I've Got 99 Problems But FLOPS Ain't One

Alexandru M. Gherghescu[1] Vlad-Andrei Bădoiu[1] Alexandru Agache[1]
Mihai-Valentin Dumitru[1] Iuliu Vasilescu[1] Radu Mantu[1] Costin Raiciu[1,2]
[1]University Politehnica of Bucharest [2]Broadcom Inc.

Abstract

Hyperscalers dominate the landscape of large network deployments, yet they rarely share data or insights about the challenges they face. In light of this supremacy, what problems can we find to solve in this space? We take an unconventional approach to find relevant research directions, starting from public plans to build a $100 billion datacenter for machine learning applications [53]. Leveraging the language models scaling laws, we discover what workloads such a datacenter might carry and explore the challenges one may encounter in doing so, with a focus on networking research. We conclude that building the datacenter and training such models is technically possible, but this requires novel wide-area transports for inter-DC communication, a multipath transport and novel datacenter topologies for intra-datacenter communication, high speed scale-up networks and transports, outlining a rich research agenda for the networking community.

CCS Concepts

• **Networks** → **Data center networks**; **Network design principles**.

Keywords

Datacenter Networking, Large Language Models Training

ACM Reference Format:

Alexandru M. Gherghescu, Vlad-Andrei Bădoiu, Alexandru Agache, Mihai-Valentin Dumitru, Iuliu Vasilescu, Radu Mantu, Costin Raiciu. 2024. I've Got 99 Problems But FLOPS Ain't One. In *The 23rd ACM Workshop on Hot Topics in Networks (HOTNETS '24), November 18–19, 2024, Irvine, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3696348.3696893

1 Introduction

The past few years have witnessed a revolution in the field of natural language processing, with large language models (LLMs) emerging as extraordinary tools for a wide range of language tasks (summarization, translation, classification, natural language inference, question answering, text generation [5, 8, 13, 35, 37, 48, 72]). The underlying assumption is that the number of parameters of a model is the single biggest indicator of model quality on a diverse range of benchmarks [42, 50, 58, 60]. It is then understandable that many companies invest resources in training such models. As public models reach hundreds of billions of parameters [8, 11, 15, 47, 60], and closed models like ChatGPT presumably surpass the trillion-parameter threshold [55], scaling laws [25, 32] predict no immediate end to this growth. We are in the midst of a race to develop ever-larger models, with datacenter computing at the heart of this competition.

Training language models only requires matrix multiplications, but the sheer size of these matrices makes compute a valuable resource. Accelerators (GPUs, TPUs) have emerged as the de-facto standard of model training, dominating both compute and network traffic generation in AI-focused datacenters. Clusters comprising tens of thousands of accelerators are used currently for training Transformer-based [72] models. Distributed training frameworks [42, 50, 76] have been developed to mask network bottlenecks and keep the accelerators busy during model training. It is well understood today that the network is the primary bottleneck, as various works [11, 31, 33, 42, 51, 67] have reported being able to use accelerators at only 40-60% capacity when scaling-out, due to the inherent limitations present in large-scale model training (synchronization points, large number of peers etc.).

What are the main challenges in building a $100B datacenter? Foremost among these is cooling and securing sufficient electrical power, as we anticipate consumption on the order of gigawatts — a scale not readily available in a single location within the United States. Once these fundamental infrastructure challenges are addressed, we encounter the primary technical impediment: training scalability. As we approach systems with millions of GPUs, several critical questions arise: How will training methods adapt to such massive scale? What will be the upper limits of model size? How will networking requirements evolve to support this unprecedented level of computation and data movement?

This study provides insights into the future of large-scale AI infrastructure and the technical hurdles that must be overcome to realize the next generation of massive language models, with the primary focus on model training.

HOTNETS '24, November 18–19, 2024, Irvine, CA, USA

ACM ISBN 979-8-4007-1272-2/24/11
https://doi.org/10.1145/3696348.3696893

2 Infrastructure

We base our calculations on the recently announced Blackwell GB200 GPU from Nvidia [29, 54, 56], with novel NVL72 (72 GPUs / rack) packaging [43, 65]. The NVL72 racks cost $3M, have a TDP of 130 kW in full load [22, 26], and can output 1,440 PFlops in sparse FP4 computation (half of that for dense operations). Each GPU has 192 GB of memory and NVLink speeds of 14.4 Tbps (the so called *scale-up network*), with a 800 Gbps NIC connected to the *scale-out network*.

Following current trends in data center budgeting [74], an allocation of 70% of the budget to compute seems reasonable, giving us an approximative 23.3K racks containing **1.67M GPUs** with a maximum power consumption of about **3 GW** and **16,800,000 PFlops** of FP4 performance for dense operations (no sparsity) . That is, the datacenter can sustain $16.8e21$ FP4 floating point operations per second in full load.

Assuming that storage and networking require an additional fifth of the power needed for compute [74], we calculate a total power consumption of 3.6 GW for IT equipment. Combined with Microsoft's reported Power Usage Effectiveness (PUE) numbers, mainly between 1.15 and 1.3 [40], the total power consumption of the data center would end up between the ranges of 4.16 GW and 4.71 GW; this is consistent with public reports about OpenAI datacenter plans [6]. Where can we find this much energy in the US?

One option is to build new generation capacity, but this takes time. The easier option is to use existing spare capacity in the near future. We surveyed US grid operators [71] and identified the maximum disposable power for all grids by computing the difference between the max registered power generation and max registered power demand in the last two years, with the most promising shown in Table 1.

Balancing Authority	Max Available (MW)	Region
PJM	9915	Mid-Atlantic
SRP	2634	Southwest
NEVP	2209	Northwest
BPAT	2143	Northwest

Table 1: Maximum available energy for the top energy producer grids in the US.

While PJM is a viable candidate, it covers multiple states. To identify a single geographic point, we grouped power sources within a specific radius into power nodes. We then connected these nodes with vertices representing high-voltage transmission lines. We excluded nodes where coal dominates energy production, as coal plants are being phased out gradually [9]. Additionally, considering that nuclear power plants require downtime for maintenance and refueling [70], there needs to be enough interconnect capacity the size of the largest producer in each node to ensure uninterrupted datacenter operation. Following this approach , we couldn't find a single location meeting all requirements. However, expanding our search to a 100 km radius revealed several suitable options near Washington city. Consequently, from a power perspective we are constrained to divide the datacenter into multiple units. This is consistent with news from Microsoft [16] and chinese hyperscalers [66] that distributed training across datacenters is needed, and indeed possible.

Given the need to divide the datacenter, we propose an east-west coast split due to several advantages. This geographical separation allows us to tap into diverse renewable energy sources rather than relying on the sources available in a single region. It also enhances fault tolerance against both natural and man-made disasters, aligning with the east-west division of the US power grid. Additionally, having datacenters on both coasts could provide low-latency inference capabilities to users in both regions after the initial training phase. For the west coast unit, a suitable location could be in the Northwest regions, close to the Pacific DC Intertie powerline. We have not explored splitting the datacenter into three or more locations, leaving such analysis for future research.

3 Model scaling

What model can we train using the above infrastructure? We start with the classical Transformer model [72], since it is the most studied and understood architecture, with the fairly standard architectural variations (RoPE [61], GeLU activation [24], pre-layer normalization [8], no shared embeddings [8, 67, 68], full attention heads [8], no biases [8, 67, 68]), in order to draw bounds on model size. We later explore Mixture of Experts as well. We ignore linear attention and sub-quadratic models [19, 45, 63] in this work, due to existing concerns for benchmark and long-range dependencies performance, and also because the training time should be similar.

3.1 Scaling laws

Given the estimated compute budget found in section 2, we proceed to determine suitable model sizes. We use true FP4 arithmetic precision supported by hardware, instead of quantization (on-the-fly decoding at compute time), in an attempt to maximize device compute and memory utilization. We leave such approaches to future research, as we're mainly concerned with networking, but note that there have been avenues studying lower-precision training [18, 38, 62, 73].

Training time. Based on scaling laws shown in Fig. 1, assuming no restrictions on interconnect or bandwidth and full GPU utilization, we select a few suitable model sizes that can be trained on our target datacenter. We look at both Kaplan et al. [32] and the 3 approaches from Hoffmann et al. [25], and choose Hoffmann et al.'s approach number 2 (see Fig. 1). We have examined in detail both 50T and 100T models, reaching qualitatively similar conclusions. For brevity, we use the 103.8T parameter model in the rest of our paper. For now we ignore imperfect networking, non-ideal operating conditions, memory utilization, GPU arithmetic intensity and other such

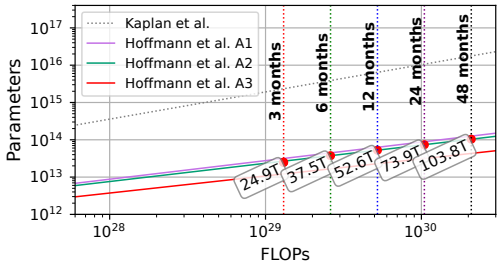


Figure 1: Loss-optimal model sizes

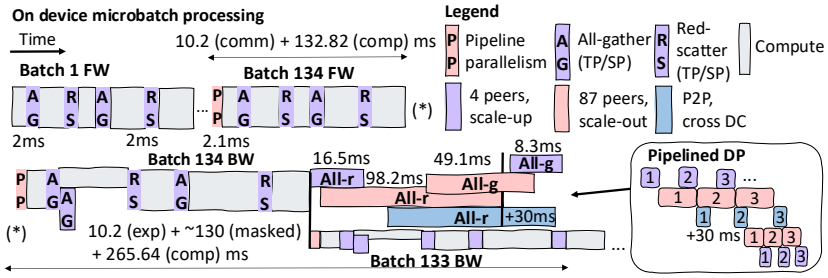


Figure 2: 3D parallelism: per device forward+backward zoom-in

factors that reduce GPU utilization and inevitably increase training time. We cover the effects of imperfect networks in §6.

3.2 The architecture of the model

For the 100T model size, we follow previous work on ideal depth-to-width ratio [34]. We also make several assumptions, as we are in uncharted territory:

Vocabulary size of 256,000 tokens. Crucially, this helps capture multi-lingual data, as well as most of the Unicode characters. Models such as *Gemma* [64] or *PaLM* [11] have popularized this size.

134 layers, 256 attention heads and 244,224 hidden size, totaling 96 trillion parameters ("100T" from now on).

Sequence length of 32k tokens. The memory size of the self-attention layer scales quadratically with context length, limiting the amount of tokens that can be processed at once. A number of techniques [10, 14, 15, 46, 75] can be used to extend the context length after pre-training, mitigating this problem.

4 Model Materialization

Until now, we've built a theoretical model, which we move to materialize in this chapter. We start from *3D parallelism* [42], as it's been shown to scale better for larger Transformer models compared to techniques such as *ZeRO-DP* [50] or *FSDP* [76]. We'll distribute the layers of the model to GPUs in such a way as to keep tensor parallelism computation local (inside the NVLink scale-up domain), while trying to minimize or overlap the communication times of pipeline parallelism and data parallelism, which happen over the scale-out 800 Gbps links.

The total aggregate memory needed to train the model, 384.14 TB, is composed of the memory for the model itself (144.04 TB for the FP8 master model, computation of forward / backward passes in FP4 precision), gradients (48.01 TB), optimizer states (192.05 TB), as well as residual memory from activations, which we detail next.

Because 1 layer cannot fit inside a single GPU, we partition the weights and activations with the scheme introduced by [33] (i.e. data, pipeline and tensor/sequence parallelism, as well as selective activation recomputation). We partition the activations across multiple devices, with no overlaps. We calculate activations in bytes, per device, as $sbh(\frac{10}{t})$, where s is the sequence length, b is the batch size and h is the hidden dimension. In order to save device memory, we offload the activations for p pipeline parallel stages [42] to host memory [50–52], and bring them back in the backward pass. We determine the tensor parallel size t by manually trying all 72 values (as there's only 72 devices per rack), and picking the smallest t that can hold the model and active activations memory, arriving at $t = 18$ (4 layers per rack, ~160 GB per device and 696 total data parallel replicas).

Importantly, we note that the layers inside a rack are all *the same layer* of different model replicas, as opposed to consecutive layers of the same model replica. We show in section 4.1 why we chose this approach.

Computation. With the above partitioning, we calculate the theoretical ideal computation time per layer using the formulas from [25, 33, 42] (we assume the backward pass is twice the forward pass), and obtain 132.82 *ms* for the forward pass (a mere 0.97% increase in computation compared to no activation recomputation).

4.1 Network Communication

With 4 layers per rack, we have 18-way tensor/sequence parallelism, 134-way pipeline parallelism, and 696-way data parallelism (348-way per DC). Fig. 2 captures the computation and overlapped communication during the forward and backward passes for 1 device holding a chunk of a layer.

Tensor and Sequence Parallel (TP/SP). As per [42], inside a layer, tensor/sequence parallelism needs to run two all-gather and two reduce-scatter operations per forward pass, as there are 4 synchronization points (the outputs of the attention layer, the feed-forward layer, and the 2 layer norms), for a tensor of size sbh. The resulting communication overhead is given by $2 \times sbh \times r = 7.82\ GB$, where r is precision (0.5 bytes for FP4), and results in 8.2 *ms* over the 1.8 TBps NVLink network (we approximate 2 all-gathers and 2 reduce-scatters as 2 all-reduce with a ring implementation). This communication cannot be overlapped with computation, as tensor/sequence parallelism introduces output-input dependencies between sub-layers (the attention, feed forward and the 2 layer norms, as mentioned above). A similar communication cost is incurred for the backward pass.

Pipeline Parallel (PP). Between racks, the communication for pipeline parallelism reduces to a point-to-point (P2P) exchange, for both forward and backward. Because of sequence parallelism [42], each device only needs to send its chunk of the input to the device with the same rank in the next layer (such a chunk has $\frac{sbh}{t} \times r = 217\ MB$). This leads to a communication time of 2.17 *ms* over the 800 Gbps scale-out network (once for forward, once for backward). Similarly to tensor parallelism, this communication is exposed, since it introduces dependencies between 2 consecutive layers.

Data Parallel (DP). After exchanging the data as described above, we need to run all-reduce operations in the backward pass to synchronize gradients between model replicas.

To keep communication time low, especially for the gradient all-reduce in the backward pass, we use a hierarchical data-parallel approach. Since we partitioned the model to have 4 similar layers of different model replicas in a single rack, we outline a 5-stage approach to pipeline this DP communication.

The first step is to all-reduce the gradients for the 4 layers inside a rack. With gradients of 358 GB per layer, this takes 16.57 *ms*. The second step is an all-reduce inside a datacenter. In order to avoid extra communication, each layer in a rack (out of the 4) only exchanges $1/4th$ of its gradients with other racks. At 89.5 GB per chunk, this takes 98.27 *ms* over the scale-out network, with 87 peers per DC. In the third step, each chunk needs to be all-reduced with its corresponding chunk from the other DC (this is just communication between pairs of 2 devices, with gradients of size $\frac{89.5\ GB}{18\ devices \times 87\ racks} = 57\ MB$ per device, with roughly 800K devices per DC), with an RTT of 60 *ms*. An all-gather in the fourth step in the scale-out network, between the 87 peers in a DC, reconstructs the gradient chunks (the $1/4th$-sized chunks per layer), with a time of 49.14 *ms*, followed by a scale-up all-gather, to finally "merge" back the chunks into the layers, with a time of 8.3 *ms*. Although this seems like a lot of time, these communication stages can be easily pipelined, as they happen over different networks. Instead of fully committing the device gradients for each of the 5 communication steps outlined above, only small batches of data can be pipelined at a time through the steps, facilitating network overlap for the 3 different networks (scale-up, scale-out and cross-DC). See figure 2 for a visual explanation.

In practice, each of the 72 devices in a rack operates mostly independently of the others. In the first step, 1 device interacts with 3 others in the same rack (as there are 4 layers). In the second step, 1 device interacts with 86 devices in 86 other racks, per DC (devices with the same rack rank), which hold the same layer replica. In the third step, there is pair-wise communication between DCs. In the fourth step, another communication between 87 peers happens, and finally in the last step 1 device interacts with 3 devices in the scale-up network, similarly to the first step.

5 Mixture of Experts

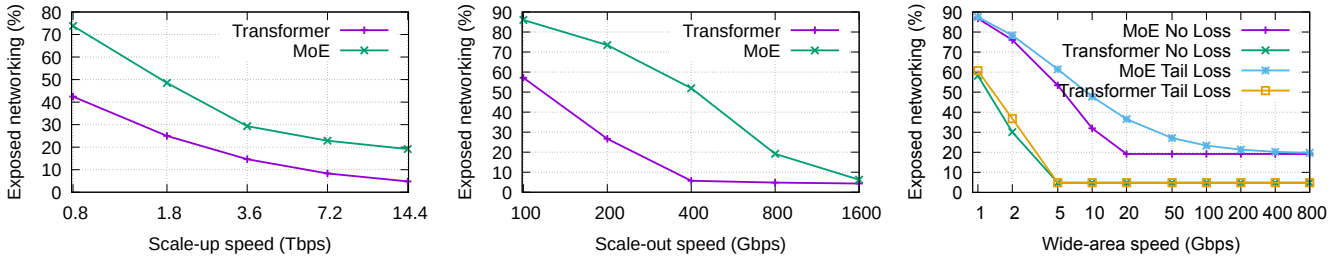
We also study scaling to 100T parameters using the Mixture of Experts architecture [17, 30, 57]. In line with previous work [30], we choose 8 experts per feed-forward layer, and $top_k = 2$ active experts to which tokens are routed per pass.

Model architecture. We choose to keep the total number of parameters similar (100T), but have 8 experts per feed-forward layer. This results in an 8x17T model, with 118 *layers*, 256 *attention heads* and 109,568 *hidden dimension*. We use the same precision, vocabulary size and context length, as well as router networks before each feed-forward layer. As the number of total parameters is similar, the memory required for the model amounts to roughly the same value (385.54 TB). We similarly keep 4 layers per rack (18 devices per layer). Due to less layers for the MoE model, we have 788 total replicas, and ~181 GB per device.

Computation and communication change from the dense Transformer. Since the forward and backward passes are routed through only 2 experts from the 8 total, the MoE model acts as a $28.4T model$ (only $28.4T$ *parameters are active* during computation). This yields a computation time of 45.18 *ms* per forward pass and twice per backward, per layer. Communication time also changes due to the lower number of layers and hidden dimension. Tensor/sequence parallelism, as well as pipeline parallelism, have to communicate inputs of size 1.75 *GB*, resulting in 3.68 *ms*, and 0.97 *ms*, respectively, per forward pass, per layer. Similarly to the dense Transformer, this cannot be overlapped. The communication pattern for data parallelism doesn't change. Although only 2 experts are routed per forward and backward pass, all experts need to have their gradients synchronized with other replicas. With a total of 408 *GB* of gradients per layer to exchange, the 5 stages detailed in chapter 4.1 last, respectively: 18.9 *ms* scale-up all-reduce, 112.23 *ms* scale-out all-reduce with 98 total peers, cross-DC pair-wise communication, 66.1 *ms* scale-out all-gather and 9.4 *ms* scale-up all-gather.

Routing network. In order to avoid extra communication of the router, we split the 8 experts on all of the 18 devices (each device gets a chunk of $1/18th$ of each of the 8 experts), in regular tensor parallel fashion. Since the router is not big, we additionally save communication in the forward pass by replicating the router on each device. There is extra communication in the backward pass for the router's gradient, but the size is very small compared to the other model parameters. This sharding strategy should also help balance computation on all devices, irrespective of expert chosen.

Implications. The backward computation time is less than the DP gradient exchange, even with an ideal network, suggesting we are now in a network-bound regime, as opposed to compute-bound, as was the case for the dense Transformer.



(a) Effect of scale-up speeds **(b) Effect of scale-out speeds** **(c) Effect of wide-area speeds**

Figure 3: Effect of network capacity on exposed communication.

6 Network topology and transport protocols

The time required to finish a step in the distributed model training includes the compute and the time waiting for the network, the so-called exposed communication time.

Our predictions show that, for each batch of training data, the GPUs will take approximately 132 *ms* to perform the calculations for the forward pass, and 264 *ms* for the backward pass for the standard model, and 45 / 90 *ms* for the Mixture of Experts model. Much of the networking is overlapped with computation, and thus hidden. The scale-out communication for pipeline parallelism, and the scale-up communication for tensor/sequence parallelism is however exposed; the scale-out all-reduce for the MoE model is also partly exposed. In our analysis we ignore pipeline effects—namely exposed communication at the start and end of each training batch—and only focus on the steady state, where all GPUs are busy (see Fig. 4 in [42]).

Our model predicts that exposed networking takes 5% for the standard transformer and 20% for MoE, per training step, with 14.4 Tbps scale-up and 800 Gbps scale-out capacity, non-blocking networks and optimal transport protocols. For MoE, the data-parallel all-reduce takes 112 *ms* of which 90 *ms* are masked. In general, training MoE models is more network intensive, and this is reflected in more exposed networking time.

How do scale-up and scale-out speeds affect exposed networking? Figure 3a shows how different scale-up speeds affect exposed communication time when scale-out speed is 800Gbps. When scale-up speeds are the same as scale-out (800Gbps), exposed time is more than 40% for the standard transformer and 75% for the MoE version. Increasing the speeds to multi-terabit dramatically reduces exposed networking to 5-20%, highlighting the importance of the scale-up network for distributed model training.

Figure 3b shows the effect of different scale-out networking speeds while keeping the scale-up speed at 14.4 Tbps per GPU. Using just 100Gbps per GPU increases exposed networking to 55-85%, mostly because of the data-parallel communication across racks. Increasing the scale-out speed beyond 400Gbps, does not reduce exposed networking time for the standard transformer but it does help MoE; using 1.6 Tbps scale-out brings MoE exposed networking to less than 5%.

Finally, figure 3c shows how the provisioned wide-area capacity, per GPU, affects exposed networking. If we have a lossless ideal transport and at least 20 Gbps of wide-area capacity per GPU, the 30 *ms* propagation delay between the east and west coast datacenters is completely masked by compute. However, if we have tail loss that inflates the tail FCT by one RTT (60ms), the networking becomes exposed, especially when provisioned capacity per GPU is lower. When retransmissions are required, especially for MoE (backward compute just 90 *ms*), wide-area networking becomes exposed unless we have very high wide-area speeds.

6.1 Building the scale-out network

How do we connect 800K GPUs with 800Gbps NICs in each of our two datacenters? The largest switches on the market today have 51.2 Tbps bisection bandwidth, and can be configured with 64 800Gbps ports. To build a fully provisioned Fat Tree [2], we need 4 tiers with a total of 87.5K switches and 2.4 million switch-to-switch links. Most of these links are tens of meters in length, which makes cheap, passive DAC cables infeasible; optical transceivers cost 1K USD even at 400 Gbps, so the estimated price tag for the wires alone is $5 billion. A common budget for networking is 10% of the datacenter cost [74], so $5 billion per datacenter; our standard Fat Tree design costs much more - we thus need cheaper alternatives, and the key is to reduce the number of tiers.

> What is the most cost-effective network for ML clusters that does not increase exposed networking time?

Notwithstanding new research, there are two approaches which can be used: increasing the effective switch radix with multiple planes [21] and reducing the number of endpoints per network with independent rails [12]. We discuss these below.

Using multiple-planes is a way to achieve higher effective switch radix with the same switching chips [21]. The same 51.2T chip can be also configured as a 128x400Gbps, 256x200Gbps or 512x100Gbps ports switch.

To have four planes, we will build switches that contain four 51.2T ASICs and have 256 front panel ports. Inside the switch, each front panel port is divided into four 200Gbps

links each of which is connected to a separate switching ASIC via backplane copper traces. The total NIC bandwidth is still 800Gbps, but it is obtained by connecting to 4 separate networks (planes) which use the same wires and switches but are otherwise independent. Switch-to-switch links follow the same pattern, where 4 200Gbps links share a wire.

With 256-port switches, three tiers are (more than) sufficient to connect all the GPUs. Multi-plane thus reduces the number of switches and switch-to-switch links by a third compared to the standard Fat Tree. Another significant benefit is that it enables better traffic locality, as we now have four times more GPUs connecting to the same TOR switch.

Using multiple rails. How should we interconnect the GPUs to the scale-out network? Current cloud network practice [59] is to connect all GPUs in the same rack to the ToR switch. However, with an ML network, these GPUs are already connected via the faster scale-up network, so the scale-out ToR would carry little local traffic. Another observation is that GPUs from different racks typically communicate in patterns, e.g. GPU 1 from rack 1 would communicate with GPU 1 from rack 2 during data-parallel training.

What if we connect only one GPU from each rack to a separate scale-out network? The resulting network would have just 11K GPUs, totaling 72 independent networks, or 72 rails (see Figure 4). Assuming 64 port switches, we would still need three tiers for this network to interconnect all GPUs. However, if we use a 4x 200 Gbps multi-plane approach, two switch tiers are sufficient to interconnect the entire datacenter.

> What are the implications of multi-rail, multi-plane networks to job scheduling (e.g. Slurm) and fault tolerance? (2) How much further can the costs of the scale-out network be reduced by using traffic locality? (3) Given a target model, what is the optimal combination of multi-plane and multi-rail topologies?

The downside of a full multi-rail approach is that GPU N from one rack can only talk to GPU N from other racks. Any communication to other GPUs in remote racks must involve the source or destination rack scale-up network. In the case of GPU failures, this creates obvious issues. It thus makes sense to reduce the number of rails to allow more freedom in task placement, as long as it does not increase the number of tiers.

Taken together, multi-rail and multi-plane can greatly reduce network costs. For instance, using 76 rails and 4 planes requires 39K switching chips and around 800K switch-to-switch cables per DC. This is a 50% reduction in the cost of the switching chips and 66% reduction in cost of links. Finally, provisioning the topology to be full bisection does not make sense since the traffic has locality, and this locality can be exploited by the cluster manager. This enables further savings.

6.2 Building the scale-up network

Current scale-up networks are either proprietary (for instance NVLink / NVSwitch for Nvidia [44], or InfinityFabric [4] for AMD) or rely on Ethernet (e.g. Intel's Gaudi [27]). A recent industry consortium was formed to standardize scale-up networking [39]. Interconnect architectures have evolved from mesh interconnects between small numbers of accelerators (e.g. NVLink, Gaudi2) to switch-based (e.g. NVSwitch for the Blackwell). In particular, the NVSwitch approach has each GPU connect to many switching chips, creating a single tier multi-plane network [1, 44].

We also note that the workload has evolved from cache-coherent communication (each GPU has access to the entire memory of the node) to using collectives instead, making the scale-up network look like a regular network.

Given the lack of standards and growing bandwidth needs, there are many open research questions in this space.

> What is the right (1) interconnect topology, (2) networking layer, (3) transport protocol, (4) congestion control and (5) API for scale-up? (6) Are in-network collective implementations useful (e.g. Nvidia Sharp) or are end-to-end approaches sufficient as models grow in size?

6.3 Transport protocols

A multi-plane, multi-rail and oversubscribed topology for scale-out is sufficient on paper, but what transports shall we use? The only choice today is to use transports based on RDMA, either IP (RoCEv2) or InfiniBand. Both are lossless and single-path, with similar performance for AIML traffic (the latter has slightly lower latency). The industry has been moving towards standardizing an Ethernet transport, to ensure there are multiple sources for networking hardware [69].

How far are existing transports from the optimal transfer times? Point-to-point traffic pattern is particularly tough, as it goes across multiple switch tiers. We simulated synchronized P2P transfers required for our 100T model using RoCEv2 in a fully-provisioned two-tier FatTree topology with 8192 hosts and 800 Gbps links. The optimal Flow completion Time (FCT) is around 1 *ms*, yet RoCEv2 takes 9 *ms* to finish, as shown in figure 5. This is due to collisions because of ECMP [3, 23, 49] and due to head of line blocking due to PFC.

Lossless networks are notoriously difficult to manage [20]; buffers are getting smaller in BDPs as switches are getting faster, meaning that PFC will trigger more often. Furthermore, congestion control for RDMA such as DCQCN [77] or HPCC [36] starts at line rate and only reacts to congestion after the 1st RTT, meaning it can still cause PFC. Indeed, the industry is moving towards best-effort operation [69].

If we use a state of the art congestion control protocol [7], best effort networks and single path, the FCT drops to 7 *ms*. This increases exposed networking from 5% / 20% with an ideal

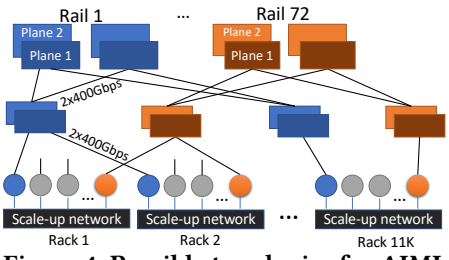


Figure 4: Possible topologies for AIML

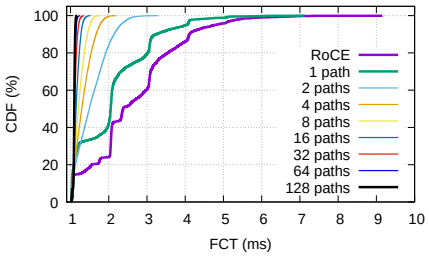


Figure 5: FCT for pipeline parallelism

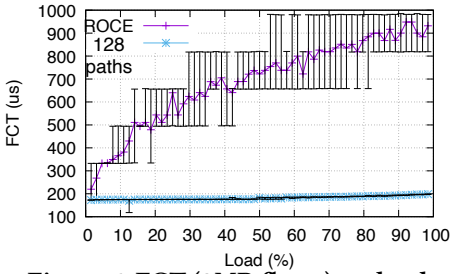


Figure 6: FCT (2MB flows) vs. load

transport to 8% / 25%. If we use transports such as NDP [23] or Homa [41] that spread traffic across all paths we get within 5% of the optimal FCT, as shown in figure 5. Note that collisions affect FCT even when the network load is smaller, and do not depend on flow size or the linkspeed; figure 6 shows the the tail FCT for 2 MB flow permutation over a 100 Gbps network when only a fraction of the nodes participate.

What is the right design for a multipath transport protocol that can load balance effectively in the presence of failures, can handle packet losses and is implementable in hardware at 800 Gbps+ speeds? What is the right split of functionality between switches and the end-hosts with regards to spraying?

Finally, the strongest argument for multipath is when multi-plane designs are used: single path transports require application changes (typically the collective library) that manually break up the data across the planes, resulting in sub-optimal load balancing. Furthermore, within a rail, we would have similar FCT inflation due to collisions as in our experiment above. Indeed, the industry is adopting multipath transmission and lossy operation as the way forward [69]. However, existing RoCE hardware implements go-back-N and requires PFC to operate well, so big changes are needed.

Wide area transport. The scale-out transport must be multipath and capable of running at 800 Gbps over small RTT links; what about the inter-DC transport? The all-reduce involves all training nodes, each transferring ~300 MB per iteration in around 100 *ms*. This requires at least 5 Gbps per flow. We can use TCP, but the CWND adaptation creates unnecessary loss; coupled with a fairly large RTT, retransmissions create unnecessarily long tail FCT which increases exposed networking (figure 3c). If we adopt a controller similar to B4 [28], we can perform both traffic engineering and admission control on the wide area links to ensure congestion control is not needed and all traffic is admittable, but this adds further synchronization issues to be solved. Even with perfect scheduling we still incur loss due to bit errors, which require retransmissions. These could be avoided by adding redundancy to the outgoing traffic. Finally, the wide-area traffic must be prioritized within data center networks, which means that multipath traffic must be able to cope with non-equal capacity links.

How should a wide-area transport be implemented to enable near-perfect operation yet be simple to implement and easy to debug? What are the interactions of wide-area traffic and sprayed traffic?

7 Summary

Significant breakthroughs are required to keep up with the ever-growing demands of next-generation LLMs. Based on public information on datacenter building plans from Microsoft and OpenAI, estimations of model sizes and industry shifts, as well as existing research and practices, we are able to glean into the challenges that building next generation AIML training datacenters will bring.

The hybrid network of scale-up and scale-out is replacing the traditional Clos cloud network, while multiple-planes and multiple-rails will further complicate communication in ML datacenters. This means that some nodes cannot directly talk to each other, and that traffic must be split across multiple planes. We (and the industry [69]) believe multipath communication is necessary to make best use of the hardware and mitigate the inflation single path protocols bring to tail flow completion times. Scheduling and placement also need to evolve to account for these new networks, and things are further complicated by failure domains for tensor parallelism being rigidly confined to the rack level. Adding wide area transport into the mix comes with its own set of unique challenges. It's of utmost importance to leave headroom for redundancy and failure recovery while keeping utilization high; for example, by mixing long running training workloads with transient inference jobs.

It's also important to note that advances in network monitoring are required to cope with multiple networks (multiple rails and planes for scale-out plus scale-up) and blazingly fast data speeds, as current sampling, single network approaches appear to no longer capture the relevant bigger picture.

Acknowledgments We thank the reviewers for their constructive feedback. This work is supported by VMware gift funding.

References

[1] Hazem A. Abdelhafez, Christopher Zimmer, Sudharshan S. Vazhkudai, and Matei Ripeanu. 2019. AHEAD: A Tool for Projecting Next-Generation Hardware Enhancements on GPU-Accelerated Systems. In *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 583–592. https://doi.org/10.1109/IPDPSW.2019.00103

[2] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. 2008. A Scalable, Commodity Data Center Network Architecture. In *Special Interest Group on Data Communication (SIGCOMM)*. ACM.

[3] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat. 2010. Hedera: Dynamic Flow Scheduling for Data Center Networks. In *Networked Systems Design and Implementation (NSDI)*. USENIX Association.

[4] AMD. 2020. AMD Infinity Fabric™ Link. Retrieved 2024-10-13 from https://www.amd.com/content/dam/amd/en/documents/instinct-tech-docs/other/56978.pdf

[5] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[6] Bloomberg. 2024. OpenAI Pitched White House on Unprecedented Data Center Buildout. Retrieved 2024-10-14 from https://www.bloomberg.com/news/articles/2024-09-24/openai-pitched-white-house-on-unprecedented-data-center-buildout

[7] Tommaso Bonato, Abdul Kabbani, Daniele De Sensi, Rong Pan, Yanfang Le, Costin Raiciu, Mark Handley, Timo Schneider, Nils Blach, Ahmad Ghalayini, Daniel Alves, Michael Papamichael, Adrian Caulfield, and Torsten Hoefler. 2024. SMaRTT-REPS: Sender-based Marked Rapidly-adapting Trimmed & Timed Transport with Recycled Entropies. *arXiv preprint arXiv:2404.01630v1* (2024). https://arxiv.org/html/2404.01630v1

[8] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*. 1877–1901.

[9] C. Dahl, R. Cui, D. Cui, C. Squire, S. Kennedy. 2022. Pathways to achieving a 2030 coal phase-out in the United States.

[10] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307* (2023).

[11] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[12] Salvador Coll, Eitan Frachtenberg, Fabrizio Petrini, Adolfy Hoisie, and Leonid Gurvits. 2002. Using Multirail Networks in High-Performance Clusters. (08 2002).

[13] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[14] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753* (2024).

[15] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[16] Dylan Patel and Daniel Nishball and Jeremie Eliahou Ontiveros. 2024. Multi-Datacenter Training: OpenAI's Ambitious Plan To Beat Google's Infrastructure. Retrieved 2024-10-14 from https://www.semianalysis.com/p/multi-datacenter-training-openais

[17] William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research* 23, 120 (2022), 1–39.

[18] Maxim Fishman, Brian Chmiel, Ron Banner, and Daniel Soudry. 2024. Scaling FP8 training to trillion-token LLMs. arXiv:2409.12517 [cs.LG] https://arxiv.org/abs/2409.12517

[19] Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752* (2023).

[20] Chuanxiong Guo, Haitao Wu, Zhong Deng, Gaurav Soni, Jianxi Ye, Jitu Padhye, and Marina Lipshteyn. 2016. RDMA over Commodity Ethernet at Scale. In *Special Interest Group on Data Communication (SIGCOMM)*. ACM.

[21] Yibo Guo, William M. Mellette, Alex C. Snoeren, and George Porter. 2022. Scaling beyond packet switch limits with multiple dataplanes. In *Proceedings of the 18th International Conference on Emerging Networking EXperiments and Technologies* (Roma, Italy) *(CoNEXT '22)*. Association for Computing Machinery, New York, NY, USA, 214–231. https://doi.org/10.1145/3555050.3569141

[22] Jigar Halani. 2024. The Evolution of Data Centers: How AI Factories Are Leading the Charge!. In *Datacenter Summit 2024*. https://dcs.greenbusinesscentre.com/dcs2024presentations/1.pdf

[23] Mark Handley, Costin Raiciu, Alexandru Agache, Andrei Voinescu, Andrew W. Moore, Gianni Antichi, and Marcin Wójcik. 2017. Re-architecting Datacenter Networks and Stacks for Low Latency and High Performance. In *Special Interest Group on Data Communication (SIGCOMM)*. ACM.

[24] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415* (2016).

[25] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, E Buchatskaya, T Cai, E Rutherford, DdL Casas, LA Hendricks, J Welbl, A Clark, et al. 2022. Training compute-optimal large language models. arXiv 2022. *arXiv preprint arXiv:2203.15556* 10 (2022).

[26] Jen-Hsun Huang. 2024. Keynote. In *GPU Technology Conference 2024*. https://www.youtube.com/watch?v=Y2F8yisiS6E

[27] Intel. 2024. Intel Gaudi 3 AI Accelerator White Paper. Retrieved 2024-10-13 from https://www.intel.com/content/www/us/en/content-details/817486/intel-gaudi-3-ai-accelerator-white-paper.html

[28] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. 2013. B4: Experience with a globally-deployed software defined WAN. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 3–14.

[29] Jarred Walton. 2024. Nvidia's next-gen AI GPU is 4X faster than Hopper: Blackwell B200 GPU delivers up to 20 petaflops of compute and other massive improvements. Retrieved 2024-10-01 from https://www.tomshardware.com/pc-components/gpus/nvidias-next-gen-ai-gpu-revealed-blackwell-b200-gpu-delivers-up-to-20-petaflops-of-compute-and-massive-improvements-over-hopper-h100

[30] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088* (2024).

[31] Ziheng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, et al. 2024. {MegaScale}: Scaling large language model training to more than 10,000 {GPUs}. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. 745–760.

[32] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361* (2020).

[33] Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Reducing activation recomputation in large transformer models. *Proceedings of Machine Learning and Systems* 5 (2023), 341–353.

[34] Yoav Levine, Noam Wies, Or Sharir, Hofit Bata, and Amnon Shashua. 2020. The depth-to-width interplay in self-attention. *arXiv preprint arXiv:2006.12467* (2020).

[35] M Lewis. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[36] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang, Zheng Cao, Ming Zhang, Frank Kelly, Mohammad Alizadeh, et al. 2019. HPCC: High precision congestion control. In *Proceedings of the ACM special interest group on data communication*. 44–58.

[37] Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[38] Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764* (2024).

[39] Mark Nossokoff and Tom Sorensen - Hyperion Research. 2024. UALink Group Formed to Develop Standard High Speed Low Latency Interconnects for Accelerated HPC and AI Systems. Retrieved 2024-10-13 from https://hyperionresearch.com/wp-content/uploads/2024/07/Hyperion-Research-Special-Analysis-UALink-Group-Created-for-Accelerator-Interconnect-July-2024.pdf

[40] Microsoft. 2024. Measuring energy and water efficiency for Microsoft datacenters - Microsoft Datacenters. https://datacenters.microsoft.com/sustainability/efficiency/ 08 October 2024.

[41] Behnam Montazeri, Yilong Li, Mohammad Alizadeh, and John Ousterhout. 2018. Homa: A Receiver-driven Low-latency Transport Protocol Using Network Priorities. In *Special Interest Group on Data Communication (SIGCOMM)*. ACM.

[42] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, et al. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–15.

[43] NVIDIA. 2024. Nvidia reveals Blackwell B200 GPU, the 'world's most powerful chip' for AI. Retrieved 2024-10-01 from https://www.nvidia.com/en-us/data-center/gb200-nvl72/

[44] Nvidia. 2024. NVLink and NVLink Switch. Retrieved 2024-10-13 from https://www.nvidia.com/en-us/data-center/nvlink/

[45] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. 2023. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048* (2023).

[46] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071* (2023).

[47] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446* (2021).

[48] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research* 21, 140 (2020), 1–67.

[49] Costin Raiciu, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. 2010. Improving Datacenter Performance and Robustness with Multipath TCP. In *Special Interest Group on Data Communication (SIGCOMM)*. ACM.

[50] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–16.

[51] Samyam Rajbhandari, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. 2021. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning. In *Proceedings of the international conference for high performance computing, networking, storage and analysis*. 1–14.

[52] Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. 2021. {Zero-offload}: Democratizing {billion-scale} model training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. 551–564.

[53] Reuters. 2024. *Microsoft, OpenAI plan $100 billion data-center project, media report says.* Retrieved 2024-10-07 from https://www.reuters.com/technology/microsoft-openai-planning-100-billion-data-center-project-information-reports-2024-03-29/

[54] Ryan Smith. 2024. NVIDIA Blackwell Architecture and B200/B100 Accelerators Announced: Going Bigger With Smaller Data. Retrieved 2024-10-01 from https://www.anandtech.com/show/21310/nvidia-blackwell-architecture-and-b200b100-accelerators-announced-going-bigger-with-smaller-data

[55] Maximilian Schreiner. 2024. *GPT-4 architecture, datasets, costs and more leaked.* Retrieved 2024-10-07 from https://the-decoder.com/gpt-4-architecture-datasets-costs-and-more-leaked/

[56] Sean Hollister. 2024. Nvidia reveals Blackwell B200 GPU, the 'world's most powerful chip' for AI. Retrieved 2024-10-01 from https://www.tomshardware.com/pc-components/gpus/nvidias-next-gen-ai-gpu-revealed-blackwell-b200-gpu-delivers-up-to-20-petaflops-of-compute-and-massive-improvements-over-hopper-h100

[57] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).

[58] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053* (2019).

[59] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat. 2015. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (London, United Kingdom) *(SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 183–197. https://doi.org/10.1145/2785956.2787508

[60] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990* (2022).

[61] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing* 568 (2024), 127063.

[62] Xiao Sun, Naigang Wang, Chia-Yu Chen, Jiamin Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Viji Srinivasan, and Kailash Gopalakrishnan. 2020. Ultra-low precision 4-bit training of deep neural networks. *Advances in Neural Information Processing Systems* 33 (2020), 1796–1807.

[63] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. 2023. Retentive network: A

successor to transformer for large language models. *arXiv preprint arXiv:2307.08621* (2023).

[64] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295* (2024).

[65] Ajay Tirumala and Raymond Wong. 2024. NVIDIA Blackwell Platform: Advancing Generative AI and Accelerated Computing. In *2024 IEEE Hot Chips 36 Symposium (HCS)*. 1–33. https://doi.org/10.1109/HCS61935.2024.10665247

[66] Tom's Hardware. 2024. China makes AI breakthrough, reportedly trains generative AI model across multiple data centers and GPU architectures. Retrieved 2024-10-14 from https://www.tomshardware.com/tech-industry/artificial-intelligence/china-makes-ai-breakthrough-reportedly-trains-generative-ai-model-across-multiple-data-centers-and-gpu-architectures

[67] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[68] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[69] Ultra Ethernet Consortium. 2023. Overview of and Motivation for the Forthcoming Ultra Ethernet Consortium Specification. Retrieved 2024-10-14 from https://ultraethernet.org/wp-content/uploads/sites/20/2023/10/23.07.12-UEC-1.0-Overview-FINAL-WITH-LOGO.pdf

[70] U.S. Energy Information Administration (EIA). 2023. U.S. summer nuclear outages rose in 2023, returning to 2021 level. https://www.eia.gov/todayinenergy/detail.php?id=60682# 08 October 2024.

[71] U.S. Energy Information Administration (EIA). 2024. U.S. Electricity Overview (U.S. Lower 48). https://www.eia.gov/electricity/gridmonitor/dashboard/electric_overview/US48/US48 08 October 2024.

[72] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[73] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453* (2023).

[74] Weissberger, Alan. 2024. AI adoption to accelerate growth in the $215 billion Data Center market. https://techblog.comsoc.org/2024/09/15/bofa-ai-adoption-to-accelerate-growth-in-the-215-billion-data-center-market/

[75] Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024. Soaring from 4K to 400K: Extending LLM's Context with Activation Beacon. *arXiv preprint arXiv:2401.03462* (2024).

[76] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277* (2023).

[77] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron, Jitendra Padhye, Shachar Raindel, Mohamad Haj Yahia, and Ming Zhang. 2015. Congestion Control for Large-Scale RDMA Deployments. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (London, United Kingdom) *(SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 523–536. https://doi.org/10.1145/2785956.2787484