nature portfolio

https://doi.org/10.1038/s41586-025-08854-x

Supplementary information

Universal photonic artificial intelligence acceleration

In the format provided by the authors and unedited

Supplementary Information

I. Adaptive block floating point (ABFP) format with overamplification

The ABFP numerical format is designed to work with the use of analog-to-digital converters (ADCs) and digital-to-analog converters (DACs) in the photonic tensor core (PTC) that requires all numbers to be represented as fixed-point numbers. ABFP allows a group (or block) of numbers to share a common exponent that adapts based on the data's characteristics. This shared exponent efficiently captures the dynamic range of the block, enabling precise computations without the overhead of individual exponents for each number.

ABFP is particularly amenable to the $N \times N$ PTC architecture of the photonic processor where: each (activation) vector element x_j is split into N row-lines and is then modulated by the weight value w_{ij} before addition is performed at each row-line to produce the result $y_i = \sum_{j=1}^{N} w_{ij} x_j$. In our implementation, we normalize every bfloat16 vector with a scale $s_x = \max(|\vec{x}|)$ (the scale is in bfloat16) and then quantize the resulting vector $\hat{x}$ (which now has a value between -1 and 1 only) to 10 bits by the DAC, which we denote with the function Q_{10}. Similarly, we normalize every row vector of the bfloat16 weight with a scale $s_{w;i} = \max(|\vec{w_i}|)$, where $\vec{w_i}$ denotes row i of the weight matrix w. The resulting vector $\hat{w}_i$ (which has a value between -1 and 1) is then quantized to 7 bits by the DAC, denoted with Q_7. Quantization for the DACs are performed symmetrically around 0 (for signed integers) to eliminate any bias:

$$Q_n(\alpha) = clamp_{[-1,+1]}(round(\frac{\alpha}{\delta_n}) \cdot \delta_n),$$

where $clamp_{[-1,+1]}(\beta) = \max(\min(\beta, 1), -1)$ and $\delta_n = 1/(2^{n-1} - 1)$. This quantization method forgoes one level but allows the numerical representation to be symmetric around zero.

The matrix-vector multiplication is performed by the PTC and the output is quantized to 11 bits by the ADC, which is denoted with Q_{11}. The resulting fixed point output is then multiplied with the scales to reconstruct the bfloat16 output as shown below:

$$y_i = s_{w;i} s_x \times Q_{11}\{\sum_{j=1}^{N} Q_7(\hat{w_{ij}}) Q_{10}(\hat{x_j})\}.$$

Therefore, the operation $Q_{11}\{\sum_{j=1}^{N} Q_7(\hat{w_{ij}}) Q_{10}(\hat{x_j})\}$ is done in the PTC, and the multiplication with bfloat16 scales $s_{w;i} s_x$ is performed in the NCE.

ABFP separates scale factors for each row vector wij, and each column vector xj, and this scheme greatly reduces the quantization effects. Employing ABFP does come at the expense of rescaling the dot product

inputs, along with additional BFLOAT16 storage and multiplications. Importantly, we note that for inference, the weight tensors need only to be converted to and stored in ABFP representation once before any inference.

Normally, as the size of N increases, so do the number of bits required to represent the output of the multiplication (by $\log_2(N)$). If the bit precision of the ADC is constant, then more information on the output is lost as fewer lower significant bits can be captured. However, by physically increasing the gain of the analog signal, these lower significant bits can be recovered—at the cost of possibly increased saturation of more significant bits. This effectively increases the precision of the multiplication output and allows the networks to maintain its accuracy even with larger values of N.

Mathematically, we introduce an overamplification—or gain—factor $G > 1$ by increasing both the TIA gain and the laser power, such that the photonic processor operates with:

$$y_i = \frac{s_{w;i} s_x}{G} Q_{11}\{G \cdot \sum_{j=1}^{N} Q_7(\hat{w_{ij}}) Q_{10}(\hat{x_j})\}.$$

This modified operation has $Q_{11}\{G \cdot \sum_{j=1}^{N} Q_7(\hat{w_{ij}}) Q_{10}(\hat{x_j})\}$ being performed in the PTC and the multiplication with the scale $s_{y_i} = (s_{w;i} s_x / G)$ to be done by the NCE.

ABFP is a particular instance of post-training quantization. Although ABFP produces good results, significant improvement is observed with the use of quantization-aware training (QAT). While QAT can be computationally intensive, especially for really large language models, we explore the performance benefits of applying QAT in conjunction with ABFP for models. During fine-tuning, the forward pass is executed using ABFP, with a straight-through estimator of the quantization functions Q_7, Q_{10}, Q_{11} in the backward pass:

$$\partial Q(\alpha)/\partial \alpha = I, \text{ for } \alpha \text{ within the accepted range of inputs (between [-1,+1] for DACs).}$$

During QAT, we also train the network to learn about the analog noise in the hardware. We represent this noise to be an additive Gaussian noise to the result of the analog matrix-vector multiplication. We show the performance of ABFP and QAT by emulating the precision of the photonic processor hardware in a CPU/GPU system in Table S1. The results show that Conv/Linear and Transformer networks are particularly amenable to overamplification, and the DLRM recommendation network is particularly resistant against quantization and noise.

Table S1 | Numerical study of the accuracy of MLPerf networks with ABFP and QAT.

Model Type	Model	Task	FP32 (%)	ABFP with G=1 (%)	ABFP with G=4 (%)	After QAT with G=4 (%)
Conv/Linear	ResNet50	ImageNet (acc.)	76.1	0.7	73.9	75.3
Conv/Linear	SSD-ResNet34	MS COCO (mAP)	24.7	7.1	23.5	24.7
Recurrent network	RNN-T	Librispeech (1-WER)	92.6	64.0	91.8	n.a.
Transformer	BERT-Large	SQuADv1.1 (F1)	93.5	5.4	90.5	92.3
3D Conv/Linear	3D U-Net	BRaTS 2019 (acc.)	85.3	80.1	85.2	n.a.
MLP	DLRM	1TB Click Logs (AUC)	80.4	80.2	80.3	n.a.

We confirmed that the photonic processor achieved an overamplification factor of 1.86. The calibration was performed by:

1. Turning off the output vector lookup tables.
2. Sending a tile of the same weight code and multiple vectors of the same value through the photonic tensor core and getting the output.
3. Computing the expected output code.
4. Dividing (2) by (3) and averaging across all the lanes and vectors.

II. Inference Throughput

For most workloads, executing 100s to 1000s of matrix-vector product (MVP) operations on a fixed set of weights is typical. To maximize utilization of the photonic tensor core during weight changes—an operation that requires approximately ten nanoseconds—a set of hardware optimizations minimizes downtime. Weights are double-buffered, enabling seamless streaming of the next set of weights in the background. This approach allows automatic weight selection during each MVP operation, synchronized by the hardware with the digital pipeline to ensure efficient, uninterrupted processing.

However, specific performance limitations and unresolved bugs reduce the system's output from its original design targets. Key issues include:

- The digital clock tree performance limits a maximum photonic tensor core clock frequency of 500 MHz.
- There is a non-functional automatic weight buffer selection mechanism, which necessitates a software-based weight change workaround. This software approach limits double-buffering capabilities and adds latency by exposing the weight settling time in the processing pipeline.
- Weight transfer from the digital chip to the photonic tensor core operates at 25 MHz.
- Software overheads related to device ISA management and PCIe DMA transfers.

Table S2 summarizes the execution performance for various network configurations, as detailed in Table 1. Four scenarios are presented:

- Hardware-based execution at 500 MHz without workarounds (dual 128x128 photonic tensor cores per digital chip, double-buffered weights, etc.).
- Hardware-based execution with all workarounds applied.
- A cycle-approximate simulator emulating the hardware without workarounds.
- A simulator run with all workarounds in place.

Scenario one approximates the hardware's design performance but at a reduced clock speed, while scenario two illustrates the impact of workaround-induced overheads. The simulation results at 500 MHz represent the device's raw compute capacity, excluding software overheads managing DMA and device ISA execution, along with operators assigned to the host CPU due to resource demands or lack of implementation. While these factors influence system-level throughput, simulation results underscore the core computational potential of the photonic tensor core and digital compute architecture.

All results were measured by timing the application's neural network calls, isolating compute-related activity from external factors such as Python-based input/output processing. Measurements were obtained using profiling instrumentation integrated directly into the software stack.

Table S2 | Photonic Accelerator Performance

Model Bype	Model	Task	HW Ideal (IPS)	HW w/ Mitigations (IPS)	Simulator Ideal (IPS)	Simulator w/ Mitigations (IPS)	Notes
Conv/Linear	Traffic Light	Traffic Light	74	69	2,071	1,658	B=1
MLP	3-layer linear	MNIST	321	214	31,863	15,411	B=50
Conv/Linear	Resnet18	MNIST					
Conv/Linear	Resnet18	ImageWoof	98	36	1,340	670	B=10; MNIST images upscaled to match ImageWoof
Conv/Linear	Resnet18	ImageNette	20	20	1,323	900	B=10
Conv/Linear	Resnet18	CIFAR10	480	468	22,214	16,216	B=100
Transformer	Bert-tiny	IMDB	458	39	1,907	752	B=1, sequence length=200
Transformer	Bert-tiny	Squad	323	34	1,859	689	B=1, sequence length=384
Conv/Enc-Dec	segnet	IIIT Pets	45	16	883	798	B=1

III. Packaging

The package dimension is 80 mm by 65 mm and is shown in Fig. S1(a). There are two DCIs each with 25,000 bumps on the top side of the interposer and four PTCs each with 6,000 bumps on the bottom of the interposer. The photonic tensor core is shown in Fig. S1(b). Top-view x-rays and cross-sectional x-rays are shown in Fig. S1(c). The fiber attach region of the photonic tensor core is shown in Fig. S1(d). There are 12 v-grooves per PTC with a pitch of 250μm.

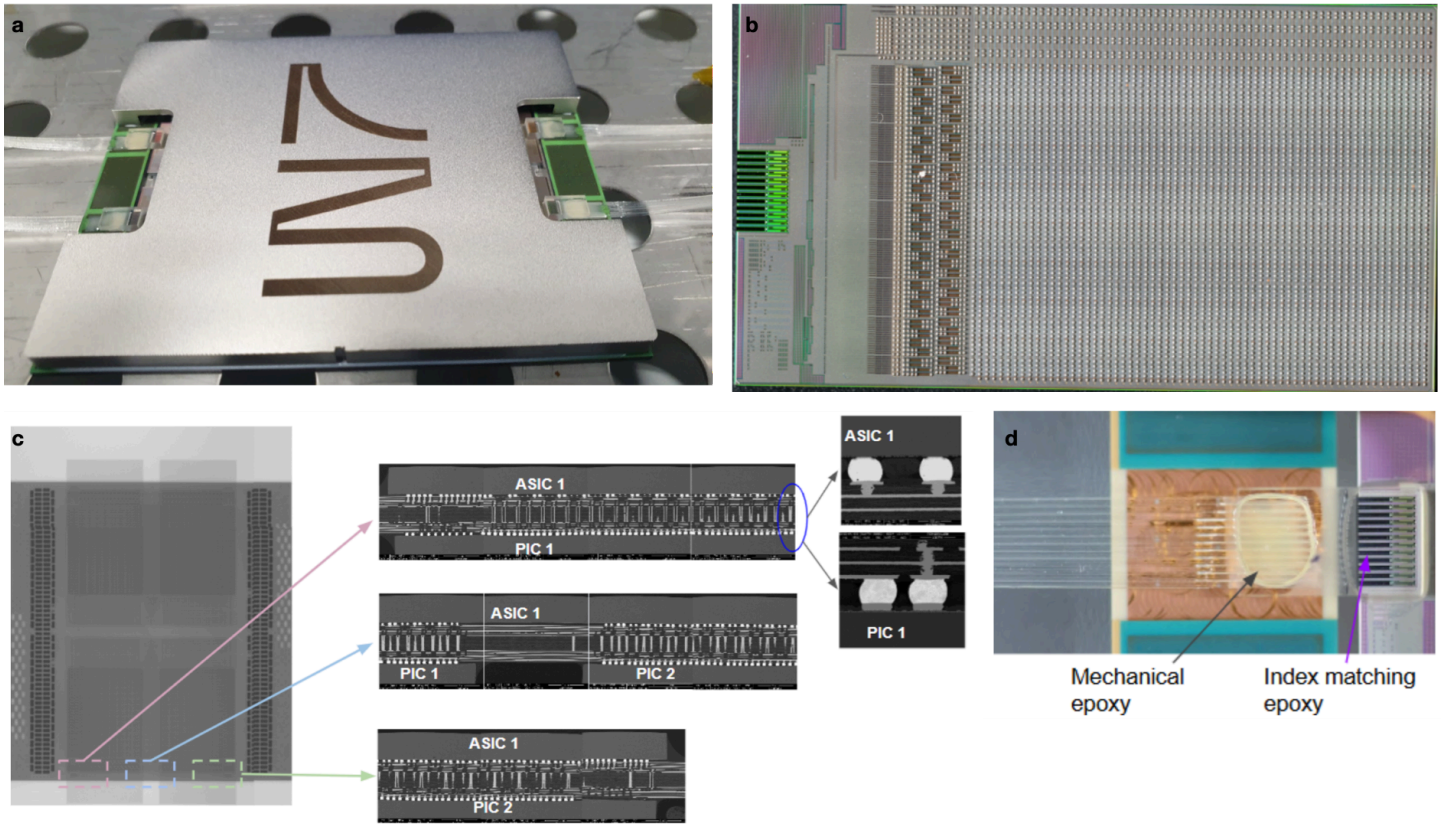

Fig. S1 | (a) Image of the complete photonic processor chip package. (b) Image of the photonic tensor core die with v-grooves visible. (c) 2D and cross-section x-rays of the photonic processor package. Interconnects visible. (d) Fiber attach assembly, including epoxy, for a photonic tensor core.

Summary Process Flow

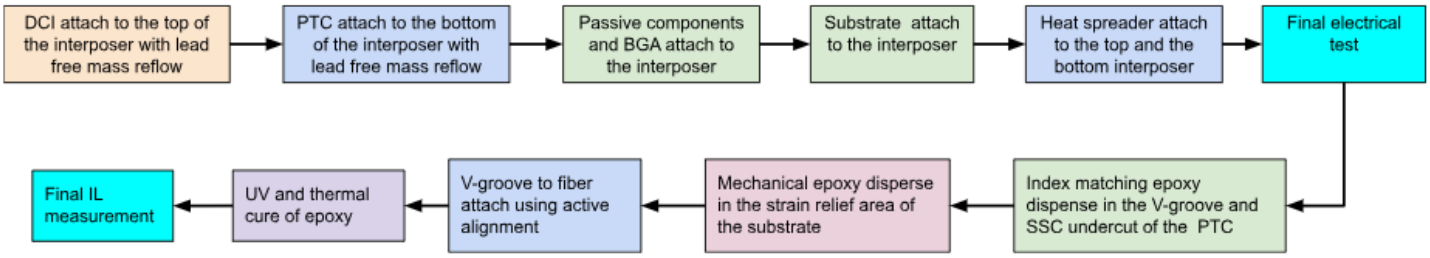

Fig. S2 | Flow diagram describing the process of building the processor package.

IV. Error Analysis

The logistic fit to the error data observed in Fig. 3(f) may arise from the interplay between analog noise, quantization effects, and preprocessing steps inherent to the photonic processor. Analog noise, predominantly Gaussian in origin due to thermal fluctuations, dark current mismatch, and device nonlinearities, undergoes transformations through quantization and aggregation processes during matrix-vector computations. Quantization discretizes the noise, while aggregation across multiple independent contributions regularizes the distribution, resulting in a composite behavior.

Preprocessing steps such as vector normalization further shape the noise distribution by amplifying tails and introducing scale-dependent effects, which align with the logistic distribution's heavier tails compared to a Gaussian. This combination of effects produces a distribution that captures characteristics of both Gaussian and logistic profiles, with the logistic fit particularly aligning in the tails of the observed data.

V. Photonic Tensor Core Calibration

The calibration process for the photonic tensor core ensures accurate and reliable operation by optimizing the performance of its key components. These components include the Laser Power Regulator (LPR), Vector Stabilizer (VSTAB), Weight Modulator (WMOD), Input Vector Modulator (VMOD), and the output subsystems comprising Transimpedance Amplifiers (TIAs) and Vector Analog-to-Digital Converters (VADCs). Calibration is critical for maintaining consistent output quality, especially given the inherent sensitivities of photonic computation to environmental and hardware variations.

Calibration is performed once per hour, a frequency determined heuristically to balance performance stability and computational overhead. This routine is managed by the photonic processor's operating system, NuttX, a free and open-source real-time operating system (RTOS) optimized for embedded systems. NuttX operates on the control processor, which consists of a quad-core 32-bit RISC-V SiFive E76MC complex, enabling efficient execution of the calibration steps.

Detailed Calibration Steps

The calibration process involves several steps, each targeting specific components and functionalities.

Step 1: VADC & IADC Self-Calibration and Tests

The initial calibration involves the Vector ADC (VADC) and Integrating ADC (IADC). The VADC undergoes radix calibration to correct for non-ideal behaviors such as capacitor mismatch, offset, inter-channel gain, and skew. The iADC self-tests ensure the proper operation of these components before proceeding to subsequent steps.

Step 2: Setting LPRs to Max Power (Low Accuracy)

In this step, each Laser Power Regulator (LPR) is set to approximately 92% of its maximum power using the internal iADC. This level is chosen to provide sufficient margin for potential laser power fluctuations while maintaining adequate optical power for subsequent operations.

Step 3: Setting VSTAB to Quadrature Point (Low Accuracy)

The Vector Stabilizer (VSTAB) is set to an approximate quadrature point using internal iADCs. The quadrature point is the state where the optical power at the two output ports of the Vector Modulator (VMOD) is roughly the same. This step ensures balanced operation of the VMOD.

Step 4: VADC Timing Calibration

This step involves calibrating the temporal delay between the VMOD and VADCs. Accurate timing is essential for proper synchronization and signal processing within the system.

Step 5: Calibrating VMOD Slope Signs

To ensure consistency in the differential currents drawn by the Weight Modulators (WMODs), the slopes of the VMODs are calibrated. This involves flipping the sign of the scale factor for all WMODs in a given column if the differential optical power flowing into WMODs as a function of HDAC code exhibits a negative slope.

Step 6: Calibrating VSTABs (Precise Calibration)

The VSTABs are now precisely calibrated using VADCs. This step refines the initial setting achieved in Step 3 by employing the more accurate VADCs, ensuring that the VSTABs are locked to a true quadrature point.

Step 7: Calibrating LPRs (Precise Calibration)

The LPRs are precisely calibrated to ensure that the amount of optical power going into the VMODs is roughly equal and remains constant over time. This involves determining the maximum common range of powers across all columns and requesting FSMs to lock to their respective codes.

Step 8: Calibrating WMODs

The WMODs are calibrated by adjusting their scale factors to match their slopes. This ensures accurate encoding of weight values by the ratio of currents drawn from row lines.

Step 9: Calibrating Input Vector (VMODs)

The input vector, represented by the VMODs, is calibrated to ensure accurate vector encoding. This involves sweeping through VADC codes and adjusting the VDAC LUTs to achieve the desired linearity and range.

Step 10: Calibrating Output Vector (TIAs+VADCs)

The output vector, consisting of the Transimpedance Amplifiers (TIAs) and VADCs, is calibrated to ensure accurate conversion of optical signals to digital outputs. This involves sweeping through input currents and populating the VADC LUTs to achieve the desired linearity and range.

The following criteria characterize a "good" calibration:

- LPRs maintain a consistent optical power level near their maximum, providing sufficient margin for fluctuations.
- VSTABs are accurately locked to their quadrature points, ensuring balanced operation of VMODs.
- WMOD slopes are matched, ensuring accurate encoding of weight values.
- Input and output vectors exhibit linearity and span the desired range, ensuring accurate signal representation and conversion.

VI. Comparative Analysis of Photonic Processors

To facilitate quantitative comparison between different photonic processor implementations, we introduce the Photonic Processor Performance Index (PPPI). This metric captures the key performance characteristics that determine a photonic processor's computational capabilities while accounting for implementation costs.

Metric Definition

The PPPI is defined as:

$PPPI = (T \times 2^B) / (P \times A)$

where:

- T: Throughput in TOPS (Tera Operations Per Second)
- B: Bit precision (product of weight and activation bits)
- P: Power consumption in Watts
- A: Chip area in mm²

This formulation captures the fundamental trade-offs in photonic processor design in terms of (TOPS x precision) / (W x mm²) The numerator represents computational capability: larger matrices enable more parallel operations, faster clock speeds enable quicker computation, and exponential bit precision (2^B) accounts for the exponential growth in numerical resolution. The denominator accounts for the implementation cost in terms of chip area and power consumption, primary constraints in practical implementations. The exponential scaling of bit precision reflects the fact that each additional bit doubles the precision of the computation.

Comparative Analysis

Table S3 | comparison of reported photonic processors, including all measured specifications and calculated PPPI values where sufficient data is available.

Reference	Matrix Size	Throughput	Bit Precision (W×A)	Clock Speed	Chip Area	Power Consumption	PPPI*	Notes
This work (2024)	4×128×128 (65,536)	65.5 TOPS	7×11 (77)	500 MHz avg, 2 GHz peak	349 mm²×4	78W electrical, 1.6W optical, 250mW / PTC	10^{20} all in 10^{23} PTC only	Full system implementation with ADCs/DACs
Xu et al. (2021)	10×3×3 (90)	10 peak, 3.8 avg TOPS	8-bit (64)	62.9 GHz	Not reported	Not reported	--	Processes 250k pixel images at 3.8 TOPS
Feldmann et al. (2021)	4×4 (16)	~1 TOPS†	8×8 (64)	1 GHz	~50 mm²†	146 mW	10^{18}	Phase-change photonic tensor core
Dong et al. (2023)	48×48 (2,304)	0.014 TOPS	8×8 (64)†	1 kHz	~100 mm²†	157 mW electrical	10^{16}	Higher-dimensional processing
Wang et al. (2022)	4×4 (16)	0.001 TOPS	8×8 (64)†	1 MHz	~25 mm²†	3.8 mW	10^{17}	Single-photon detection approach
Peng et al. (2022)	64×64 (4,096)	Unknown	7-bit (49)	1 GHz	Unknown	Not reported	--	Level of integration unclear

* PPPI = (Throughput × 2^Bit_Precision) / (Power_Consumption x Chip_Area)
† Estimated values based on paper descriptions
-- Cannot be calculated due to missing specifications

Methodology Notes

1. All throughput values are normalized to TOPS for consistency
2. Where activation bits are not specified, weight bits are used for both weight and activation precision
3. Estimated values (marked †) are derived from available information in publications
4. Power consumption includes both electrical and optical power where reported

These results distinguish this work. Balancing computational efficiency and accuracy for state-of-the-art AI models is critical. The power cost of adding a bit of precision to the activation readout ADC scales as $2^{2^{\Delta B}}$, where ΔB is the change in the number of bits. This equation is valid in the thermal noise-limited regime (where our ADC and many high-performance ADCs operate). The PTC presented here performs tensor operations with 11 bits of precision (set by the output ADC on the DCI)—the full processor uses the ABFP16 number format with floating point precision, rendering direct comparisons to other photonic computing results challenging.

A more direct comparison to the photonic processor literature can be achieved by isolating just the energy cost of encoding vectors and weights during the operation of the PTC, resulting in a power consumption of 0.25 W and a computational efficiency of 262 TOPS/Watt (including the weight digital-to-analog converter power cost). This highlights both the efficiency of the PTC architecture and the importance of accounting for total processor power requirements.

The computational latency is currently dominated by the communication latency of the PCIe 4.0 IP (milliseconds) and digital pipeline. This bottleneck was not apparent in prior photonic processor research since those efforts did not feature digital I/O capable of communicating with standard computing infrastructure like CPUs at appropriate speeds. While the current design does not prioritize computational latency, we note that the time-of-flight latency, which is the metric typically optimized in photonic tensor core research, is approximately 200 picoseconds and is set by the propagation time of the optical signal from the vector modulator to the weights and from the weights to the transimpedance amplifier. Future work can focus on low-latency communication protocols and realizing shallow digital pipelines after the PTC.

As discussed above, our clock tree design limited our chip to running at 500 MHz sustained clock—however, the peak clock frequency of 2 GHz results in 262 TOPS of ABFP16 throughput. Our digital chip architecture can be significantly optimized through clock gating and more advanced process nodes (12 nm), among many other optimizations. This processor fares well against state-of-the-art AI accelerators.

Table S4 | comparison of this work to the highest-performance AI processors in the market.

Year	Processor	Peak FP16, ABFP16 Throughput (TFLOPS)	TDP (W)	Peak TOPS/Watt
2017	Tesla V100	125	300	0.42
2018	Tesla T4	65	70	0.93
2020	A100 80GB	312	400	0.78
2022	H100	990	700	1.41
2023	L40S	366.5	365	0.99
2024	This Work	65 avg, 262 peak	78 avg, 150 peak	0.81 avg, 1.75 peak