
Supplementary Material For: Optical Transformers

Anonymous Author(s)

Affiliation

Address

email

Table 1: Model configurations for optical Transformers. $M = 10^6$.

Model	n	d	h	L	Non-emb. Params
Tiny	1024	192	12	12	15M
Small	1024	384	12	12	40.6M
Base	1024	768	12	12	123.7M
Large	1024	1536	12	12	416.3M

Table 2: Pretraining hyperparameters for optical Transformer models. All models were trained with the **AdamW** [15] optimizer.

Model	Steps	Batch	lr	β_1	β_2	ϵ	Weight decay	Dropout	Schedule	Warmup	Stop
Tiny	90000	32	2e-4	0.9	0.999	1e-8	0.02	0.1	Cosine	2500	-
Small	90000	32	2e-4	0.9	0.999	1e-8	0.02	0.1	Cosine	2500	-
Base	90000	32	2e-4	0.9	0.999	1e-8	0.02	0.1	Cosine	2500	-
Large	90000	32	2e-4	0.9	0.999	1e-8	0.02	0.1	Cosine	2500	82500

A Optical Transformer Training Hyperparameters

The optical Transformer models were pretrained on the Wikitext-103 [16] dataset and used the same tokenizer as GPT2 [20]. All models used **Xavier uniform initialization** [10]. The architectures are in Table 1. Embedding layers were initialized with a normal distribution with $\sigma = 0.02$. We used the AdamW [15] optimizer, with weight decay applied to parameters which were not embedding, gains, or biases. Dropout was applied after every linear layer (including those in attention), as well as on the attention matrix and after the $\text{softmax}(\frac{QK^T}{\sqrt{d_h}})V$ product in the attention calculation. The values of the parameters used for the training scheme are in Table 2.

After pretraining the models were quantized via our 8-bit QAT scheme. For QAT we used the RMSProp optimizer [26]. The parameters we used for the training are in Table 3. To clamp weights and activations we employ two different approaches: first, we kept running statistics of minimum and maximum values with an exponential moving average (EMA, with parameter α) for every layer and use those to clamp. Second, we recorded the minimum/maximum statistic throughout the network for a forward pass to apply a clipping scheme. Specifically, we clamped weights and activations to percentiles of the maximum values collected for each layer. The outputs were either rounded to the nearest integer during QAT, or stochastically rounded to nearby values. Finally, for the Base-sized model we used to run the experiments, we directly used the lookup tables (LUT) instead of “simulating” the quantization of inputs and weights (though outputs are still quantized). Table 4 details our use of these various techniques in the models.

For evaluation we used the perplexity (PPL) metric to measure the language modelling performance on Wikitext-103. We evaluated the perplexity over the entire validation set, and ran the model with context length 1024 (the same as in training) and a 1024-token stride length.

B ONN Experimental Procedure

B.1 Experimental Setup

Our setup is a SLM-based matrix-vector/vector-vector multiplier. The setup is shown in Figure 1 with a simplified illustration in Figure 2, and works as follows: Vectors corresponding to the inputs and weights are rearranged into squares of pixels and loaded onto the display and SLM respectively. They are aligned such that the light from display pixels will reach the corresponding pixels on the SLM. First, light from the display enters into the polarizing beam splitter (PBS), and reaches the SLM through a half-wave plate (HWP) which rotates its polarization. The phase is then modified by the SLM and reflected back through the half-wave plate, rotating the polarization again based on the phase difference. Then, the PBS only admits light of a certain polarization along one of its arms, aimed at a camera for detection. Summation of the output pixels is performed digitally. This SLM-HWP-PBS arrangement effectively creates an amplitude modulating SLM, where the output at each pixel is the element-wise product of the input pixel and corresponding weight pixel.

Table 3: Quantization aware training hyperparameters for optical Transformer models. All models were trained with the **RMSProp** [26] optimizer. Quantization parameters are in Table. 4.

Model	Steps	Batch	lr	α	ϵ	Weight decay	Dropout	Schedule	Warmup	Stop
Tiny	7327	64	1e-5	0.99	1e-8	1e-5	0.1	Cosine	2500	-
Small	7327	64	1e-5	0.99	1e-8	1e-5	0.1	Cosine	2500	-
Base	7327	64	1e-5	0.99	1e-8	1e-5	0.1	Cosine	2500	5500
Large	7327	32	1e-5	0.99	1e-8	1e-5	0.1	Cosine	2500	5500

Table 4: Hyperparameters for optical Transformer Quantization. We perform QAT with both a percentile-clipping approach and by clamping based on an exponential moving average (EMA) of model statistics with factor γ . For the Base-sized model that is used in our experiments (LUT-Base), we use lookup tables (LUT) for inputs and weights instead of quantization.

Model	Overall Config		EMA γ	Attention Clipping			Feed-Forward Clipping		
	Precision	Rounding		Input ₁	Input ₂	Output	Input	Weights	Output
Tiny	8-bit	Stochastic	-	99.99%	99.9%	99.9999%	99.99%	99.9%	99.9999%
Small	8-bit	Stochastic	-	99.99%	99.9%	99.9999%	99.99%	99.9%	99.9999%
Base	8-bit	Stochastic	-	99.99%	99.9%	99.9999%	99.99%	99.9%	99.9999%
Large	8-bit	Stochastic	-	99.99%	99.9%	99.9999%	99.99%	99.9%	99.9999%
LUT-Base	LUT	Stochastic	-	99.99%	98%	99.9999%	99.99%	99%	99.9999%
Tiny	8-bit	Deterministic	0.999	-	-	-	-	-	-
Small	8-bit	Deterministic	0.999	-	-	-	-	-	-
Base	8-bit	Deterministic	0.999	-	-	-	-	-	-
Large	8-bit	Deterministic	0.999	-	-	-	-	-	-

36 The OLED display has multiple color channels and a broad spectrum. For easier modulation by the
37 SLM, we used a band-pass filter and only green light.

38 The components we used are:

- 39 • Organic light-emitting diode (OLED) display (Google Pixel 2016)
- 40 • Reflective liquid-crystal modulator (1920-500-1100-HDMI, Meadowlark Optics)
- 41 • Half-wave plate (PH10ME-532, Thorlabs)
- 42 • Polarizing beam splitter (CCM1-PBS251, Thorlabs)
- 43 • Zoom lens for imaging onto SLM (Resolv4K, Navitar)
- 44 • Zoom lens and objective lens for imaging onto detector (1-81102, Navita and
- 45 XLFLUOR4x/340, Olympus)
- 46 • Band-pass filter (FF01-525/15-25, Semroc)
- 47 • Camera for detection (Prime 95B Scientific CMOS Camera, Teledyne Photometrics)

48 This setup works as a good bench for testing the precision of optical Transformers by performing
49 optical dot products involved in attention and feed-forward layers. Even though the optical dot
50 products were performed one at a time, it is sufficient for showing that Transformer operations
51 can run with the accuracy of ONNs, since matrix-vector and matrix-matrix products are merely
52 collections of many dot products run in parallel.

53 B.2 Calibration and Lookup Tables

54 We used several techniques to reduce errors, map inputs to SLM/display values, and to convert
55 detected outputs back to neural network values.

56 First, we developed a specialized data-pixel encoding scheme to reduce systematic errors. We noticed
57 that a large source of error was with a limitation of our hardware—in particular the SLM pixels have
58 cross-talk (pixels may affect their neighbors if they have very different values) and misalignment
59 in the experimental setup may lead to corrupted outputs. To help with these issues, we created
60 “macropixels”—each input element (and weight) does not occupy one pixel on the display (SLM) but
61 rather is mapped to a 3x3 grid of pixels, all with the same value. For the attention layers, we used 5x5
62 macropixels for the results we report, but later discovered that with 3x3 the performance is essentially
63 the same. We also rearranged vectors into square blocks of pixels so that significantly nonzero

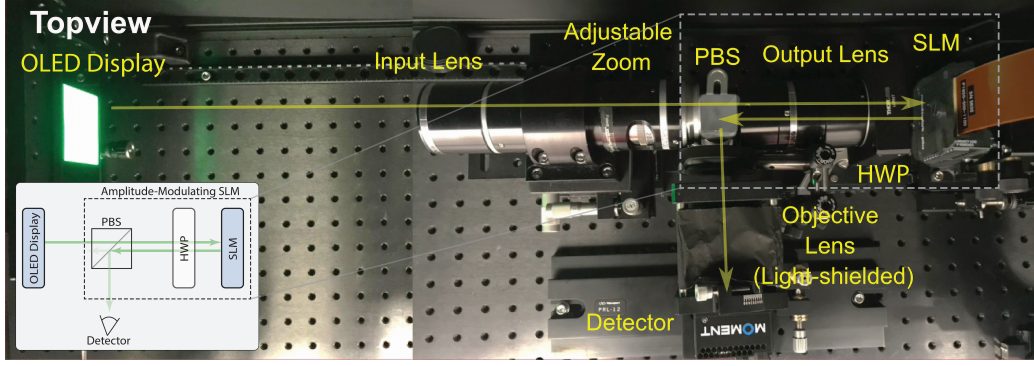


Figure 1: Photo of experimental setup used for running Transformer dot-product operations. Inset: simplified illustration of the experimental system. Spatial light modulator (SLM) + half-wave plate (HWP) + polarizing beam splitter (PBS) arrangement is effectively an amplitude-modulating SLM. The system works as follows: in our experiments, a vector is loaded as pixels on the organic light-emitting diode (OLED) display, and weights on the SLM. The input light enters through the PBS towards the SLM, passing through the HWP twice as the SLM reflects it. The SLM and HWP together rotate the polarization of the light, such that the amount reflected by the PBS towards the detector for each pixel is roughly the product between the pixel value and the corresponding weight on the SLM. The summation of these element-wise products by the detector yields the dot product.

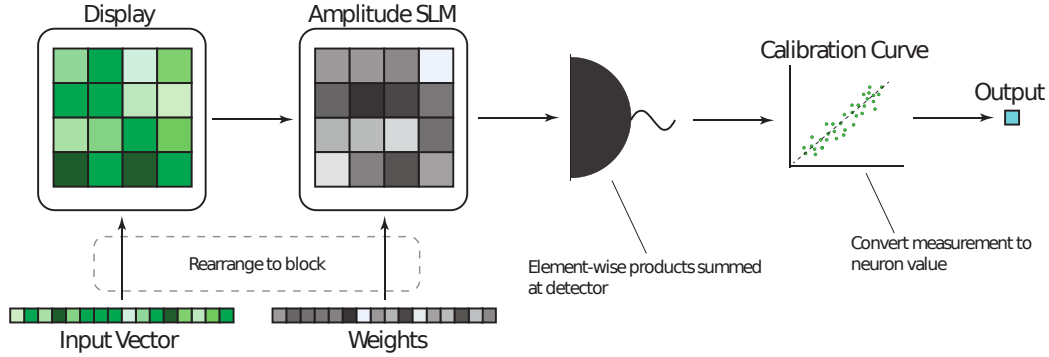


Figure 2: Simplified illustration of experimental setup operation. Weights are loaded and rearranged into a block on spatial light modulator (SLM) to prevent crosstalk between pixels of drastically different values. Data is rearranged on display accordingly. Measurements are looked up against calibration curve to obtain the final output value.

weights are nearby each other Figure 2. For the vectors to better fit in the center of the field of view (where there is less distortion/misalignment) we computed the dot products using only the 400 largest weight elements (the corresponding input elements are loaded). While this may introduce some inaccuracy in the final results, we found that the benefits of computing the element-wise products more accurately outweigh the drawbacks of pruning the weights; the outputs were still quite accurate to the ground-truth dot-product values (see main text, Figure 3). We suspect that this was the case because:

- Transformer weights are not entirely dense; some weights were already zero.
- Because our setup only supports non-negative data anyway, we use the four-pass approach (Section 3.3, main text). This means that for any given dot product, roughly half the weights and activations will be zero before considering the previously mentioned sparsity.
- Meanwhile, a second consequence of this four-pass approach is that roughly half of activations will be zero as well, possibly rendering some of the pruned weights irrelevant.

- Transformers still perform well when pruned, and luckily larger models can be pruned more heavily [13]. While our pruning method is quite basic, the number of weights pruned was light (ie. $< 75\%$) compared to what is possible with more advanced methods.

This approach was not necessary for attention calculations, since the dot products were sufficiently small to fit them entirely (64 elements).

Next, we consider the lookup tables (LUTs) of the display and SLM in the setup. In order to optimize the experimental results, the model used for experiment was trained to be aware of the realistic, discrete mappable values supported by the system. The display has a LUT with 256 unique levels (1000 levels total, but many are the same as others) and the SLM has roughly 128 unique levels (256 total). So they are roughly capable of 7 and 8 bit precision. The SLM also cannot fully extinguish input light—the minimum modulation is 2% of the maximum transmission. Thus, the minimum absolute values of the weights were mapped to 0.02 instead of 0.

After applying these approaches, we finally collected the calibration curve, which maps the output intensity measurements to neuron values in the neural network and allows us to determine the experimental setup’s systematic error. To do this we sampled randomly inputs and outputs of the layers we wished to run, computed their dot products both digitally and in experiment, and created a data set of experimental measurements and ground-truth-digital dot-product outputs. We then performed linear regression to find a mapping between experimental output and the correct values, effectively creating another lookup table. Then when future dot products were computed experimentally, the output was passed to this linear regression model (or it can literally be stored as a lookup table) to get the output. We used many photons and averaged outputs across multiple shots for each input, eliminating shot noise—any remaining error in this calibration scheme we defined as the system’s systematic error.

It is important to note that in general other optical systems might have different causes of error from ours, but the overall accuracy of our system is representative of a typical ONN nowadays.

B.3 Model Design Optimization

Transformers tend to have large dynamic ranges in their activations and weights [3]. In particular, we found that systematic error is proportional to some characteristic amplitude of the output. So, because it scales roughly with the sizes of outputs, having large outlier values can increase the systematic error and worsen the calibration for all other values in the representable range. Furthermore, after quantization in a naive, linear scheme, large outliers mean that huge ranges of outputs which are seldom used are assigned to many of the quantization levels, while the rest of the small, common outputs are squashed into few buckets—so the model precision is poor. This can be an issue when quantizing any deep learning model, but was exacerbated here by those systematic errors and the fact that the lowest levels of the weights are 0.02 and not 0.0. Therefore, we opted for an aggressive clipping scheme and the clamped activation ReLU6 when training the model to be run (Appendix A, **LUT-base** model); they reduce the dynamic range of inputs and weights and we found that they drastically improved the ONN’s ability to run Transformer operations with smaller error. Having fewer values in the 0.02 bucket of the SLM LUT also improved QAT training stability significantly. Even though the non-zero light extinction at 0.02 is caused by the specific SLM in our setup, such issues may happen with other optical implementations made of elements with finite extinction or resolution, and here we described a method to mitigate such issues by modifying training methods.

B.4 Transformer Dot Product Samples

While the speed and parallelism limitation of our setup made it intractable to run an entire Transformer model on it, we attempted to sample dot products to run that were representative of the range of possible activation/weight statistics in the model. That way, our results would be very representative of what running the full model would be like. In particular, we found two ways in which statistics throughout the model vary: the statistics change with depth (shallow and deep layers behave differently) and operation type (matrix-matrix multiplication in attention has different statistics from MLP layers). So, given our limited ability to run operations on the setup, we sampled roughly 10000 dot products from the first (QK^T) attention operation and second MLP layer of the first and last encoder layers of the model. The inputs to the whole model were samples from the Wikitext-103 dataset. Our approach captures the range of statistics throughout a model’s different components, over its depth,

Table 5: Simulated optical Transformer precision ablation. Input precision is degraded by subsampling from lookup table (LUT), while output is quantized. Input precision is approximate, as LUT has 1000 levels, not 1024. Bold: most compressed model found in our ablation with performance very close to the baseline.

Input Precision (LUT)	Output Precision	Val. Loss
~ 10 bits	32 bits	3.0059
~ 9 bits	32 bits	3.0057
~ 8 bits	32 bits	3.0054
~ 7 bits	32 bits	3.0039
~ 6 bits	32 bits	3.0034
~ 5 bits	32 bits	3.0017
~ 4 bits	32 bits	3.0111
~ 3 bits	32 bits	3.1223
~ 5 bits	8 bits	3.0032
~ 5 bits	7 bits	3.0074
~ 5 bits	6 bits	3.0335
~ 5 bits	5 bits	3.3966

and when processing a real task’s data. The second MLP layer has dot product size $4d$, making it the hardest to run experimentally.

In sampling the dot products, we tried to sample from both operands equally. For example, one could sample 1000 dot products by taking a single input vector and 1000 weight matrix vectors, and vice-versa, but choosing random vector pairs captures dot products involving different tokens and weights. This is important because Transformer output sizes, particularly the outlier activation values, are token-dependent [3]. To maintain this balance, we sample equal rows/columns for both operands. For attention layers we sample 100 from each; For linear layers, we sampled 56 rows from the input data and 200 columns from the weight matrix W^T , where the product being computed is xW^T .

C Simulated Precision Ablation Study

To further study how the optical Transformer can perform inference at lower precisions, we conducted a simple ablation on the input and output precisions used at inference, on the 8-bit-QAT base-sized model with LUT. We opted to leave the weights at 8-bit precision, since in-place weights are not a significant energy cost, and do not take more space/memory in these analog optical systems. In Table 5 is the performance of the model at lower precisions. With 5-bit input and 7-bit output precision, the model performs as well as the baseline. The reported precision values for the LUT are approximate, since the LUT has 1000 levels instead of $2^{10} = 1024$ levels.

When using the LUT, it is also not possible to directly change the precision of the input. Instead, we employed a subsampling scheme where the precision is degraded by rounding to every n ’th integer level before using the LUT, where n is a power of 2 and represents a reduction in the effective bit precision. The LUT of our display has 1000 levels, some levels have the same value, and we simulate the model without added noise. So we say that the original precision is initially *at most* 10 bits ($2^{10} = 1024$).

D ONN Energy Calculation

The models we used to estimate the energy use of ONN systems are in Table 6. We used a variety of real models that have been introduced by other works, and then designed our family of hypothetical future models **FUTURE-*** in a similar fashion, keeping a reasonable sequence length, increasing the embedding dimension drastically, and following the trend of recent large models like PaLM [7] and MT-NLG [23] of increasing the ratio d/h , which results in favorable energy calculations due to the lower fraction of memory operations in attention.

The calculation of energy costs for ONNs requires consideration of the entire system design and the costs of the surrounding electronics—since the optical computation itself is so cheap the electronics

Table 6: Designs of models used for energy estimates. Transformers have embedding dimension d , process sequence length n , use h attention heads, and have L layers. $M = 10^6$ parameters.

Model	n	d	h	L	Parameters	Reference
GPT2	1024	768	12	12	117M	[20]
GPT2	1024	1024	16	24	345M	
GPT2	1024	1280	20	36	762M	
GPT2	1024	1600	25	48	1.5B	
Megatron	2048	1536	16	40	1.2B	[22]
Megatron	2048	1920	20	54	2.5B	
Megatron	2048	2304	24	64	4.2B	
Megatron	2048	3072	32	72	8.3B	
GPT3	2048	768	12	32	125M	[4]
GPT3	2048	1024	16	24	350M	
GPT3	2048	1536	16	24	760M	
GPT3	2048	2048	24	24	1.3B	
GPT3	2048	2560	32	32	2.7B	
GPT3	2048	4096	32	32	6.7B	
GPT3	2048	5140	40	40	13B	
GPT3	2048	12288	96	96	175B	
Turing-NLG	1024	4256	28	78	17B	[21]
MT-NLG	2048	20480	128	105	530B	[23]
Chinchilla	2048	640	10	10	73M	[11]
Chinchilla	2048	1024	16	20	305M	
Chinchilla	2048	1280	10	24	552M	
Chinchilla	2048	1792	14	26	1.1B	
Chinchilla	2048	2048	16	28	1.6B	
Chinchilla	2048	3584	28	40	6.8B	
Chinchilla	2048	8192	64	80	70B	
PaLM-like	2048	4096	16	32	8B	
PaLM-like	2048	8192	32	64	62B	[7]
PaLM-like	2048	18432	48	118	540B	
FUTURE	2048	40960	80	120	2.4T	This work
FUTURE	2048	81920	128	200	16T	
FUTURE	2048	163840	160	400	129T	
FUTURE	2048	655360	512	800	4q	

account for nearly all of the energy cost. The way the energy is accounted for is as follows: The energy E_{load} can be broken down into three components, related to the energy of the cost of reading from memory E_{read} , digital-to-analog conversion (DAC) E_{DAC} , and modulation to generate the light E_{mod} :

$$E_{\text{load}} = E_{\text{read}} + E_{\text{DAC}} + E_{\text{mod}}. \quad (1)$$

Detection energy consumption E_{det} can be broken down in a similar fashion, where

$$E_{\text{det}} = E_{\text{detector}} + E_{\text{amp}} + E_{\text{ADC}} + E_{\text{write}} \quad (2)$$

represent the costs of detecting a signal, amplifying the detected signal, performing analog-to-digital conversion, and writing to memory respectively. There is also a cost of maintaining the weights in a weights-in-place system, which we call E_{maintain} . Because this cost scales per element, it is a per-MAC cost. But based on values from efficient commercial SLM systems, it is sufficiently small (and amortized by a large clock rate) that even the largest models we do estimations for are not bottlenecked. For optical energy, we take 1 eV (single-photon energy at 1240 nm). We started with using our measured 8-bit-performance photon count of 1500/MAC for the smallest model ($d = 192$) and rescaled the value for larger ones using the constant-per-dot-product trend which we know our simulated models can match or beat.

The assumptions we used were that weights would be loaded from off-chip memory like DRAM (in the case of a chunked-weights strategy; for a full weights-in-place, one-shot approach this cost does

not exist), and that the system uses large amounts of SRAM for activations [9]. We assumed that the system only needs 5 bits worth of input precision and 7 bits worth of output precision, per the results of our ablation on the base-sized model. We still assumed 8-bit memory accesses for convenience. The actual costs for the data access and weight maintenance were assumed to be these values:

- $E_{\text{read}} = 1$ pJ/bit for off-chip memory [25], and 0.3 pJ/bit for SRAM. The SRAM estimate is based on results for DNN accelerator measurements with 9.55 pJ/32-bit access [19, 8], and cutting edge/near-future assumptions for data transport from SRAM/cache [9]. [12] estimates 14 pJ per 64-bit access, or roughly 0.22 pJ/bit, for a recent TPU architecture.
- $E_{\text{DAC}} = 10$ pJ per 5-bit sample @ 10 GHz—this is achievable with 100 mW at 30.1dB SFDR [5].
- $E_{\text{mod}} = 1$ fJ/bit @ 110 GHz with thin-film lithium-niobate modulators [28].
- $E_{\text{amp}} = 2.4$ pJ per access. A transimpedance amplifier can run at 24 mW at 70 GHz [1]. We will just assume 10 GHz. $24\text{mW} / 10^{10} = 2.4$ pJ per element.
- E_{detector} is negligible compared to E_{amp} . For example, [17] calculates the cost of detection as the capacitive discharge, $\frac{1}{2}CV^2$, with capacitance $C \sim 1$ fF and voltage $V = 0.5$ V. This results in <500 aJ of energy consumption per detection. The cost is therefore negligible compared to amplification (E_{amp}).
- $E_{\text{ADC}} = 3.17$ pJ per 7-bit sample. 10 Ghz, need 7-bits of precision, so 128 conversion steps per sample – Achievable with 24.8 fJ/c-s [14] ($24.8 \text{ fJ} \times 128 = 3.17$ pJ per 7-bit sample).
- $E_{\text{write}} = E_{\text{read}}$. Actually, write access was measured to be cheaper than read access in [8], but we use $E_{\text{write}} = E_{\text{read}}$ as a simple, conservative assumption.
- $E_{\text{maintain}} = 0.002$ fJ/MAC. Assuming 2W for operation of a 10MP SLM, with inputs shone at 10 GHz (each pixel performs one MAC every cycle). There is not much information SLM power consumption for maintenance of a fixed pattern on the LCD panel, though more typical LCD displays which update can operate in the ~ 1 W regime. For example, [24] consumes 30 mW with 180000 pixels, which would scale to 1.67 W with 10MP (at worst, multiple SLMs/LCDs could be used in order to scale up).

E Breakdown-Of-Costs For Estimated ONN Energy Usage

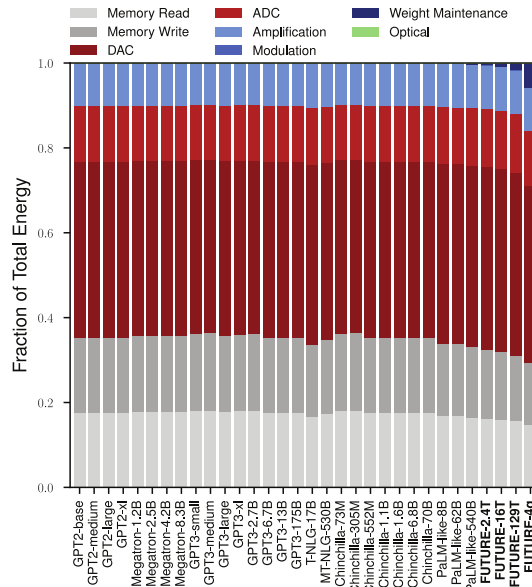


Figure 3: Breakdown of optical Transformer energy costs by energy type at 8-bit operation. Data access costs are dominant due to the high costs of DAC/ADC, but weight maintenance becomes important for large models.

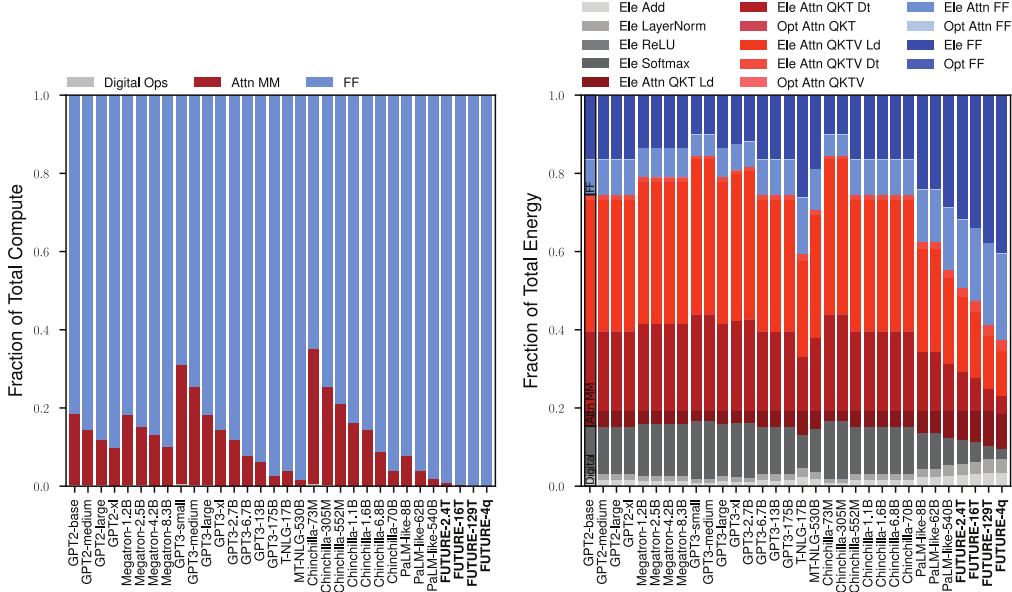


Figure 4: Breakdown of computing costs for optical Transformer models. Left: fraction of total compute used by digital operations, attention, and feed-forward components. Feed-forward layers account for most of the compute. Right: breakdown-of-costs for models by layer. The energy costs of attention operations is expensive. “Ele *” operations: electrical costs of loading (Ld), detecting (Dt), or both for data for the operation. Operations related to attention computation (ie. QK^T) are expensive for little compute. Functions computed digitally have their energy costs estimated as the cost of reading and writing to memory the required data.

In Figure 3 we see that data access costs, that is costs per element loaded/stored in memory, are most expensive. In particular, the cost of ADC and DAC are the leading contributors to the access costs, though since their cost is exponential in the bit precision, one might imagine that a future, optimized Transformer running at lower precision than our assumptions would have energy costs dominated by the actual SRAM memory costs. Also, for very large models, since the energy from weight maintenance scales with the number of MACs, it eventually will dominate if model sizes scale past that of FUTURE-4q. But future hardware would reduce E_{maintain} through improved electronics or higher clock speeds allowing for lower energy per MAC. Finally, the contribution from optical energy is vanishingly small, a consequence of the efficient photon usage scaling that we found Transformers can leverage. Were it not for this, the cost of actually performing the MACs would be orders of magnitude larger than everything else, resulting in energy usage that scales the same way as digital systems’.

Breaking down the sources of compute and energy costs in Transformer models running optically further illustrates how aspects of model/system design affect energy usage. The breakdown of compute and energy costs by source is in Figure 4. We find that as models get larger the feed-forward layers require most of the computation, but that the energy of data access in attention is still very expensive. This is because of the need to save/load many attention matrices from memory, and the fact that a weights-in-place scheme cannot be used for the matrix-matrix products because the products are between activations. Of course, this also means that there are more activations to load. In total, this means that attention layers have high energy costs for small amounts of computation. Thankfully, and interestingly, existing model design trends have moved towards focusing much harder on feed-forward layers, and so for the largest real (and our hypothetical future) models the fraction of energy cost taken by attention is low. Finally, we note that the operations we assume run on digital computers - such as nonlinear functions, in gray - do not account for much of the total energy cost (though they too are a small fraction of the total compute).

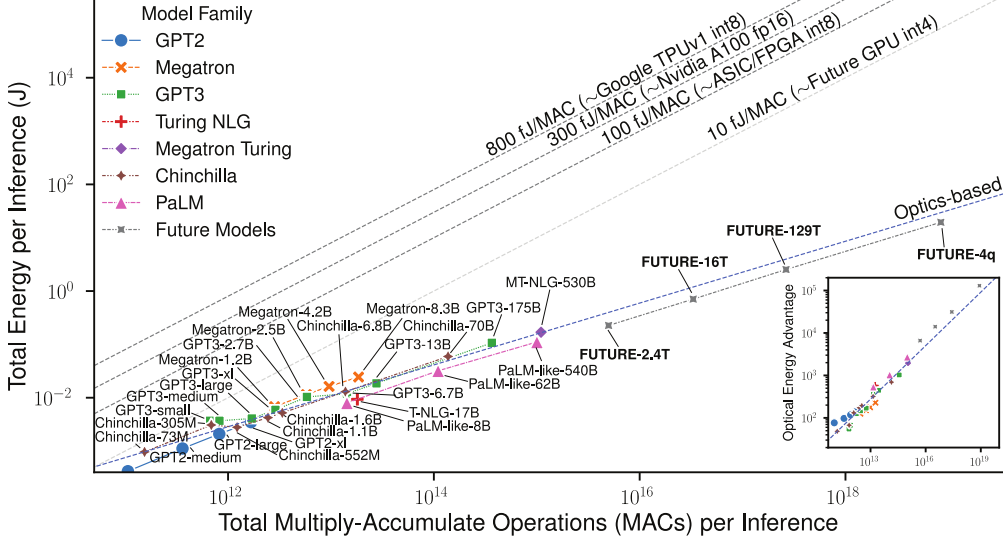


Figure 5: Energy usage estimates of forward pass for Transformers running on optical hardware, under future electronics energy cost assumptions. The energy advantages over our estimate for the current-day NVIDIA A100 GPU are larger than under our original assumptions (main text, Figure 5). $M = 10^6$, $G = 10^9$, $T = 10^{12}$, $q = 10^{15}$ parameters.

231 F Future ONN Energy Consumption

232 As optical accelerators are an emerging technology and as Transformer models continue to scale over
 233 time, it is worth considering how ONNs might improve over the next several years. For example, an
 234 interesting question to ask is how well future ONNs will do by the time it is possible to run a large
 235 model like FUTURE-4q. To investigate this, we estimated the energy costs of various Transformer
 236 models running optically again, but with the following changes and assumptions:

- 237 • $E_{\text{maintain}} = 0$ —Future weights-in-place hardware will need effectively no energy to main-
 238 tain weight information (for example, one might consider the usage of phase change materials
 239 [27]).
- 240 • E_{DAC} and E_{ADC} are 1/32 the size—we assume that electronics could achieve a $2\times$ improve-
 241 ment in fJ/c-s efficiency, while future advancements in model compression allow for 4-bit
 242 Transformer models, which are much cheaper since DAC and ADC costs scale exponentially
 243 with the number of bits [18].
- 244 • E_{read} and E_{store} are 1/5 the size—there is already a growing recognition of the fact that
 245 AI accelerators will need high efficiency and large quantities of SRAM and DRAM in the
 246 future [9, 6].
- 247 • E_{amp} $10\times$ cheaper (there are already cheaper trans-impedance amplifiers than our conser-
 248 vative estimate here, and receiver-less configuration without any amplifier has also been
 249 demonstrated [2]).

250 Under these assumptions, ONNs become far more efficient, highlighting that improvements to
 251 electronics will impact ONNs, and not just competing digital hardware. The energy scaling (Figure 5)
 252 is shifted downward for optics compared to under our previous assumptions, leading to over $1900\times$
 253 and $130,000\times$ advantages over the current A100 GPU for MT-NLG and FUTURE-4q models
 254 respectively. Of course, by the time this is possible, GPU efficiency will have improved significantly
 255 as well, and we are comparing a 4-bit accelerator to the 16-bit performance of the A100. It is difficult
 256 to predict the future efficiency of GPUs at lower precision, but it is clear that ONNs can benefit from
 257 improvements to electronics and low-precision inference.

G Scaling ONNs: System Specifications and Communication Costs in Multi-Processor ONN and GPU Setups

Implementation of a real ONN for large models might be difficult because the amount of hardware needed to maintain all the weights is exceedingly large. In Table 7 are the requirements for hardware to run the largest future model. To compute the number of weights/elements, we selected the largest MLP layer in the model, since that requires the most space for weights and activations. While detector and memory requirements are achievable, the number of required cores—each an optical component capable of performing 10M multiplications with weights—is enormous. There are some approaches to remedy this kind of memory issue in both GPUs and ONNs, and we are interested in their hardware-time-energy tradeoffs for ONNs.

One solution is to introduce chunking, where only a portion of the weights are loaded at a time, and the inputs are passed through. Then, the amount of time it takes to run is increased by a factor of the number of chunks. This also impacts the optical system’s energy advantage over digital ones in two ways. First, the weights must be loaded, but the cost can be amortized via reuse with batched inference. This comes at the expense of latency. This is a new kind of tradeoff, since digital systems cannot reuse weight data for free. Second, for each weight chunk, all inputs must be reloaded; changing the chunk number trades energy efficiency for lower hardware requirements. These energy tradeoffs are illustrated in Figure 6; other factors dominate energy usage until the chunk number is large and chunking becomes the bottleneck.

Realizing large models with GPUs will likely also require a multi-GPU strategy, which will incur overhead over the peak performance of a single GPU. We find that with a simple model of communication costs—modelling the activation reloading in both GPU and ONN systems—that ONNs can retain some of their advantages, dependant on how much system memory (or maximum number of weight elements) is available per-processor. We created a simple model to estimate the cost of this approach in GPU systems. In GPU systems, instead of splitting a model over time, the model may instead be split over multiple GPUs. This introduces an analogous tradeoff to the activation reloading in ONNs due to communication costs: if each GPU holds some chunk of weights, then after every layer, the outputs of multiplying the inputs with each chunk must be broadcasted to every GPU in an all-to-all fashion. This is in essence an all-reduce operation—after every layer, the outputs from all GPUs must be copied onto all GPUs. In total, this means the total number of activations is loaded k times, where k is the number of GPUs. As a crude but conservative estimate of these costs, we modeled this by taking the cost of running the entire model on one GPU, and then adding the energy cost of loading the activations from DRAM, multiplied by the number of chunks (GPUs). This is likely an underestimate, as broadcasting data across GPUs in a real setup requires sending data electronically over much longer distances than required for DRAM access, which would be expensive.

To determine the number of chunks, we tested multiple assumptions about device memory. We assumed a value for the amount of memory that can be used to store weights and take the total number of weights for each model divided by this memory capacity to determine the number of chunks to be used.

With these models, we found that too much chunking is detrimental to ONN performance, but that there is still some energy advantage to be had if it is used sparingly (Figure 6). In Figure 7 (top) are the energy cost estimates assuming a fixed memory of 100M weights (ie. 100MPixel SLM, or RAM with 100MB capacity if each weight is one byte). We assumed that for GPU, the cost of communication is at least that of DRAM-level communication due to the physical distances between GPUs. The curves for GPUs bend upward as the communication costs begin to take over, as do the largest models running optically. The ONNs still maintain an advantage, but the advantage stops growing with model size. Looking at the energy advantage illustrates this idea more clearly: up to a certain model size the advantage is increasing, then as the model size reaches the memory limit it begins to level off, and then the advantage begins to shrink as the cost of chunking takes over. For a small range of model sizes near this peak, the advantage is maintained, suggesting that a small amount of chunking may be useful before it quickly diminishes the energy advantage.

The optimal configuration for ONNs, obviously, is to have enough memory (cores which have weights fixed in place) so that chunking is not necessary. When plotting the advantages for larger memories (and therefore fewer chunks), the advantage gets better, and larger models become worthwhile to

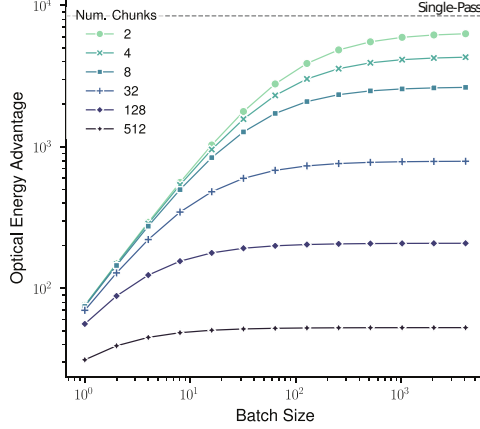


Figure 6: Optical energy advantage vs A100 (FUTURE-4q). When chunking, the cost of loading weights is amortized by increasing batch size, but the overall performance is limited by large numbers of chunks because of input data reloading.

run. In hindsight this conclusion makes sense: the benefit of ONNs is their ability to copy data (“optical fan-out”) for free for parallel computation, and so reducing this in favor of repeated memory accesses removes exactly the mechanism that gives optics-based systems their advantages. This also suggests that an “optical memory” from which fixed data can be accessed for free (or significantly less than re-access through electronics) may solve this problem, allowing for more scalable ONN design without huge amounts of hardware for weights. Currently, optics still has an advantage when using multiple cores because in principle the data could be fanned out across cores, while GPUs must pay communication costs in multi-processor setups. With a fan-out/fan-in design that can collect/spread a vector across cores, the efficiency of an entirely weights-in-place system is fully that of a single, large core.

Table 7: Requirements for optical accelerator running feed-forward layer (embedding dimension d , sequence length n) without chunking at 8-bit precision. The requirement of many cores to maintain weights for matrix-vector products (MVM) is high, and we assume the ONN system requires static RAM (SRAM) for saving and loading activations.

Model	Input Vector Elements	Detectors	MVM Cores (10^7 weights each)	SRAM (activations)
FUTURE-4.1q	2.6×10^6	2.6×10^6	170,000	5.37 GB
FUTURE-129T	6.55×10^5	6.55×10^5	11,000	1.34 GB
FUTURE-16T	3.28×10^5	3.28×10^5	2,700	671 MB
FUTURE-2.4T	1.64×10^5	1.64×10^5	671	336 MB
PaLM-like-540B	7.37×10^4	7.37×10^4	136	151 MB
MT-NLG-530B	8.19×10^4	8.19×10^4	168	168 MB
GPT3-175B	4.91×10^4	4.91×10^4	61	100 MB
General	$4d$	$4d$	$4d^2/10^7$	$4nd$

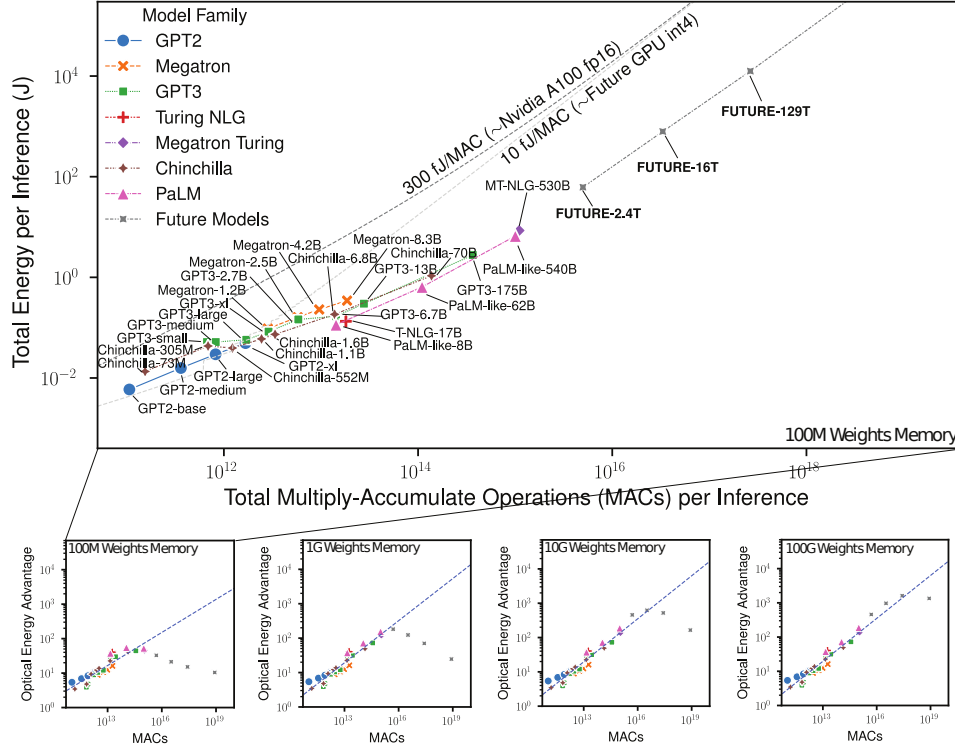


Figure 7: Energy estimates assuming a fixed processor memory size and chunking. Top: estimated energy scaling plot for Transformer models running on optical and digital hardware with 100MB of memory. As models get larger, both optical and digital systems have an upward bend in energy consumption trends, driven by communication/input-reloading-from-chunking costs. Bottom: energy advantage scaling for different memory sizes. As the memory increases, there is a maximum energy advantage for optics over NVIDIA A100 and corresponding model size before chunking costs take over. $M = 10^6$, $G = 10^9$, $T = 10^{12}$, $q = 10^{15}$ parameters.

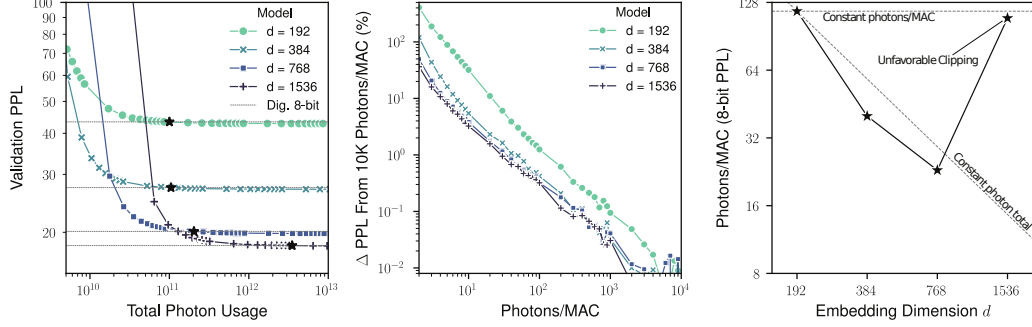


Figure 8: Behavior of optical Transformer models with varying photon usage with percentile clipping scheme. Left: Wikitext-103 validation set perplexity (PPL) versus embedding dimension d and total photons usage. 8-bit quantized digital model performance levels in dashed lines. Middle: Percent change in perplexity from ideal 10000 photon count performance still exhibits truncated power-law scaling with photons per multiply-accumulate (MAC) operation for all models. Right: Scaling of photon usage for maintaining the 8-bit digital performance versus model size. Dashed lines: constant photons per dot product (optical scaling) and constant photons/MAC analogous to digital scaling. Note that unlike for our results in the main text, smaller models beat the constant-dot-product-total scaling, but the largest model exhibits poor efficiency, as the clipping scheme used here was not well suited for it.

H Effects of Training and Quantization Scheme on Optical Scaling

Our results demonstrating favorable scaling of photon usage in Transformers show that they can be optically efficient, but in general the photon usage is affected by the training scheme and other factors like quantization. This is because approaches for optimization quantization, regularization, etc. affect the statistics of weights and activations in the network, which unlike digital systems, are tied to the resource usage. The main example of this is with weights: they are normalized before being loaded onto an ONN accelerator, and so large outliers may lead to many weights being near 0 after normalization—admitting fewer photons through to the detector. This has a direct impact on the output SNR, and so depending on weight statistics more or fewer photons may be needed in order to run at the same precision.

To discover how a different scheme might affect photon usage, we analyzed the optical scaling of our quantized optical Transformer models with percentile clipping instead of clamping based on EMA statistics. We applied the same clipping to all models (details in Table 4). These clipped models have familiar trends in their language modelling performance versus photon numbers, but we notice key differences in the photons needed to maintain 8-bit digital performance: first, the absolute number of photons needed for the smaller models (120 and 40 versus 340 and 170 of our unclipped scheme for $d = 192, 384$) is much lower—this indicates that clipping of large weight values leads to more transmission after normalization. Second, the scaling is inconsistent, with smaller models needing significantly fewer photons than the expected $1/d$ scaling, but then requiring many photons again for the largest model. The clipping scheme degraded the performance of the large model. Of course, this could be improved by designing a better scheme for the largest model such that it requires few photons; these results illustrate how differences in the training and quantization recipe could lead to a variety of outcomes, and why efficiency is achievable but not an automatic guarantee for any scheme.

References

- [1] Maruf N. Ahmed, Joseph Chong, and Dong Sam Ha. A 100 Gb/s transimpedance amplifier in 65 nm CMOS technology for optical communications. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1885–1888, 2014. doi: 10.1109/ISCAS.2014.6865527.
- [2] Saumil Bandyopadhyay, Alexander Sludds, Stefan Krastanov, Ryan Hamerly, Nicholas Harris, Darius Bunandar, Matthew Streshinsky, Michael Hochberg, and Dirk Englund. Single chip photonic deep neural network with accelerated training. *arXiv preprint arXiv:2208.01623*, 2022. URL <https://doi.org/10.48550/arXiv.2208.01623>.
- [3] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and overcoming the challenges of efficient Transformer quantization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7947–7969, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL <https://aclanthology.org/2021.emnlp-main.627>.
- [4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL <https://arxiv.org/abs/2005.14165>.
- [5] Pietro Caragiulo, Clayton Daigle, and Boris Murmann. DAC performance survey 1996-2020. URL <https://github.com/pietro-caragiulo/survey-DAC>.
- [6] Cerebras Systems. Cerebras systems: Achieving industry best AI performance through a systems approach. Technical report, Apr 2021. URL <https://8968533.fs1.hubspotusercontent-na1.net/hubfs/8968533/Whitepapers/Cerebras-CS-2-Whitepaper.pdf>.
- [7] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayanan Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling language modeling with Pathways, 2022. URL <https://arxiv.org/abs/2204.02311>.
- [8] Benoît W. Denkinger, Flavio Ponzina, Soumya S. Basu, Andrea Bonetti, Szabolcs Balási, Martino Ruggiero, Miguel Peón-Quirós, Davide Rossi, Andreas Burg, and David Atienza. Impact of memory voltage scaling on accuracy and resilience of deep learning based edge devices. *IEEE Design & Test*, 37(2):84–92, 2020. doi: 10.1109/MDAT.2019.2947282.
- [9] Yaosheng Fu, Evgeny Bolotin, Niladrish Chatterjee, David Nellans, and Stephen W. Keckler. GPU domain specialization via composable on-package architecture. *ACM Trans. Archit. Code Optim.*, 19(1), dec 2021. ISSN 1544-3566. doi: 10.1145/3484505. URL <https://doi.org/10.1145/3484505>.
- [10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL <https://proceedings.mlr.press/v9/glorot10a.html>.

- [11] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL <https://arxiv.org/abs/2203.15556>.
- [12] Norman P. Jouppi, Doe Hyun Yoon, Matthew Ashcraft, Mark Gottscho, Thomas B. Jablin, George Kurian, James Laudon, Sheng Li, Peter Ma, Xiaoyu Ma, Thomas Norrie, Nishant Patil, Sushma Prasad, Cliff Young, Zongwei Zhou, and David Patterson. Ten lessons from three generations shaped Google’s TPUv4i : Industrial product. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 1–14, 2021. doi: 10.1109/ISCA52012.2021.00010.
- [13] Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E. Gonzalez. Train large, then compress: Rethinking model size for efficient training and inference of transformers. In *Proceedings of the 37th International Conference on Machine Learning, ICML’20*. JMLR.org, 2020.
- [14] Juzheng Liu, Mohsen Hassanpourghadi, and Mike Shuo-Wei Chen. A 10GS/s 8b 25fJ/c-s 2850um² two-step time-domain ADC using delay-tracking pipelined-SAR TDC with 500fs time step in 14nm CMOS technology. In *2022 IEEE International Solid- State Circuits Conference (ISSCC)*. IEEE, February 2022. doi: 10.1109/isscc42614.2022.9731625. URL <https://doi.org/10.1109/isscc42614.2022.9731625>.
- [15] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2019. URL <https://openreview.net/forum?id=Bkg6RiCqY7>.
- [16] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017. URL <https://openreview.net/forum?id=Byj72udxe>.
- [17] David A B Miller. Attojoule optoelectronics for low-energy information processing and communications. *Journal of Lightwave Technology*, 35(3):346–396, 2017. URL <https://doi.org/10.1109/JLT.2017.2647779>.
- [18] Boris Murmann. ADC performance survey 1997-2020. <http://web.stanford.edu/~murmman/adcsurvey.html>, 2020. Online Accessed: 2021-02-18.
- [19] Flavio Ponzina, Miguel Peon-Quiros, Andreas Burg, and David Atienza. E²CNNs: Ensembles of convolutional neural networks to improve robustness against memory errors in edge-computing devices. *IEEE Transactions on Computers*, 70(8):1199–1212, August 2021. doi: 10.1109/tc.2021.3061086. URL <https://doi.org/10.1109/tc.2021.3061086>.
- [20] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL <https://openai.com/blog/better-language-models/>.
- [21] Corby Rosset. Turing-NLG: A 17-billion-parameter language model by Microsoft, Feb 2020. URL <https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/>.
- [22] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism, 2019. URL <https://arxiv.org/abs/1909.08053>.
- [23] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. Using DeepSpeed and Megatron to train Megatron-Turing NLG 530B, a large-scale generative language model, 2022. URL <https://arxiv.org/abs/2201.11990>.

- 448 [24] Sony Corporation. *LCX027AKB Datasheet (PDF) - Sony Corporation*. Sony Corpo-
 449 ration. URL [https://pdf1.alldatasheet.com/datasheet-pdf/view/47550/SONY/](https://pdf1.alldatasheet.com/datasheet-pdf/view/47550/SONY/LCX027AKB.html)
 450 [LCX027AKB.html](https://pdf1.alldatasheet.com/datasheet-pdf/view/47550/SONY/LCX027AKB.html).
- 451 [25] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep
 452 neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.
 453 URL <https://doi.org/10.1109/JPROC.2017.2761740>.
- 454 [26] Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running
 455 average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):
 456 26–31, 2012.
- 457 [27] Matthias Wuttig, Harish Bhaskaran, and Thomas Taubner. Phase-change materials for non-
 458 volatile photonic applications. *Nature photonics*, 11(8):465–476, 2017. URL <https://doi.org/10.1038/nphoton.2017.126>.
- 460 [28] Mengyue Xu, Yuntao Zhu, Fabio Pittalà, Jin Tang, Mingbo He, Wing Chau Ng, Jingyi Wang,
 461 Ziliang Ruan, Xuefeng Tang, Maxim Kuschnerov, et al. Dual-polarization thin-film lithium
 462 niobate in-phase quadrature modulators for terabit-per-second transmission. *Optica*, 9(1):61–62,
 463 2022.