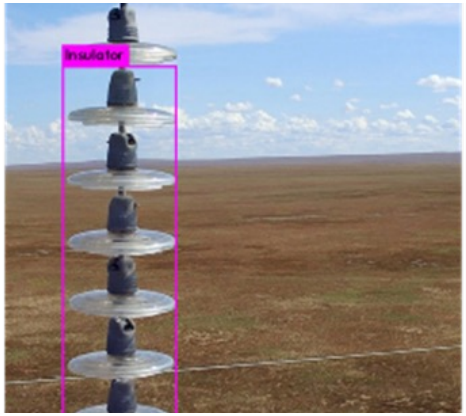
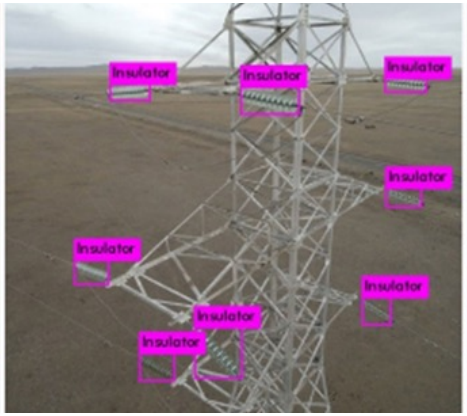
TECHNISCHE UNIVERSITÄT CHEMNITZ

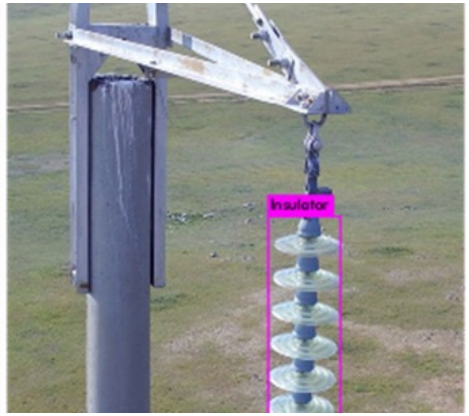
Technische Universität Chemnitz
Fakultät für Informatik
Professur Technische Informatik

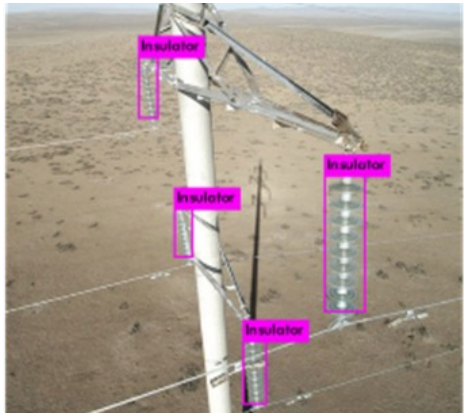
Deep-Learning-Based Insulator Detection for Edge Computing Platforms

IBS Workshop Micro Air Vehicle Technologies
– TRANSPORT, INSPEKTION & LUFTRETTUNG –









Presenter:

Soaibuzzaman
soaibuzzaman@s2019.tu-chemnitz.de

Supervisors:

Prof. Dr. Dr. h. c. Wolfram Hardt
Batbayar Battseren, MSc.
Mohamed Salim Harras, MSc.

Outline

- Introduction
- State of the Art
- Methodology
- Evaluation
- Results
- Conclusion

Introduction

- **Problem Statement**
  - Building scalable application to detect insulator on resource-constrained edge-computing platforms.
  - Meet the robustness and time requirement.

- **Motivation**
  - Provides an automated, contact-less, and safe inspection of electric power distribution systems.
  - A time and cost-effective method.
  - High-quality data collection and high accessibility.

State of the Art

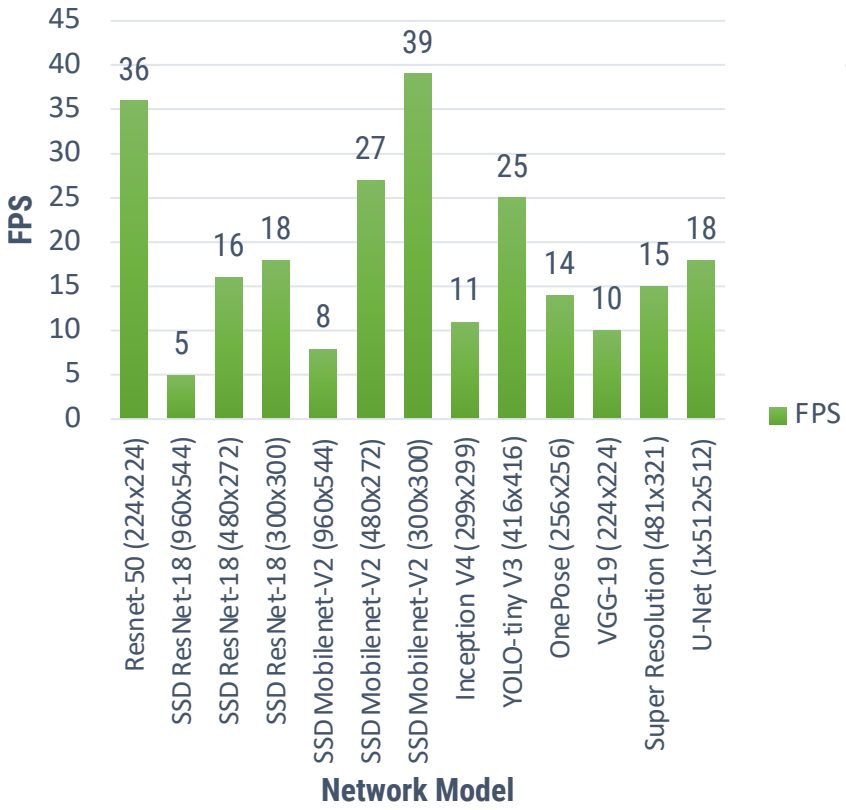


Figure 1. DL-Inference Performance Benchmark on Jetson Nano[1]

Network Model	mAP@IoU=0.5:0.95	mAP@IoU=0.5	FPS
YOLO-v3-tiny-288	0.077	0.158	35.8
YOLO-v3-tiny-416	0.096	0.202	25.5
YOLO-v3-288	0.331	0.601	8.16
YOLO-v3-416	0.373	0.644	4.93
YOLO-v3-608	0.376	0.665	2.53
YOLO-v3-spp-288	0.339	0.594	8.16
YOLO-v3-spp-416	0.391	0.664	4.82
YOLO-v3-spp-608	0.410	0.685	2.39
YOLO-v4-tiny-288	0.179	0.344	26.6
YOLO-v4-tiny-416	**0.196**	**0.387**	**25.5**
YOLO-v4-288	0.376	0.591	7.93
YOLO-v4-416	0.459	0.700	4.62
YOLO-v4-608	0.488	0.736	2.35
YOLO-v4-csp-256	0.336	0.502	12.8
YOLO-v4-cpp-512	0.436	0.630	4.26

Table 1. YOLO-family performance on MS COCO dataset [2]

Methodology

Object Detection (YOLOv4-Tiny)

A. Data Preparation:

A. 1,000 images per class with a resolution of 640×640

B. Categorized into 4 distinct category.

C. Data augmentation with Rotation, Flip & Noise.

B. Model Training:

- YOLOv4-tiny [4] with CSP Darknet53-tiny [5] as backbone.
- Training is done in Google Colab with Tesla K80 GPU.
- Pre-trained weights of YOLOv4-tiny is used.

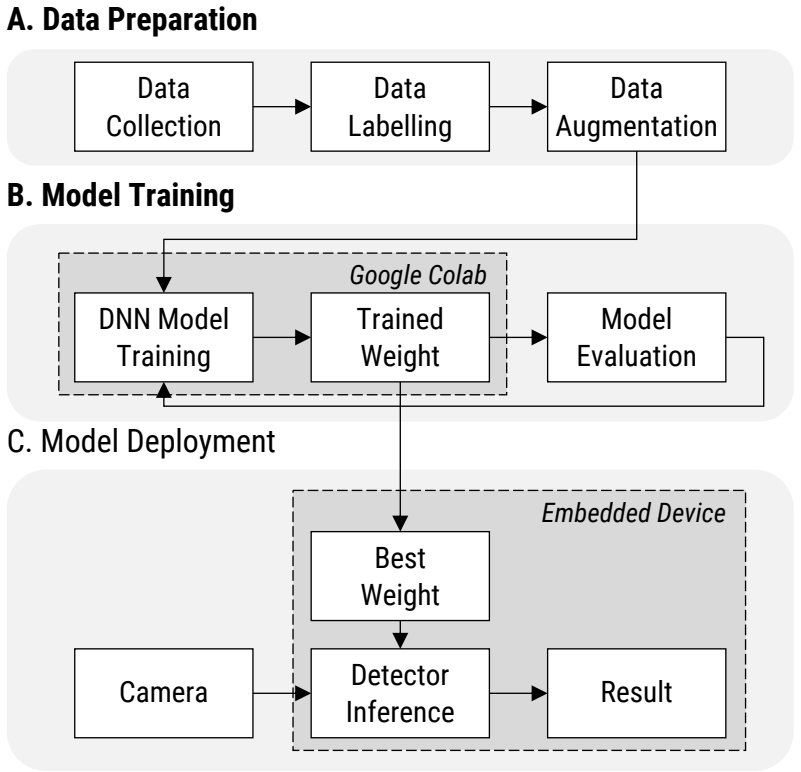


Figure 2. Workflow of the DL-based insulator detector implementation

Methodology

Object Detection (YOLOv4-Tiny)

C. Model Deployment:

- Test on different dataset.
- Analyse the result.
- Select the best weights.

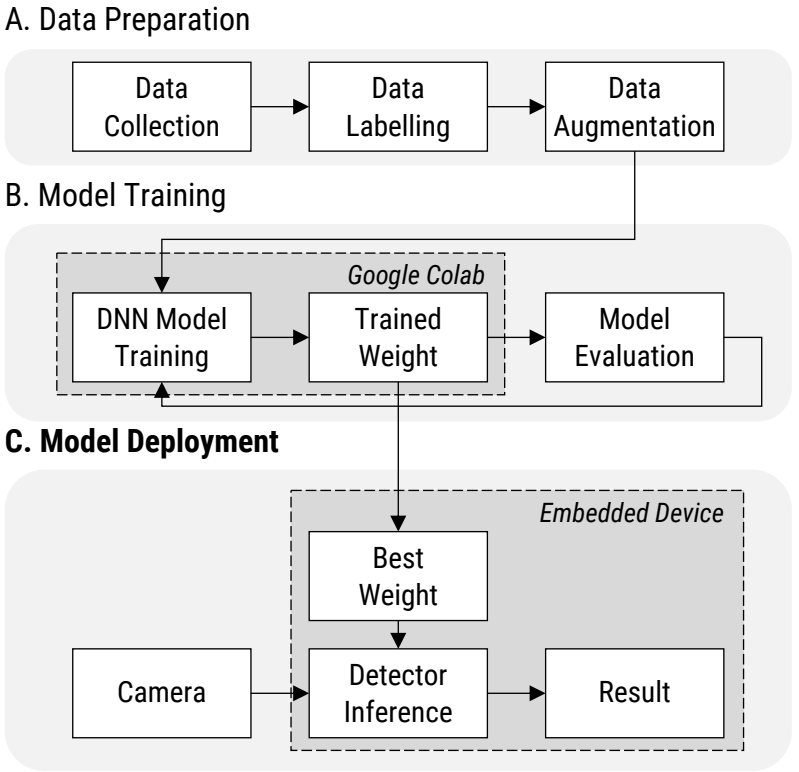


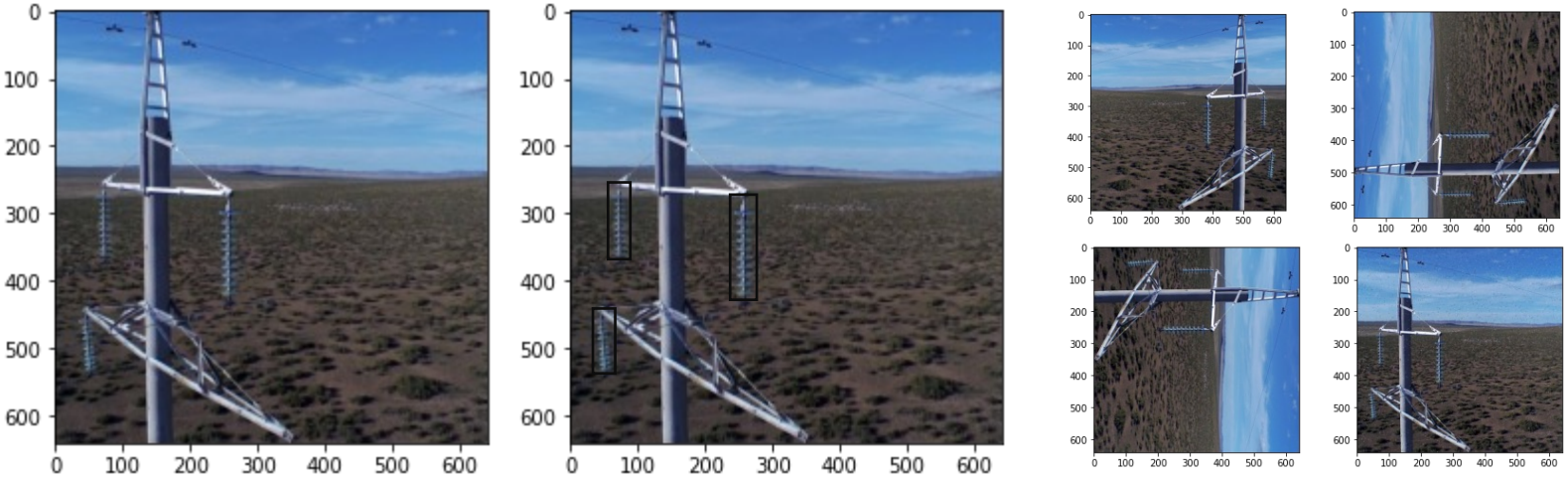
Figure 2. Workflow of the DL-based insulator detector implementation

A. Data Preparation

Collecttion

Lebelling

Augmentation



(a) (b) (c)

Figure 3. Data preparation steps-
(a) Original image.
(b) Labelled image.
(c) Augmented images: horizontal flipped, 90° anti-clockwise and 90°clockwise rotation, and 2% salt & pepper noise.

B. Model Training
DNN Training

- **Configuration**[a]
  - Batch = 64
  - Subdivision = 16
  - Max batches = classes×2000, but not less than the number of training images and not less than 6000.
  - Steps = 80% and 90% of the max batches i.e., steps=4800; 5400.
  - Filters = (classes + 5) × 3 i.e., 18 for our one class.
  - Width = 416 & Height = 416.
  - Learning rate = 0.00261

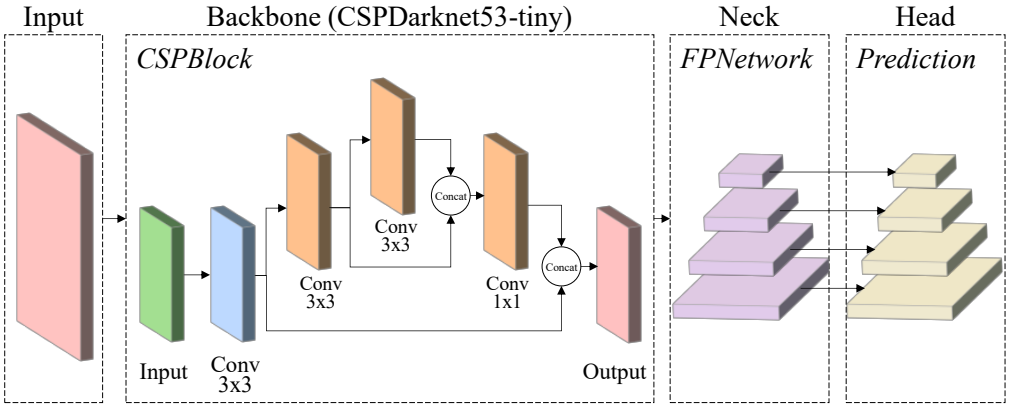


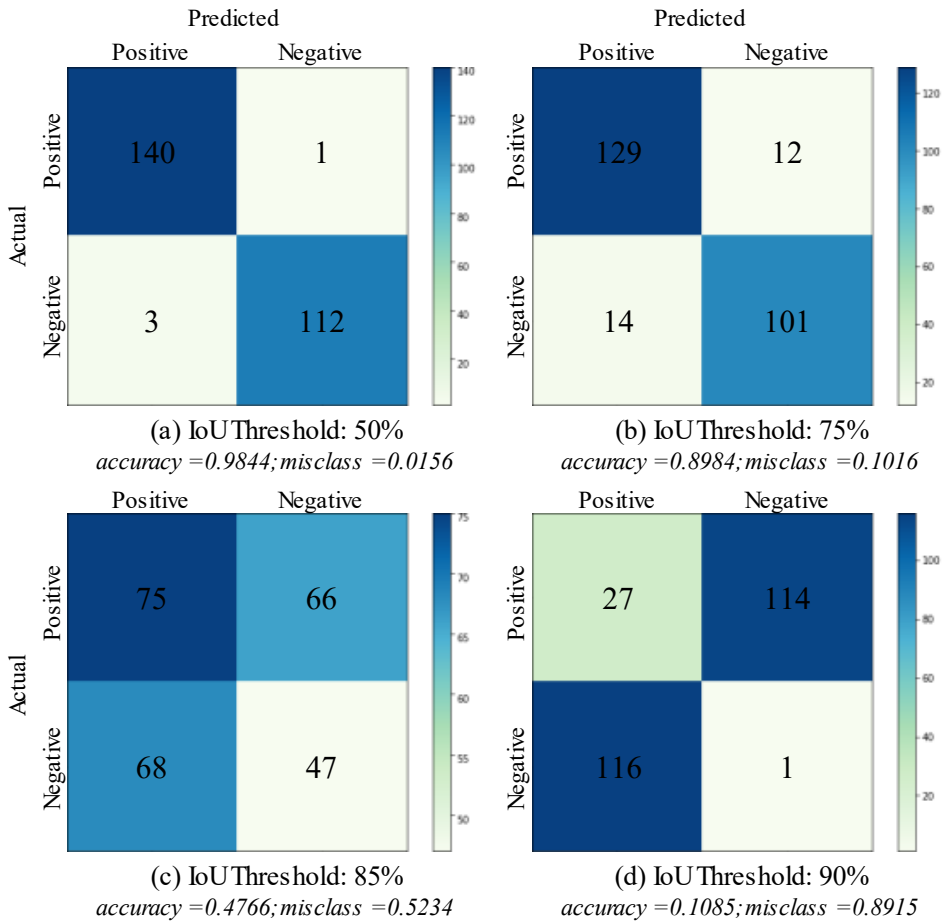
Figure 4. Architecture of proposed YOLOv4-tiny model with CSPDarknet53-tiny

[a] https://github.com/AlexeyAB/darknet

B. Model Training
Model Evaluation (1)

Confusion matrix based evaluation metrics:

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ (1)
- $Precision = \frac{TP}{TP+FP}$ (2)
- $Recall = \frac{TP}{TP+FN}$ (3)
- $F1 = 2 * \frac{Precision*Recall}{Precision+Recall}$ (4)
- $IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$ (5)



(a) IoUThreshold: 50%
accuracy =0.9844; misclass =0.0156

(b) IoUThreshold: 75%
accuracy =0.8984; misclass =0.1016

(c) IoUThreshold: 85%
accuracy =0.4766; misclass =0.5234

(d) IoUThreshold: 90%
accuracy =0.1085; misclass =0.8915

Figure 5. Confusion matrix results for Dataset 1
(a) IoU threshold at 50%
(b) IoU threshold at 75%
(c) IoU threshold at 85%
(d) IoU threshold at 90%

B. Model Training
Model Evaluation (2)

- Model is evaluated under different IoU threshold.
- Model performs excellently on 0.5 IoU threshold with mAP of 78.6% – 98.9%.
- Inference time-mAP trade-off for larger threshold.

Table 2. Model Evaluation Result

	Dataset 1				Dataset 2				Dataset 3			
IoU thr.	**0.50**	0.75	0.85	0.90	**0.50**	0.75	0.85	0.90	**0.50**	0.75	0.85	0.90
Accuracy	0.98	0.89	0.47	0.11	0.90	0.36	0.16	0.14	0.84	0.51	0.28	0.17
Precision	0.98	0.90	0.52	0.19	0.85	0.43	0.17	0.08	0.82	0.44	0.18	0.06
Recall	0.99	0.91	0.53	0.19	0.98	0.49	0.20	0.09	0.82	0.44	0.18	0.06
F1-score	0.98	0.66	0.53	0.19	0.91	0.46	0.19	0.09	0.82	0.44	0.18	0.06
IoU	0.83	0.77	0.46	0.17	0.64	0.36	0.15	0.07	0.61	0.37	0.16	0.05
mAP (%)	**98.9**	89.7	32.4	5.01	**97.0**	24.9	3.96	0.85	**78.6**	27.0	4.80	0.62

C. Model Deployment:
Image Acquisition

- Achieved via FLIR industrial cameras and Spinnaker SDK [3].
- Two different implementations: integrated acquisition & modular acquisition.
- Key difference is the data flow of image data.

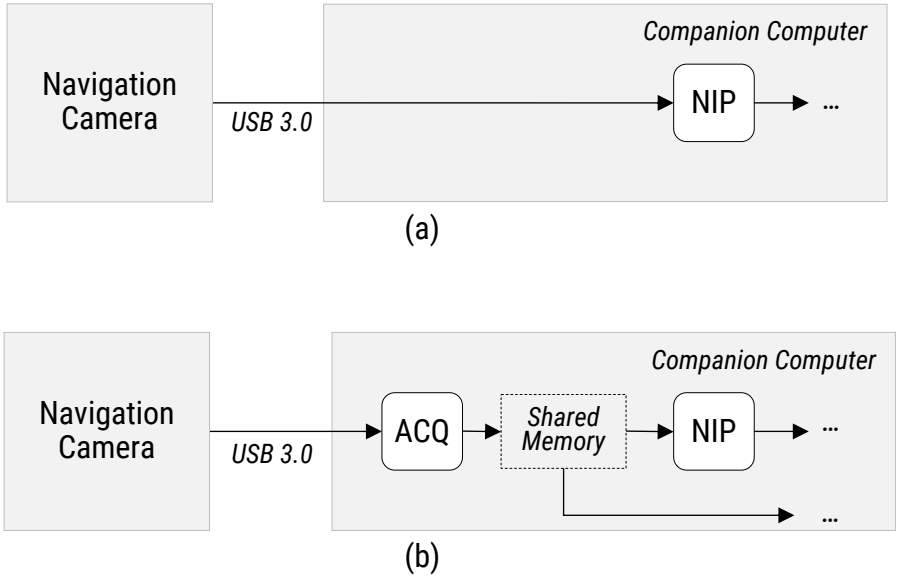


Figure 6. Image acquisition solutions
(a) Integrated acquisition
(b) Moduler acquisition

C. Model Deployment:
Inference

- OpneCV DNN
  - Detection Time (avg.): 130.0 ms/frame.
  - FPS: 7 (approx)
  - Optimized for Intel processor.
- Darknet Module
  - Detection Time (avg.): 73.48 ms/frame.
  - FPS: 15 (approx)
  - Native implementation.

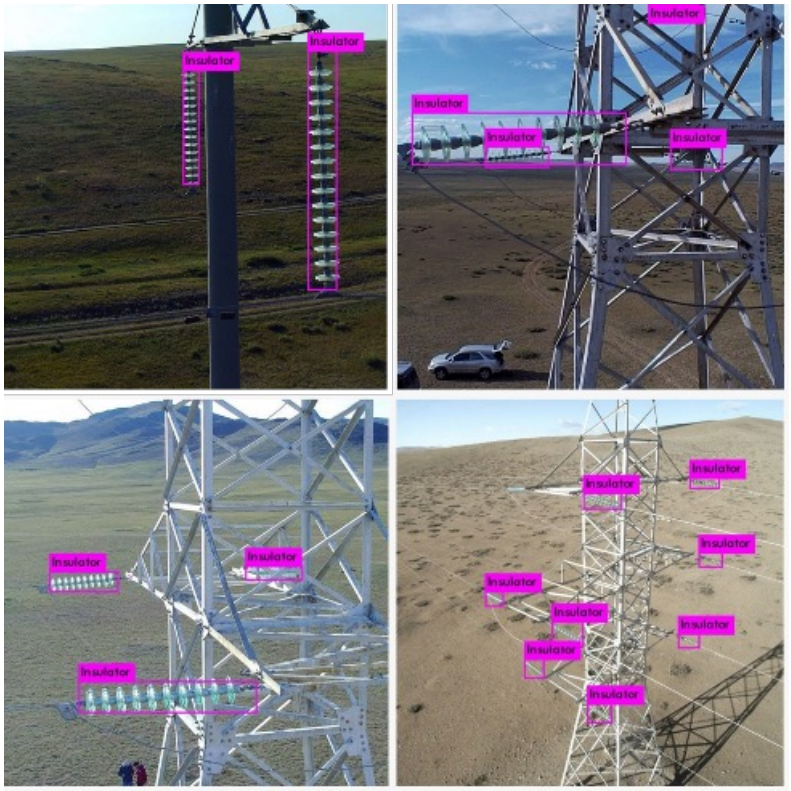


Figure 7. Detection result of high voltage power line insulators.

Results

Image Acquisition with FPS

- Modular acquisition is 36x faster on Jenson Nano and 23x on Jetson Xavier NX.
- No significant advancements on different power modes.

Table 3. Embedded Platform Test Result.

Device	**Jetson Nano (128 CUDA cores)**			**Jetson Xavier NX (384 CUDA cores)**			
Integrated Acquisition							
Power mode	5W	10W	Avg.	10W	15W	20W	Avg.
Acquisition [ms]	2.246	1.839	2.042	2.057	1.723	1.655	1.811
Detection [ms]	111.1	75.67	93.39	24.09	19.739	18.42	20.75
FPS	9.0	13.21	11.10	41.50	50.66	54.28	48.81
Modular Acquisition							
Power mode	5W	10W	Avg.	10W	15W	20W	Avg.
Acquisition [ms]	0.062	0.050	0.056	0.083	0.079	0.070	0.077
Detection [ms]	112.8	73.48	93.17	24.26	19.45	18.82	20.84
FPS	**8.859**	**13.60**	**11.23**	**41.20**	**51.39**	**53.13**	**48.57**

Results

Image Processing

- Darknet inference module performs better compared to OpenCV DNN modeule on ARM architecture.

Table 4. Comparison Of The Insulator Detection Algorithm On Embedded Platform.

Detection Method	**Traditional Image Processing**		**Deep-Learning based Model YOLOv4 tiny**		
Device	Odroid XU4	Odroid H2	Jetson Nano	Jetson Xavier	Google Colab
Processing unit	CPU	CPU	GPU	GPU	GPU
CUDA cores	-	-	128	384	4992
Image source	Camera	Camera	Camera	Camera	File
Image resolution	1024x768	1024x768	1024x768	1024x768	416x416
Acquisition [ms]	7.140	1.140	**0.050**	**0.070**	0.106
Detection [ms]	175.330	65.030	**73.489**	**18.820**	14.743
FPS	5.480	15.110	**13.607**	**53.134**	67.344

Conclusion

- DL-based insulator detector solution is developed, and the performance measured on different edge computing devices.
- Reach a maximum frame rate of:
  - **13.607 fps** on a **Jetson Nano**
  - **53.134 fps** on the **Jetson Xavier NX**
- Shared memory-based data sharing method is outperformed the conventional method in the image acquisition stage.
- The ongoing research will compare different DNN architectures to find the optimal model respecting the speed/accuracy tradeoff.

References

[1] “Jetson Nano: Deep Learning Inference Benchmarks,” *NVIDIA Developer*, Apr. 25, 2019. https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks (accessed Oct. 29, 2021).

[2] J. K. Jung, *YOLO performance on MS COCO dataset*. 2021. Accessed: Oct. 29, 2021. [Online]. Available: https://github.com/jkjung-avt/tensorrt_demos

[3] “Spinnaker SDK.” http://softwareservices.flir.com/Spinnaker/latest/index.html (accessed Oct. 29, 2021).

[4] Z. Jiang, L. Zhao, S. Li, and Y. Jia, “Real-time object detection method based on improved YOLOv4-tiny,” *arXiv:2011.04244 [cs]*, Dec. 2020, Accessed: Sep. 14, 2021. [Online]. Available: http://arxiv.org/abs/2011.04244

[5] “Darknet: Open Source Neural Networks in C.” https://pjreddie.com/darknet/ (accessed Oct. 29, 2021).