Assessment #6

Research Assessment

Name: Aryan Patel

Date: October 25, 2026

Subject: Machine Learning

MLA Citation:

Aljami, Hadeel Muhammad, et al. "Benchmarking YOLOv8 Variants for Object Detection Efficiency on Jetson Orin NX for Edge Computing Applications." *Computers*, vol. 15, no. 2, 1 Feb. 2026, article 74, https://doi.org/10.3390/computers15020074.

Analysis:

The authors tested the five different sizes of the YOLOv8 models (nano, small, medium, large and extra large) on the widely used embedded computer NVIDIA Jetson Orin NX, which is commonly used in robotics. They evaluated all models on PyTorch and NVIDIA's own TensorRT, which is designed to speed up running the models on NVIDIA's hardware. TensorRT managed to cut the time by 52.4% to 63.3% in the vehicle video test, but it crashed in the package test at a batch size of 8 while PyTorch continued to run. They measured the performance of their package system with a batch size of 1 and found it to be 23.67 frames per second, but increased to 32.04 at a batch size of 4, which they say is optimal for real-time use.

What I learned was that it wasn't necessarily the model itself that was the best option, it was the way the model was being utilized. The same weights needed less than half the time to execute as a TensorRT engine, while the TensorRT engine was only able to process a batch of 4. PyTorch was slower and kept running till all memory was used up for the two largest models, at batch size 128. I was hoping to be able to get this for free as an add-on to the export to a faster

format, but this paper proved otherwise. All the speed was achieved from an export and a setting and none from retraining, which was good. The other thing that surprised me was that they had the limit reported at all, and they say that the previous benchmarks didn't report the limit failure.

This is related to Assessment 3. There I wrote that mAP never measures parts of a deployed system, for example, mapping boxes to slots, and tracking cars between frames. Another thing that mAP doesn't measure is time, this paper notes. A detector that would return a score of 0.99 but which took a long time to return would be of little use for a robot that moves around, as the object would have moved by the time the answer was received. The speed of my detector if it ends up on an embedded board such as the Orin NX is part of the result. So, regarding my own vision work, I have two questions: 1) Is my test set a real test and 2) Can the model run on the hardware it will be running on?

I'd split the paper into 3 different settings, the format of the files (PyTorch vs TensorRT), batch size and image size (640 vs 1200 pixels). This has led me to doubt a belief of mine. I was under the impression that the larger batch equated to faster since there's more work being done all at once at the GPU. It's per image, not per answer. This is also in line with a trend I've had in my other evaluations. With Assessment 3 the split of the evaluations determines which architecture wins, and with this the format and batch size determine. In Assessment 1 80 percent of workers reported an increase in productivity but only 37 percent of companies reported any impact on earnings. Good number was always from one setting and never moved to the setting that counts.

I don't like the idea of 4 being the optimum batch size for real-time use. According to Table 4, it is observed that the amount of time taken for the detection per frame has reduced from 27.07 ms to 16.27 ms and the amount of time taken per batch has increased from 40.59 ms to

116.88 ms. If the robot is reading one live camera, then a batch of 4 will wait for 4 frames, then wait for all 4 frames to complete. The first frame in the batch is already about 100ms old when the GPU begins to run on it, if the camera is running at 30 fps. This is my estimation, not the paper's. I believe they're measuring throughput when a robot requires latency, which are different. Batch 4 is fine to use for a video file, but Batch 1 may be the one to use for a robot and the paper's fps for it is 23.67.

I'm also uncertain as to the extent of the trustworthiness of the numbers. The data were split as 80/10/10, but the package data set was comprised of 12042 training images, 1187 for validation and only 16 for testing, closer to 91:9:0.1 percent split. It is not specified in Table 3 which split it is based on and the "accuracy" of the vehicle is not defined and can range from 0.61 to 0.75. There may be some effect on accuracy with FP16 conversion but there is no mAP reported in the TensorRT models, so there is no accuracy result to go with the speed result. The abstract states that the time per image is reduced by 42%, but the body states 39.9%. Also the 451.24 MB figure for PyTorch is not found elsewhere in the paper. Again there's no big change, but I found a mismatch just as I did in Assessment 3, and it was a reminder to check things for myself. I do not understand why it only consumed 42.96 MB of the GPU memory at 4 batch size but at 8 batch size it failed to work with the TensorRT on a 16 GB board. I suspect it has to do with how the engine is constructed, but the paper does not test that.

The authors also used an active cooling kit, and left the power consumption and heat to future research. My two questions for my next paper on a battery powered robot would be what are some of the things to consider for power, and what are some considerations for the thermal behavior of embedded boards. First, I'll record the time that it takes my detector to process a batch size of 1, after taking the picture and before I see the result. I'll also record the 95th

percentile, since the slow frames will have more of an effect on the robot than the average frames. Secondly I will export my model to another faster format and compare the speed and the mAP of the two models for my session based split that I had designed for Assessment 3 as this paper does not provide a mAP number. Third, I'll be adding the following to the model card that I'll be writing: hardware, file format, batch size and latency.

Annotated article: Annotated Article - Assessment #6.pdf