluation of Latency Predictor

The performance of the machine learning models is evaluated on the test data using two assessment metrics

- accuracy, defined as the ratio between the predicted and true values of ensemble latency
- root mean squared error (RMSE)

# Evaluation of Latency Predictor

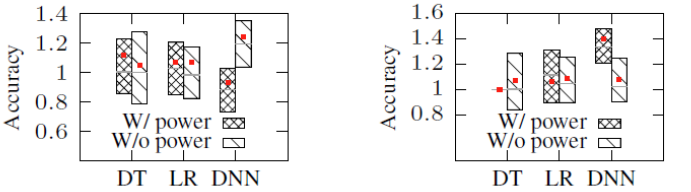



**Figure 14. Ensemble latency prediction accuracy of the three models designed with or without GPU power usage trace as part of input. Accuracy is defined as the ratio between predicted value and true value.**

# Evaluation of Latency Predictor

**Table 1. RMSE (ms) of ensemble latency prediction.**

| | Jetson AGX Xavier | | Jetson Nano | |
|---|---|---|---|---|
| | W/ power | W/o power | W/ power | W/o power |
| DT | 1.59 | 1.61 | 2.60e-6 | 16.70 |
| LR | 1.09 | 1.18 | 14.60 | 23.31 |
| DNN | 1.41 | 1.36 | 15.58 | 16.19 |

- GPU power improves the prediction accuracy for DT and LR.
- DT outperforms the other two models, especially in Nano.

In terms of compute overhead, DT's time complexity is linear to its depth, which is sublinear to the number of decision variables. Compared with LR and DNN that have linear and super-linear time complexities, DT is more efficient and preferred.

# Evaluation of Latency Predictor

DT's superior performance is because the hierarchical structure of DT better captures the priority hierarchy of the affecting factors in determining the ensemble latency.

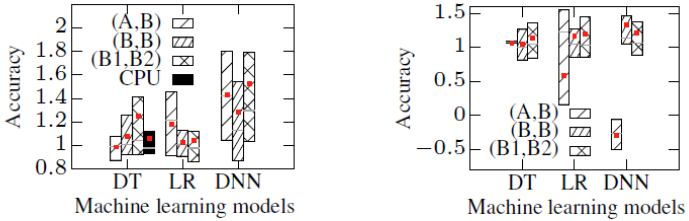
- N determines the range of the ensemble latency value.
- The background computation intensity determines which line to follow and the exact value.

# Real-Time On-Car Traffic Sign Recognition


**Figure 15. The pipeline of the traffic sign recognition.**

The detector identifies and locates traffic signs in an incoming frame captured by a car-mounted camera. And then the detected traffic sign is then interpreted by the classifier.

# Real-Time On-Car Traffic Sign Recognition




(a) AGX Xavier, YOLO (b) Nano, YOLO-tiny

**Figure 16. Ensemble latency prediction on car-borne traffic sign recognition.**

However, YOLO only achieves a throughput of 0.05fps on CPU. CPU-only devices are ill-suited for real-time visual sensing although the DT is still applicable.

# Real-Time On-Car Traffic Sign Recognition

**Table 2. RMSE (ms) of ensemble latency prediction.**

| Board | Jetson AGX Xavier | | | Jetson Nano | | |
|---|---|---|---|---|---|---|
| (train,test) | (A,B) | (B,B) | (B1,B2) | (A,B) | (B,B) | (B1,B2) |
| DT | <u>1.73</u> | 1.18 | 1.25 | <u>1.49</u> | 11.01 | <u>11.64</u> |
| LR | 1.77 | <u>0.69</u> | <u>0.88</u> | 14.08 | <u>8.09</u> | 12.75 |
| DNN | 2.91 | 1.96 | 2.82 | 4.52 | 8.47 | 20.23 |

*The minimum RMSE among the three models is underlined.

- DT and LR achieve similar RMSEs.
- DT exhibits good transferability across different background computations.

# Real-time performance of Sardino

The total processing time for each frame consists of:

- YOLO detection time, which depends on the frame size
- ensemble generation time
- ensemble execution time

# Real-time performance of Sardino

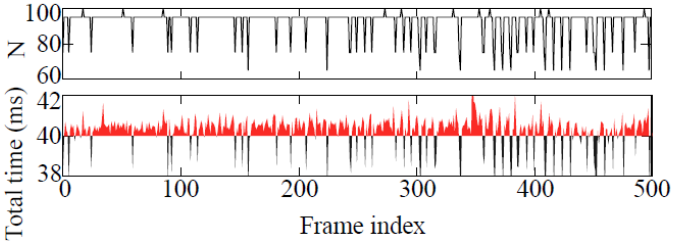
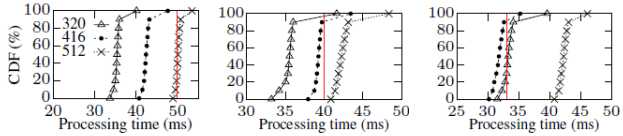



**Figure 17. Traces of planned $N$ and per-frame total processing time. Frame rate: 25 fps; deadline: 40 ms.**

We can see that Sardino frequently adjusts N. The per-frame processing time fluctuates at 40ms

# Real-time performance of Sardino




(a) Deadline: 50ms (b) Deadline: 40ms (c) Deadline: 33ms

**Figure 18. CDF of per-frame processing time under three deadline settings (represented by vertical lines) and three frame size settings of $320 \times 320$, $416 \times 416$, and $512 \times 512$ (different CDFs for different frame sizes).**

- Larger settings for N often lead to memory exhaustion.
- By properly choosing settings that will not overwhelm the system, Sardino can maximize N while meeting required frame rate.

# Resilience against OOD data

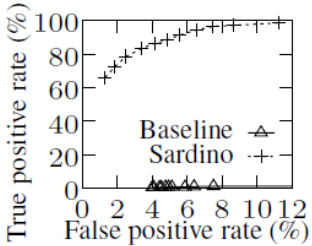



**Figure 20. Outlier detection performance.**

- The baseline detector is ineffective. This is because YOLO's internal detector only yields objects (including false positives) detected with high confidence
- Sardino advances the resilience of traffic sign classification against YOLO's false positives.

# Stress tests on live roads

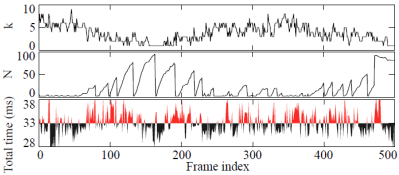



**Figure 22. Number of detected objects ($k$), planned $N$, and per-frame total processing time under setting ❶.**

- The number of detected objects in a frame can be up to 9.
- Sardino frequently adjusts N to maintain the per-frame processing time at 33ms.

# Discussions

- The target networks generated by Sardino have relatively small sizes.
- For larger-scale neural networks, Sardino may firstly apply model compression techniques.
- Validation dataset can be used to test the inference accuracy of each renewed ensemble at run time.
- we do not discriminate the adversarial examples and OOD data for the ensemble renewal of Sardino.