YOLO Performance Comparison

Cuauhtemoc Lona, Peter Tran, Rafael Baltazar, Kyle Acosta

ECE 4300.01 Computer Architecture

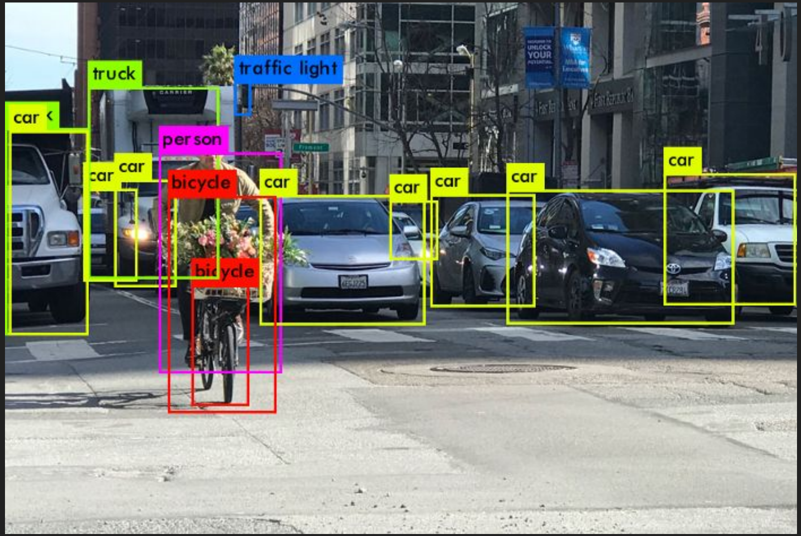
Motivation

- Tested and benchmarked the performance of objection detection of the YOLO11m model between CPUs and GPUs
  - Seven computers/laptops were tested; seven CPUs and seven GPUs
- Wanted to compare different computer architecture incorporating the algorithm
  - Performance, utilization, implementation, etc.

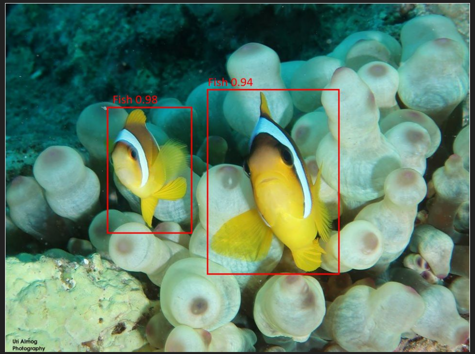
Introduction - What is YOLO

- YOLO (You Only Look Once) is an algorithm for object detection

- Uses an end-to-end network that makes predictions of bounding boxes and class probabilities all at once
  - Differs from the repurposed classifiers approach



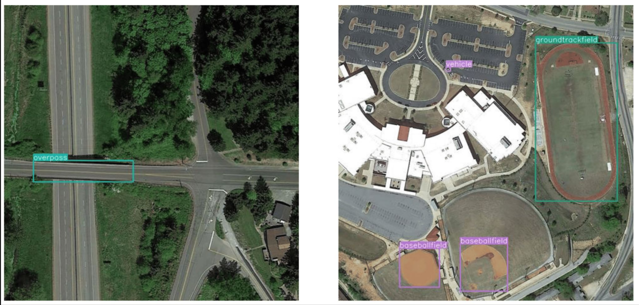
State of the art - Utilizations in the real world



- Traffic Sign Detection and Recognition, Basel
  - A way to combat and lower the amount of car crashes

- Underwater Environment Object Detection, Journal of Ocean University of China
  - An improvement to marine resources and to underwater exploration

- Small Object Detection for Aerial Images,Basel
  - Improving the inaccurate detection for large aerial images



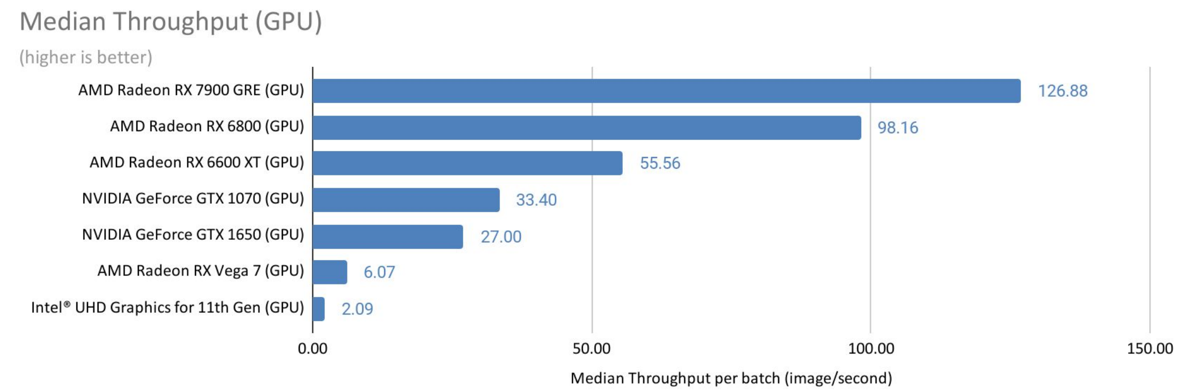
Testing Methodology

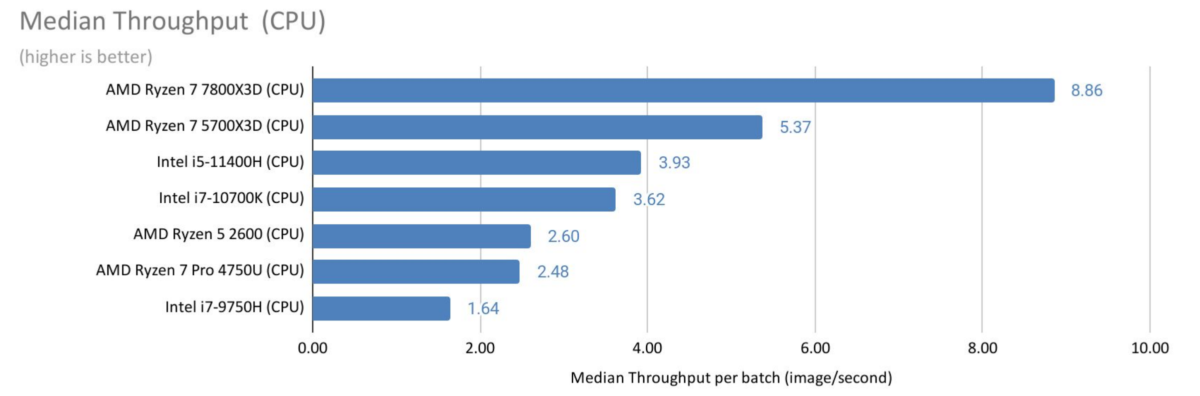
- **Python 3.10.5**, **ultralytics 8.3.39**, **onnxruntime-directml 1.20.1**, **opencv 4.10.0.84**
- **HWiNFO64 8.16** and **psutil 6.1.0** used for system hardware monitoring
  - psutil for CPU and RAM usage, HWiNFO for CPU/GPU power consumption and VRAM usage
  - HWiNFO measurements had to be manually started before and after the actual benchmark test
  - Benchmark script would log batch number, inference time, throughput, process CPU/RAM usage
- Ultralytics YOLO11m model exported to ONNX with **imgsz of 640**, **dynamic batch size up to 64**
- **GPUs accelerated with DirectML**
- Performed **inference** on first **128 images of COCO** (Common Objects in Context) **Val 2017** dataset; **batch size of 16** to fit all systems
  - GPU benchmarked first, then CPU right after
- Removed any unnecessary background processes that may take up system resources during benchmark

System Setups

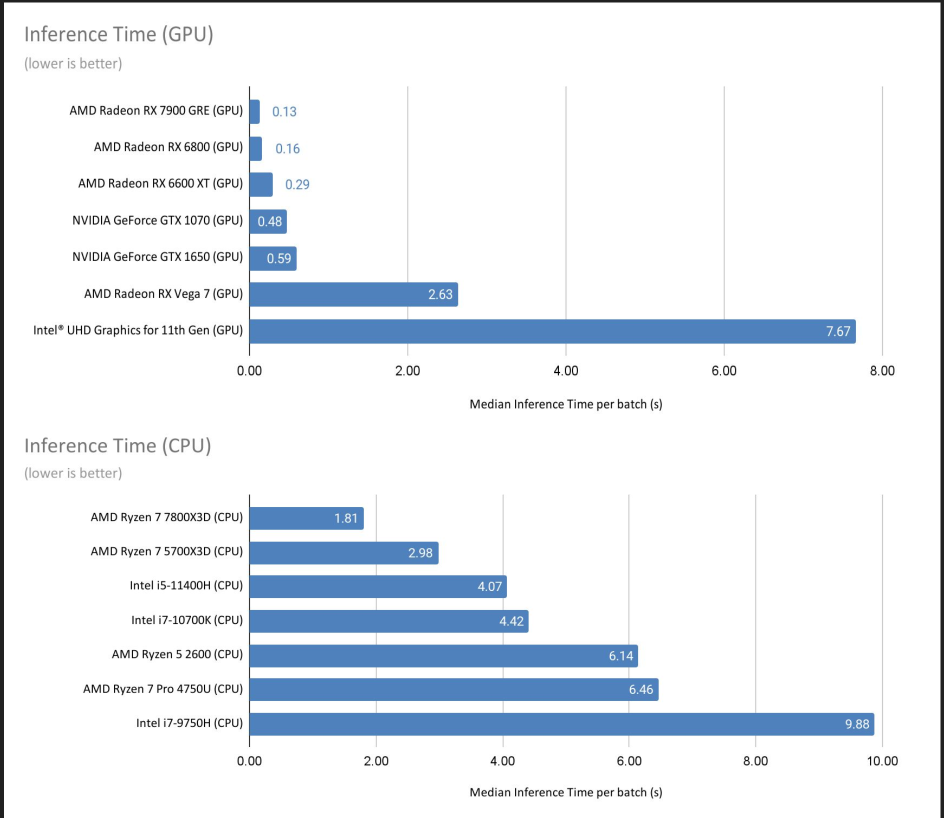
raflaptop	Intel i5 11400H 32GB DDR4 3200MT/s	**Tiger Lake-H** | **6c/12t**, 2.7-4.5GHz, 12MB L3	Intel UHD Graphics for 11th Gen	**Tiger Lake Xe** | **16 EUs**, 350-1450 MHz	Windows 10 22H2
peterlaptop	AMD Ryzen 7 Pro 4750U 16GB DDR4 3200MT/s	**Zen 2** | **8c/16t**, 1.7-4.1GHz, 8MB L3	AMD Radeon RX Vega 7	**Vega** | **7 CUs**, 448-1800MHz	Windows 10 22H2
kylelaptop	Intel i7 9750H 16GB DDR4 2667MT/s	**Coffee Lake-H**, **6c/12t**, 2.6-4.5GHz, 12MB L3 |	NVIDIA GeForce GTX 1650	**Turing** | **896 SUs, 14 SMs**, 1485-1665 GHz **4GB GDDR5 128-bit**	Windows 11 24H2
rafpc	AMD Ryzen 7 5700X3D 32GB DDR4 3200MT/s	**Zen 3** - **8c/16t**, 3.0-4.1GHz, 96MB L3	AMD Radeon RX 6800	**RDNA2** | **3840 SUs, 60 CUs, 60 RTs**, 1700-2105MHz **16GB GDDR6 256-bit**	Windows 10 22H2
rafpc2	AMD Ryzen 5 2600 16GB DDR4 2667MT/s	**Zen+** - **6c/12t**, 3.4-3.9GHz, 16MB L3	NVIDIA GeForce GTX 1070	**Pascal** | **1920 SUs, 15 SMs**, 1506-1683 MHz **8GB GDDR5 256-bit**	Windows 10 22H2
peterpc	Intel i7 10700K 16GB DDR4 3200MT/s	**Comet Lake** - **8c/16t**, 3.8-5.1GHz, 16MB L3 |	AMD Radeon RX 6600 XT	**RDNA2** | **2048 SUs, 32CUs, 32RTs**, 1968-2589MHz **8GB GDDR6 128-bit**	Windows 10 22H2
kylepc	AMD Ryzen 7 7800X3D 32GB DDR5 6000MT/s	**Zen 4** - **8c/16t**, 4.2-5GHz, 96MB L3	AMD Radeon RX 7900 GRE	**RDNA3** | **5120 SUs, 80 CUs, 80RTs**, 1288-2245MHz **16GB GDDR6 256-bit**	Windows 11 24H2

Performance Evaluations Pt. 1

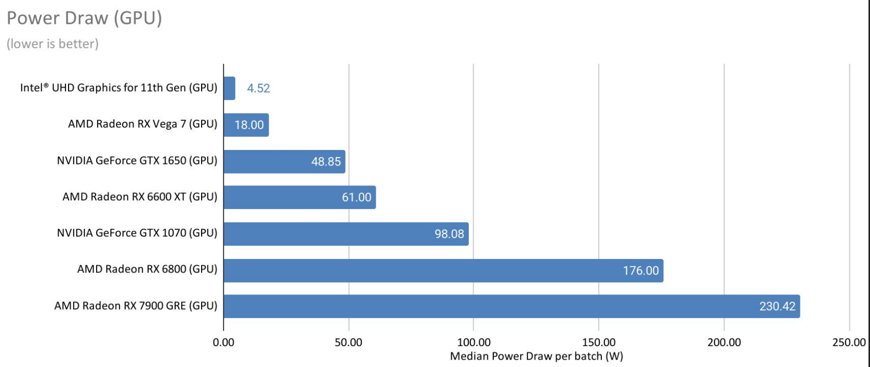


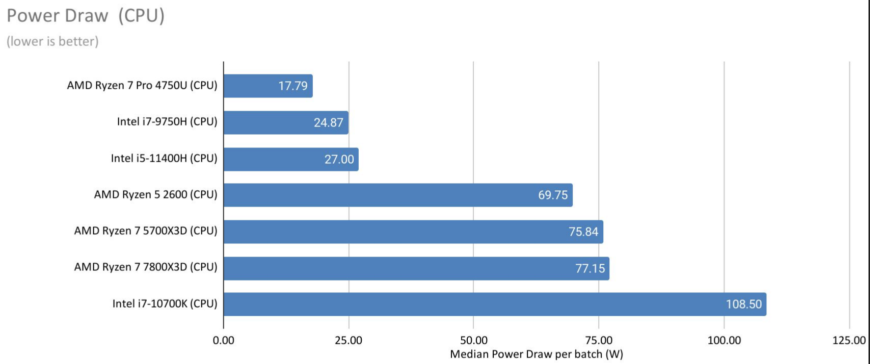


Performance Evaluations Pt. 2

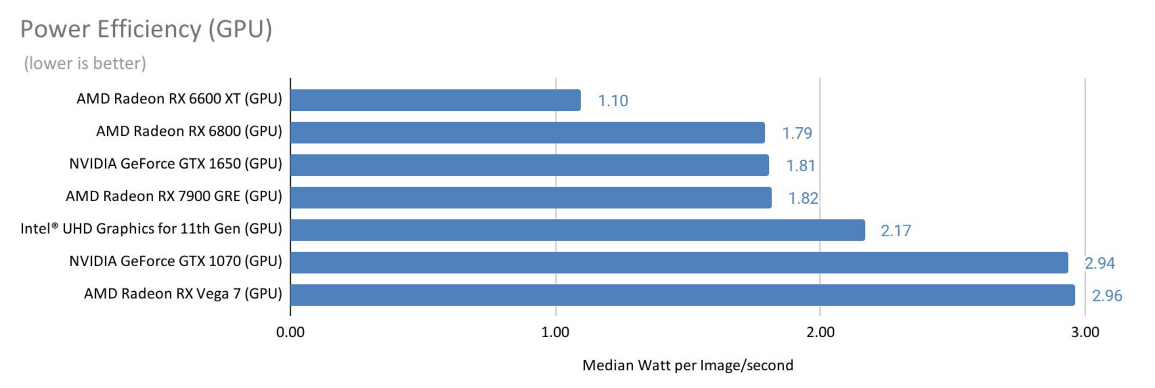


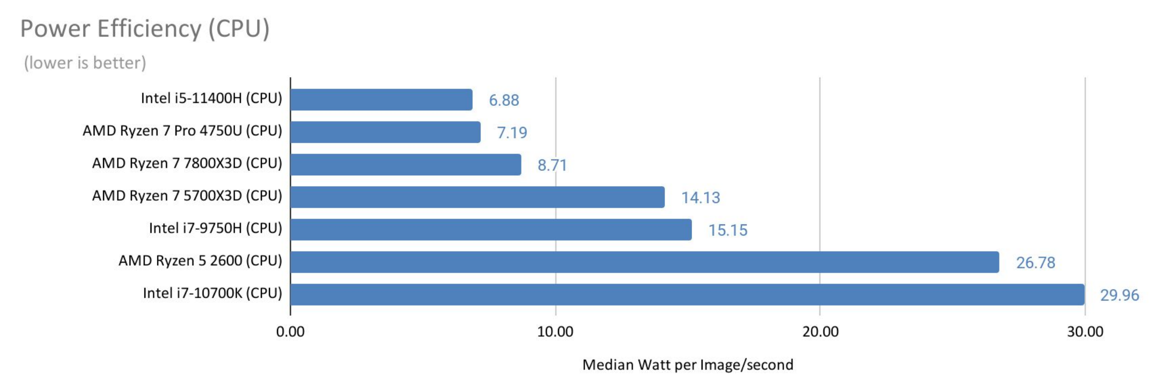
Performance Evaluations Pt. 3



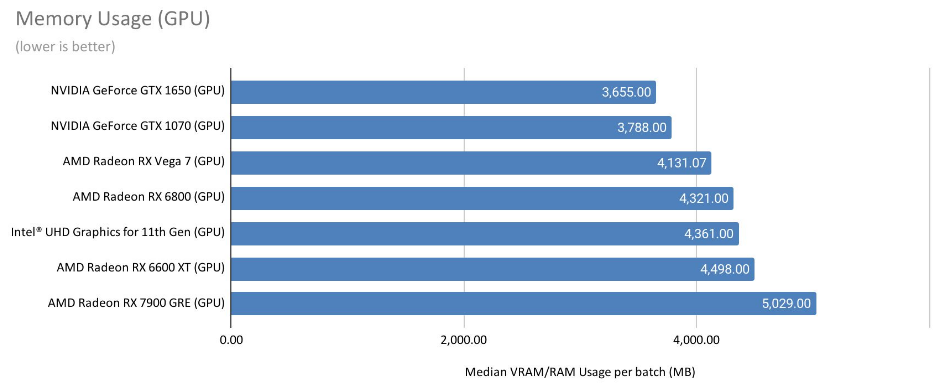


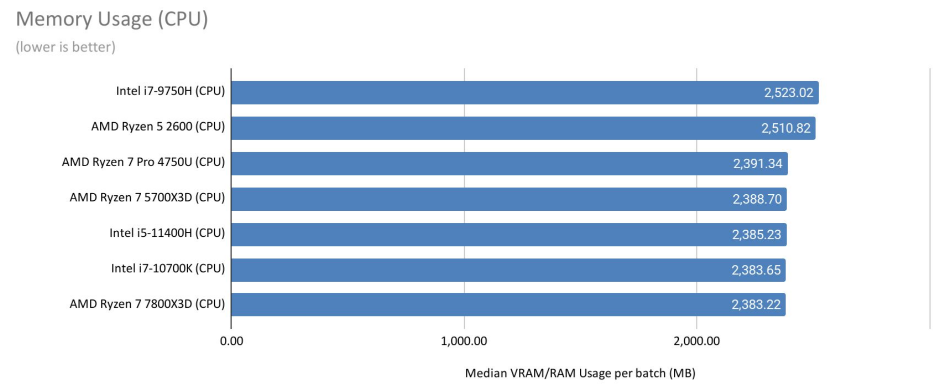
Performance Evaluations Pt. 4



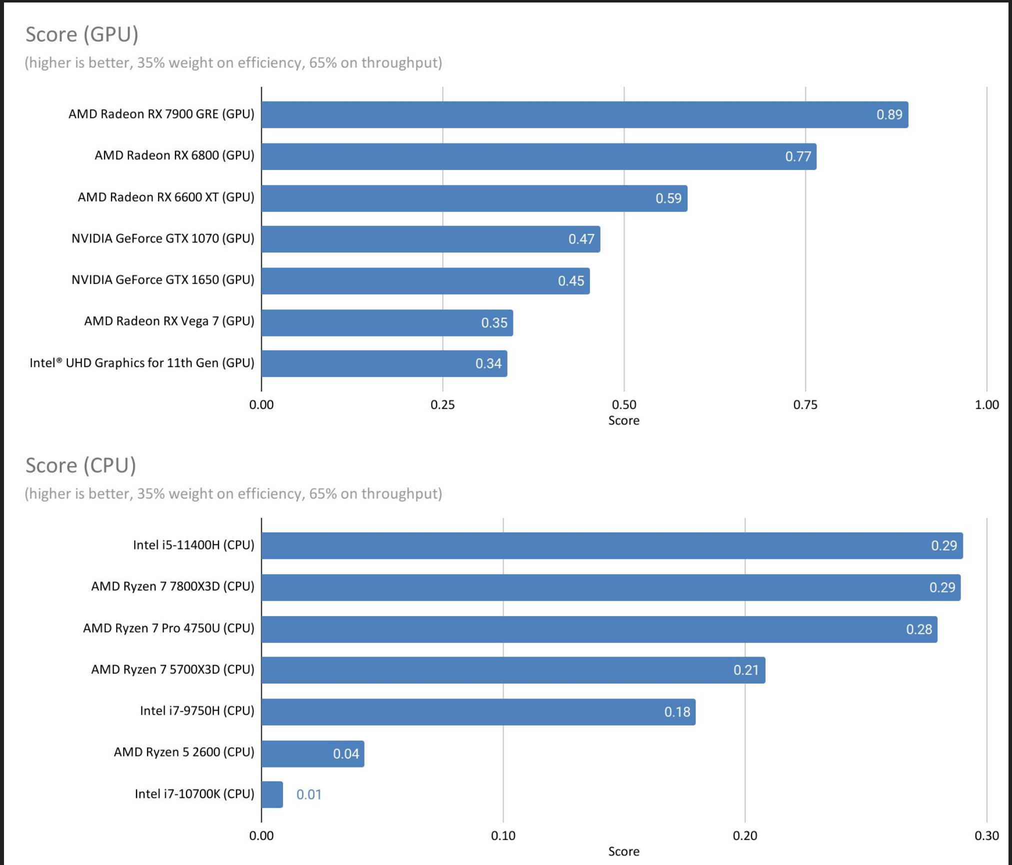


Performance Evaluations Pt. 5





Performance Evaluations Pt. 6



Analysis and Conclusions

- Based on our scoring system of 35% weight on efficiency, 65% on throughput, all our GPUs outperformed all of our CPUs
  - Lowest GPU score: Intel® UHD Graphics for 11th Gen at **.34**
  - Highest GPU score: AMD Radeon RX 7900 GRE at **.89**
  - Lowest CPU score: Intel i7-10700K at **.01**
  - Highest CPU score: Intel i5-11400H at **.29**
- GPUs are the most efficient
  - Range of **1.10 - 2.96 W** per img · sec for GPUs, versus range of **6.88 - 29.96 W** per img · sec for CPUs; range of **2.32x to 27.23x** better efficiency
- Integrated GPUs tend to struggle versus CPUs, but discrete GPUs have massive advantage
  - RX Vega 7 and Intel UHD Graphics measured **6.07** and **2.09** img/sec, while CPUs ranged from **1.64 - 8.86** img/sec
  - The slowest discrete GPU, the GTX 1650 is already faster than all CPUs and integrated GPUs
- Object detection is heavily dependent on parallel processing

Future Work

- More consistent/accurate data gathering by lining up timestamps across measurements and having a distinct time between each reading
- Have each setup use the same operating system with the exact same version
- Absolutely ensure all setups are absolutely focusing on the benchmark only with no resources put towards anything else
- Perform GPU benchmarking with NVIDIA CUDA / AMD ROCm to compare their performance versus DirectML and other GPUs

References

- Flores-Calero, Marco, et al. “Traffic sign detection and recognition using Yolo Object Detection Algorithm: A systematic review.” *Mathematics*, vol. 12, no. 2, 17 Jan. 2024, p. 297, https://doi.org/10.3390/math12020297.
- Yang, Yuyi, et al. “UGC-Yolo: Underwater Environment Object Detection based on Yolo with a global context block.” *Journal of Ocean University of China*, vol. 22, no. 3, 13 May 2023, pp. 665–674, https://doi.org/10.1007/s11802-023-5296-z.
- Hu, Mengzi, et al. “Efficient-lightweight YOLO: Improving small object detection in Yolo for aerial images.” *Sensors*, vol. 23, no. 14, 15 July 2023, p. 6423, https://doi.org/10.3390/s23146423.