Create pipeline using DL Streamer, define system scalability for Intel HW

by -Bhavya Jain, Rishabh Dhiman

1. Abstract

This benchmark evaluates Intel's Deep Learning Streamer (DL Streamer) pipeline performance on an Asus Zenbook S14 laptop featuring an Intel Core Ultra 2 256V CPU (Lunar Lake) with integrated Intel Arc 140V GPU and 47 TOPS Neural Processing Unit (NPU). We assessed a two-stage AI pipeline (person detection followed by age classification) under CPU-only, GPU-accelerated, and NPU-accelerated execution modes, targeting a frame rate of 15 FPS per video stream. Results showed that the GPU provided the highest throughput (up to ~223 FPS total at 15 streams) but was ultimately limited by CPU usage. The NPU maintained significantly lower CPU utilization but faced memory bandwidth limitations beyond ~7 streams, while the CPU-only pipeline saturated at ~7 streams due to compute constraints.

2. Introduction

Real-time video analytics has become crucial for smart city infrastructure, surveillance systems, and intelligent transportation applications. These applications require efficient, scalable video analytics solutions. Intel's DL Streamer, built upon the GStreamer framework, facilitates rapid development and optimization of AI video analytics pipelines utilizing heterogeneous computing platforms, including CPU, GPU, and NPU accelerators. This study aims to quantify the performance scalability, throughput, and resource utilization of a representative AI pipeline on Intel’s latest Lunar Lake architecture, comparing CPU, GPU, and NPU performance.

3. Experimental Setup

- **Hardware Platform**: Asus Zenbook S14 with Intel Core Ultra 2 256V processor (15th Gen Lunar Lake), consisting of 2 Performance cores, 6 Efficiency cores, Intel Arc 140V GPU (8 Xe-cores at 1.95 GHz), 47 TOPS Intel AI Boost NPU, and 32 GB LPDDR5X RAM running Ubuntu 24.04.2 LTS.
- **Software Framework**: Intel DL Streamer (v2024.1), OpenVINO Toolkit, and GStreamer framework. RTSP video streams simulated using local server to mimic real-world CCTV feeds.
- **Pipeline Architecture**: SSD-Mobilenet for person detection followed by an age classification model from Intel’s Open Model Zoo. GStreamer pipeline leveraged gvadetect and gvaclassify plugins. Resource utilization metrics included total and per-stream FPS, CPU load, memory consumption, and network throughput measured during 30-second trials.

4. Model Throughput Benchmarking

Each model was benchmarked using the following procedure:

1. **Model Loading:** IR files (.xml and .bin) loaded via OpenVINO Runtime Core API.
2. **Warm-Up:** 20 dummy inference iterations performed to prime the pipeline and caches.
3. **Measurement:** 20 timed inference iterations executed sequentially.
4. **FPS Calculation:** FPS computed as 20 divided by total elapsed seconds.
5. **Streams Calculation:** Maximum concurrent streams determined by floor division of measured FPS by the target 15 FPS.

Results:

Model	Measured FPS	Max Streams (15 FPS)
age-gender-recognition-retail-0013	3361.5	224
head-pose-estimation-adas-0001	3317.9	221
facial-landmarks-35-adas-0002	1687.6	112
vehicle-detection-adas-0002	127.7	8
pedestrian-detection-adas-0002	124.2	8
human-pose-estimation-0001	19.5	1
semantic-segmentation-adas-0001	5.0	0

Analysis:

- **High-throughput Models:** Age-gender and head-pose estimation models exhibited exceptional throughput exceeding 3300 FPS, making them ideal for extremely high-density deployments with over 200 simultaneous streams.
- **Moderate-throughput Models:** Vehicle and pedestrian detection models provided moderate throughput (~125 FPS), suitable for approximately 8 simultaneous streams at 15 FPS.
- **Low-throughput Models:** Human-pose estimation and semantic segmentation models performed under 20 FPS, indicating they are too resource-intensive for multiple real-time streams on CPU alone.

5. Hardware Benchmarking (CPU/GPU/NPU)

Models tested: Age & Gender Recognition, Person Detection

CPU-Only Mode

Throughput Scaling

- Scaled linearly from **1→5 streams** (30 FPS→132 FPS total).
- Beyond **5 streams**, combined FPS plateaued (~125 FPS at 7–9 streams), indicating the CPU's compute limit.

Per-Stream Performance

- Maintained > 26 FPS/stream up to 5 stream.
- Drops to ~18 FPS (7 streams) and ~14 FPS (9 streams), failing the 15 FPS real-time target.

Resource Utilization

- **CPU**: Jumped to ~92 % at 5 streams and stayed ≥ 90 % thereafter—clear compute saturation.
- **Memory**: Grew modestly (38 %→48 %), never exceeding capacity.
- **Network**: ~0.6 Mbps per stream; negligible impact.

Bottleneck Diagnosis

CPU is the limiting factor once multiple inferences run in parallel. Memory and I/O remain underutilized.

Table 1: CPU Pipeline Performance

Streams	Combined FPS	Per-Stream FPS	CPU (%)	Memory (%)	Network (Mbps)	Bottleneck
1	30.23	30.23	17	38.1	0.6	Memory
3	89.67	29.89	47.3	39.6	1.7	CPU
5	131.97	26.39	92	41.9	2.9	CPU
7	125.30	17.90	91.1	44.5	4.1	CPU
9	125.60	13.96	90.6	47.9	5.3	CPU

GPU-Accelerated Mode

Throughput Scaling

- Near-ideal up to **5 streams**, sustaining ~30 FPS/stream (150 FPS total).
- Continued to climb sub-linearly to ~223 FPS at 15 streams.

Per-Stream Performance

22 FPS/stream at 7 streams; falls to ~14.8 FPS/stream at 15 streams.

Resource Utilization

- **CPU**: Low (~41 %) at 5 streams, then steep rise to ~85–92 % at ≥ 7 streams—pipeline coordination and decoding burden.
- **Memory**: Moderate growth (47 %→52 %).
- **GPU**: Underutilized headroom implied by rising CPU load without GPU saturation.

Bottleneck Diagnosis

Initially limited by memory/cache warm-up (1–5 streams), then by **CPU**, which handles frame scheduling, GStreamer threads, and metadata even when inference is offloaded.

Table 2: GPU Pipeline Performance

Streams	Combined FPS	Per-Stream FPS	CPU (%)	Memory (%)	Network (Mbps)	Bottleneck
1	30.00	30.00	7.9	46.7	0.6	Memory
3	89.63	29.88	27.3	46.9	1.7	Memory
5	149.30	29.86	41.1	47.2	2.8	Memory
7	160.30	22.90	85	47.8	4	CPU
9	178.10	19.79	87.3	48.5	5.2	CPU
11	201.90	18.35	85.2	49.4	6.4	CPU
13	219.57	16.89	84.7	50.3	7.6	CPU
15	222.60	14.84	91.6	51.8	8.8	CPU

NPU-Accelerated Mode

Throughput Scaling

- Super-linear scaling: 1→5 streams yields 30 FPS→163 FPS total (~33 FPS/stream), showing the NPU waiting on incoming frames.
- Peaks at ~202 FPS (7 streams), then drops sharply to ~154 FPS at 9 streams.

Per-Stream Performance

Sustains > 28 FPS/stream up to 7 streams; falls to ~14.5 FPS/stream at 11 streams.

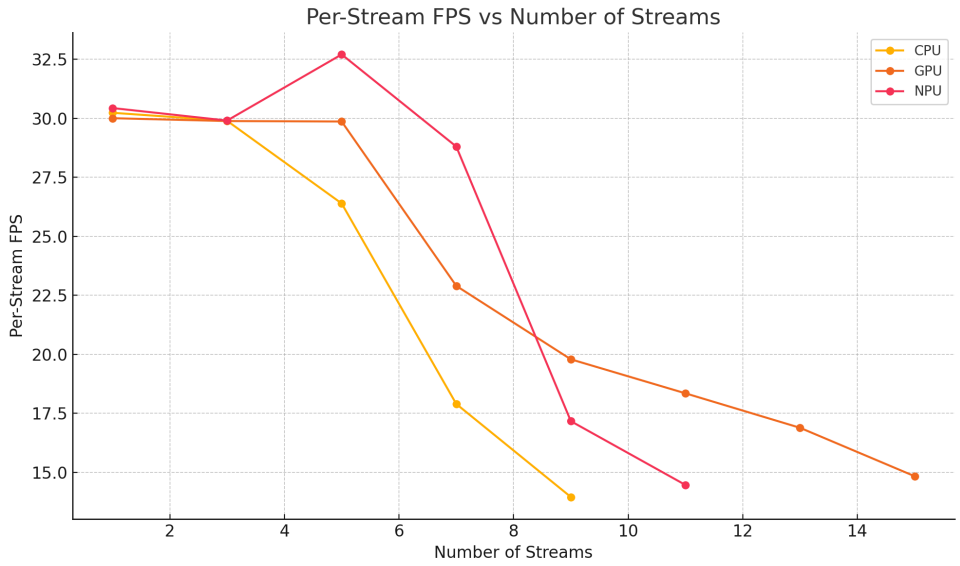
Resource Utilization

- **CPU**: Very low overhead (5–14 %), confirming effective offload.
- **Memory**: Highest usage across modes (52 %→58 %), climbing steeply with streams.

Bottleneck Diagnosis:
Memory subsystem (bandwidth or model-instance footprint) constrains the NPU's ability to handle more concurrent streams. Likely due to multiple model contexts in system RAM or limited on-chip memory for parallel inferencing.

Table 3: NPU Pipeline Performance

Streams	Combined FPS	Per-Stream FPS	CPU (%)	Memory (%)	Network (Mbps)	Bottleneck
1	30.43	30.43	5.4	52.1	0.6	Memory
3	89.70	29.90	8.3	52.7	1.7	Memory
5	163.50	32.70	10.2	53.6	3.1	Memory
7	201.63	28.80	13.7	54.7	4	Memory
9	154.50	17.17	12.9	56.2	5.2	Memory
11	159.20	14.47	14.1	57.6	6.4	Memory



6. Conclusion

- **CPU-only mode**: Suitable for small-scale or low-density stream analytics scenarios; performance quickly limited by CPU capacity.
- **GPU-accelerated mode**: Best choice for scenarios requiring high-density, high-throughput video analytics. CPU resources must be adequately provisioned.
- **NPU-accelerated mode**: Optimal for scenarios prioritizing energy efficiency and minimal CPU overhead, ideal for moderate stream counts.

System designers should utilize GPU acceleration for maximizing stream throughput, leverage NPU capabilities for power efficiency, and consider CPU-only implementations primarily for simpler or lower-load scenarios.

Future Work: Investigate multi-stream batching, hybrid GPU/NPU load balancing, asynchronous pipeline processing,& heavier model workloads to further define scalability limits.

7. References

1. Intel (2024). Intel® Core™ Ultra 7 Processor 258V Specifications. ARK Product Specs. https://www.intel.com/content/www/us/en/products/sku/240957/intel-core-ultra-7-processor-258v-12m-cache-up-to-4-80-ghz/specifications.html
2. **DL Streamer Developer Guide**. (n.d.). Intel DL Streamer Documentation. Retrieved July 4, 2025, from https://dlstreamer.github.io/dev_guide/dev_guide_index.html
3. **Ubuntu Documentation**. (n.d.). Ubuntu 24.04.2 LTS Official Documentation. Retrieved July 4, 2025, from https://help.ubuntu.com/
4. **OpenVINO™ Toolkit Documentation**. (2024). Intel OpenVINO Toolkit. Retrieved July 4, 2025, from https://docs.openvino.ai/
5. **oneAPI Developer Portal**. (2024). Intel oneAPI Toolkit Documentation. Retrieved July 4, 2025, from https://docs.oneapi.com/