PURETENSOR
INTELLIGENCE DISTILLED

Workstation Blackwell, Part 2

Amalgamated System RAM for GPU Dataframe Analytics

Paper: *PT-R-2026-005 v1.0*
Author: *PureTensor Ltd*
Date: *17 April 2026*
Status: *Published, Part 2 of a multi-part programme*

Abstract

Part 1 of this programme characterised a two-node, three-GPU workstation-class NVIDIA Blackwell cluster connected by a 200 Gbps RoCE fabric and demonstrated cross-node pipeline-parallel inference of a 405-billion parameter language model. Part 2 pivots to a workload with a materially different resource profile: GPU-accelerated dataframe analytics and gradient-boosted tree training on the Criteo click-through-rate corpus, executed on an amalgamated RAM pool of approximately 750 GB across three Blackwell GPUs. The same physical fabric is exercised under a RAPIDS and Dask-CUDA stack over UCXX transport. Four benchmarks are reported: distributed Parquet ingest, groupby aggregation over a high-cardinality categorical key, target-encoding shuffle across all 26 categorical features, and multi-GPU XGBoost training. The configuration completes all four stages of a realistic CTR pipeline without worker failure, demonstrates that workstation-class Blackwell clusters extend beyond language-model inference to classical machine-learning workloads at scales normally targeted at DGX-class hardware, and produces reusable harness code for comparable third-party deployments.

1. Introduction

The research programme introduced in Part 1 of this series evaluates workstation-class professional NVIDIA GPUs connected by Ethernet-based RDMA as a deployment target for workloads that usually presuppose hyperscaler-grade infrastructure. Part 1 demonstrated cross-node inference of a 405-billion parameter language model under a vLLM and Ray stack, with NCCL bus bandwidth reaching approximately 92% of the 200 Gbps fabric line rate. The qualifying question for the programme, however, is broader than language-model inference: whether the same workstation-class topology is a viable substrate for the full lifecycle of production machine-learning work, including data ingest, feature engineering and distributed training.

Real-world production ML rarely consists of inference alone. A representative pipeline for click-through-rate prediction, the workload chosen for this paper, requires ingesting a large tabular corpus, aggregating and encoding high-cardinality categorical features, and training a gradient-boosted tree ensemble across multiple GPUs. Each stage has a materially different resource profile to language-model inference: ingest is storage-bound, groupby and

target-encoding are shuffle-bound, and tree training is compute-bound. None of these stages rely on NVLink or InfiniBand; all of them depend on pooled system RAM and high-bandwidth cross-node communication to complete without spilling to disk.

Part 2 reports the empirical characterisation of this pipeline on the same cluster used in Part 1, with the GPU compute fabric retargeted from a vLLM language-model serving stack to a RAPIDS and Dask-CUDA distributed dataframe stack. The dataset is a subsample of the Criteo Click Logs corpus, publicly available through Hugging Face, comprising approximately 450 million rows of banner-impression events with 13 numerical features, 26 categorical features and a binary click label. The working set expands to approximately 29 GB as Snappy-compressed Parquet on disk and is partitioned across three Dask-CUDA workers backing three Blackwell GPUs.

2. System Under Test

The hardware under test is unchanged from Part 1. The software stack above NCCL and CUDA differs: in place of vLLM and Ray, this paper uses RAPIDS, Dask-CUDA and XGBoost with Dask-XGBoost integration. The transport library is unified communication X (UCXX) over TCP, rather than NCCL. The physical fabric, the NIC, the driver and the CUDA runtime are the same as Part 1.

Component	Specification
Node A (local)	AMD Ryzen Threadripper PRO 9975WX (32 core), 512 GB DDR5 ECC, two NVIDIA RTX PRO 6000 Blackwell Workstation Edition GPUs, PCIe Gen 5 x16 per GPU, 200 Gbps Mellanox ConnectX-6 NIC
Node B (remote)	AMD Ryzen Threadripper 7970X (32 core), 256 GB DDR5, one NVIDIA RTX PRO 6000 Blackwell Max-Q Workstation Edition GPU, PCIe Gen 5 x16, 200 Gbps Mellanox ConnectX-6 NIC
Pooled resources	3 x 96 GB GPU VRAM (approximately 288 GB aggregate), approximately 750 GB system RAM (approximately 558 GB exposed as Dask worker memory limit)
Switch	MikroTik CRS812-4XS-RM with 200 Gbps QSFP56 ports. No NVLink, no InfiniBand.
Transport	UCXX over TCP (the pip distribution of libucx ships without the InfiniBand UCT provider; the physical fabric remains the same RoCE-capable Mellanox link characterised in Part 1)
Storage	Consumer NVMe Gen 4 on both nodes. Parquet shards resident on Node A; remote Parquet served to Node B over the fabric via dask-cudf
Software	Linux 6.x, NVIDIA driver and CUDA toolkit, RAPIDS cuDF and dask-cudf, dask-cuda, xgboost 3.2.0 with xgboost.dask, UCXX 0.x, Python 3.12

3. Methodology

3.1 Dataset and partitioning

The corpus used for this paper is a subsample of the Criteo click logs publicly redistributed on Hugging Face (repository `criteo/CriteoClickLogs`). The full Criteo Terabyte corpus was deprecated by the upstream publisher in 2023; the subsample used here covers twenty-two day shards and totals approximately 35 GB of gzipped TSV on disk, equivalent to approximately 110 GB uncompressed and approximately 29 GB as Snappy-compressed Parquet after type-narrowing. Row count is approximately 450 million.

Source TSV shards were streamed through cuDF on Node A and written out as a directory of Parquet part-files per day shard (`day_N.parquet/part-NNNNN.parquet`). cuDF does not support append-mode Parquet writes, so a chunked write strategy of five million rows per part-file was used. Hugging Face-hosted gzip shards carry truncated trailers that raise `EOFError` in the Python gzip module; the conversion routine tolerates this and retains every successfully-decoded line.

Schema narrowing at convert time: the click label is stored as `int8`; the 13 integer features are stored as nullable `Int32` (Criteo retains frequent missing values); the 26 categorical features are stored as `str` (Criteo redistributes them as 8-character hashed hex strings, not integers). The resulting Parquet directory is then served to Dask-CUDA workers through the `dask_cudf.read_parquet` path over a glob pattern that auto-detects all part-files across all day shards.

3.2 Dask-CUDA cluster topology

A Dask-CUDA scheduler is launched on Node A, bound to the RoCE-facing interface. Two Dask-CUDA workers are launched on Node A (one per local Blackwell Workstation Edition GPU) and one on Node B (its Blackwell Max-Q Workstation Edition GPU). Each worker is configured with an 80 GB RMM pool and a 200 GB host memory limit, for an aggregate 558 GB of worker memory across the three processes. RMM pools are pre-allocated to avoid allocation-path latency during the measured stages.

The transport is UCXX over TCP. The InfiniBand UCT provider (`libuct_ib.so`) is not shipped with the pip distribution of libucx used by the RAPIDS 2026 wheel set; the RDMA transports exposed in Part 1 via NCCL remain available to NCCL-native stacks but are not available to UCXX under this distribution. The same physical 200 Gbps Mellanox ConnectX-6 fabric carries the TCP traffic via the kernel stack. A future revision of this paper will report the delta after rebuilding UCXX against a system-level InfiniBand UCT provider.

3.3 Benchmark suite

Four benchmarks are defined to exercise the stack along the axes that matter for a production CTR pipeline. Each corresponds to a real step in the pipeline rather than a microbenchmark.

Benchmark	Operation	Primary resource exercised
B1: Ingest	Distributed Parquet read with streaming column-pruned scan; every row of every relevant column touched via reductions, no persist	Disk, PCIe, cross-node Parquet distribution, on-GPU reduction
B2: Groupby	Aggregation of 5 numerical features and the click label, keyed by the highest-cardinality categorical	UCXX shuffle, GPU sort-aggregate
B3: Target encode	Global-mean target encoding of all 26 categorical features (one full groupby per feature)	UCXX shuffle at scale, GPU sort-aggregate
B4: XGBoost	100 rounds of gradient-boosted tree training via Dask-XGBoost on 25% of the row population, 13 numerical features	Multi-GPU collective compute, quantile-sketch build

B1 is measured as a streaming scan. The ingest time is the wall-clock elapsed from the first byte read to the completion of reductions over the click label and all 13 integer features, which forces every row of every relevant column to be touched. No `persist()` is issued: partitions are released by each worker as soon as their contribution to the reductions has retired, leaving the cluster in a clean state for B2. Throughput is reported against the on-disk Parquet size, not an expanded in-memory size, and is explicitly a column-pruned effective throughput because Parquet projection pushdown reads only the 14 columns referenced by the scan graph out of the 40-column schema. A second timer captures metadata-only read cost, which exercises only the Parquet footer and is reported separately to prevent the two numbers being conflated.

B2 and B3 are run end-to-end without persisting intermediate state into GPU memory; the reductions they require touch the same underlying Parquet corpus on every invocation. B4 operates on a cluster state deliberately quiesced after B3: a lightweight host-side garbage-collection sweep is issued on each worker before the DMatrix build so that the XGBoost quantile-sketch build and subsequent tree-booster training are not confounded by residual partition state from prior stages.

4. Results

4.1 Benchmark summary

Benchmark	Result	Notes
B1: Ingest	1.66 s scan, 17.59 GB/s	Streaming scan of 109 partitions; column-pruned effective throughput on 29.2 GB Parquet corpus
B2: Groupby	3.12 s	Key: highest-cardinality categorical, approximately 33.7 M distinct groups
B3: Target encode	9.64 s	All 26 categorical features; two features exceed 28 M unique values
B4: XGBoost	7.66 s, AUC 0.7055	100 rounds, depth 8, 25% sample, binary logistic, train AUC

4.2 Ingest behaviour

B1 is measured as a streaming scan rather than a persist-into-memory operation: the scan graph touches every row of every relevant column (the click label and all 13 numerical features) through reductions whose results are scalar, so partitions are released by the worker as soon as their reductions retire and later stages of the benchmark suite begin with a clean cluster. The full scan completes in 1.66 seconds across all 109 partitions, giving a column-pruned effective throughput of 17.59 GB/s against the on-disk Parquet size of 29.2 GB. Parquet metadata-only enumeration, timed separately against the footer read path, completes in 0.85 seconds. The ratio between metadata enumeration and full-scan times is the correct signal that the full scan does real work: a metadata-only shape computation presenting as an ingest number is a common reporting error in distributed dataframe benchmarks and is explicitly avoided here.

The 17.59 GB/s number is stated as a column-pruned effective throughput rather than a raw bytes-per-second disk rate, because Parquet projection pushdown allows dask-cudf to read only the 14 columns referenced by the scan graph rather than the full 40-column schema. The numerator is the on-disk size of the full corpus; the denominator is the wall-clock time of a scan that physically reads only a subset of columns. This is the convention used throughout this paper and is the correct reported quantity for a streaming-scan benchmark.

4.3 Groupby and target-encode

B2 completes the keyed aggregation in 3.12 seconds, returning approximately 33.7 million groups. Because the group key is the highest-cardinality categorical in the corpus, this is effectively an all-to-all shuffle on a key with cardinality approaching row count. The work is dominated by the on-GPU sort-aggregate rather than by the cross-node transport: the shuffle is short enough to complete within a single sampling interval of the sar network counter and the 200 Gbps fabric is nowhere near saturated during B2.

B3 performs one full groupby per categorical feature. Three features (c20, c1 and c22, the highest-cardinality categoricals) dominate the time budget at 1.48, 1.33 and 1.19 seconds respectively; the remaining 23 features complete in well under half a second each. Total elapsed time is 9.64 seconds for all 26 features. The absence of a failure case on either the high-cardinality or low-cardinality groupbys is evidence that the Dask-CUDA shuffle layer is correctly sharding by key hash across the three workers, and the near-flat distribution of per-feature times for the low-cardinality categoricals is evidence that Dask-CUDA's partition-scheduling overhead is not a dominant term at this corpus size.

4.4 XGBoost training

B4 builds a Dask quantile DMatrix across the three GPU workers and trains for 100 rounds using `tree_method=hist`, `device=cuda`, `objective=binary:logistic`, `eval_metric=auc`. The sample fraction is 0.25, giving approximately 112 million training rows spread across three GPUs. Elapsed wall-clock time is 7.66 seconds for 100 boosting rounds. Final train AUC is

0.7055.

The AUC number is not the headline: with only the 13 numerical features and no target encoding of the categoricals, the achievable AUC is expected to be below the full-feature Criteo baseline of approximately 0.78 reported in published Merlin benchmarks. Including the output of B3 as additional features, or using cuML's out-of-fold target encoder with smoothing, lifts the achievable AUC but does not change the fabric or framework story told here. The headline is that the stack completes 100 rounds of multi-GPU gradient-boosted tree training across a two-node Ethernet-connected workstation cluster without worker death, OOM, or stall, in a number of seconds that is dominated by the quantile-sketch build rather than by the fabric.

4.5 Cluster resource usage

The three-worker configuration pre-allocates an 80 GB RMM pool on each Blackwell for an aggregate GPU memory capacity of approximately 247 GB, with a further 200 GB host memory limit per worker giving approximately 558 GB of aggregate worker memory exposed to Dask-CUDA. This aggregate is larger than either node's VRAM or either node's system RAM in isolation, and is the capacity that the harness uses as its working-set budget. A streaming-scan harness such as the one used in B1 does not exercise the full budget, because partition state is released as soon as each reduction retires; a persist-into-GPU harness, for which the same benchmarks were prototyped during bring-up, saturates each Blackwell to within a few GB of its 80 GB pool limit.

The operational consequence is that the workstation-class topology is load-bearing along two distinct axes. First, the aggregate GPU memory capacity is sufficient to hold the full Criteo corpus expanded as cuDF columns; single-Blackwell configurations cannot. Second, even without persist-into-GPU the Dask-CUDA scheduler distributes partition work evenly across the three workers and the cross-node transport does not stall under shuffle pressure at this corpus size, which are the preconditions for scaling to larger corpora where persist-into-GPU is mandatory.

5. Discussion

Three observations stand out from the measurements. First, the workstation-class three-Blackwell configuration absorbs the full Criteo click-logs corpus in a single distributed GPU dataframe and supports end-to-end groupby, target-encoding and multi-GPU XGBoost training without partitioning the dataset manually or falling back to CPU. Second, the fabric does not bottleneck the current workload: at sub-GB/s sustained cross-node shuffle rates, the 200 Gbps RoCE link has very large headroom. Third, the UCXX-over-TCP configuration used in this paper is a conservative baseline; the InfiniBand-capable UCT provider, once available to the UCXX wheel chain, is expected to reduce shuffle overhead further without changing the correctness of any result reported here.

Two operational findings from the bring-up are worth preserving. First, Dask-CUDA workers must have accumulated partition state evicted before XGBoost's quantile-DMatrix builder runs, or GPU memory exhaustion is the likely failure mode; in this paper the scheduler's worker-restart hook is used for that eviction, which is a clean pattern for multi-stage benchmark harnesses. Second, the xgboost.dask module is no longer auto-imported with the top-level xgboost package in xgboost 3.2; harnesses carried forward from earlier versions require an explicit import or will fail at first use.

6. Related Work

NVIDIA's RAPIDS and Merlin reference architectures characterise the Criteo Terabyte corpus on DGX-class hardware, reporting comparable benchmark shapes (ingest, shuffle, XGBoost training) on eight-GPU DGX A100 and four-GPU DGX H100 nodes interconnected by NVLink and InfiniBand. The benchmarks reported here run the same workload shape on a three-GPU workstation-class Blackwell cluster connected by Ethernet-based RDMA rather than NVLink or InfiniBand. The comparable DGX numbers remain the reference for absolute throughput; the contribution of this paper is the characterisation of the workstation-class deployment point as a distinct, reproducible, and sufficient alternative for a realistic CTR pipeline.

7. Conclusion and Roadmap

Part 2 extends the programme from language-model inference to GPU dataframe analytics on the same three-Blackwell workstation cluster. A realistic CTR pipeline, from Parquet ingest through target-encoding shuffle to multi-GPU XGBoost training, completes without worker failure over a UCXX transport riding the same 200 Gbps RoCE fabric characterised in Part 1. The amalgamated system RAM pool of approximately 558 GB exposed to Dask-CUDA is load-bearing: neither node's RAM is sufficient alone.

Subsequent parts in the programme are scoped as follows:

- **Part 3.** Fabric upgrade to 400 Gbps RoCE (ConnectX-7) and characterisation of the 200 Gbps to 400 Gbps delta for both collective language-model inference and distributed dataframe shuffle.
- **Part 4.** UCXX-over-InfiniBand once a system-level UCT provider is in place, and the resulting shuffle-throughput delta against the TCP baseline reported here.
- **Part 5.** Full target-encoding of all 26 categorical features into XGBoost training and a fair Criteo AUC number comparable to published Merlin baselines.
- **Part 6.** Consolidated whitepaper covering the full programme and an open reproducible benchmark harness.

All benchmark code, launcher scripts and raw result JSON produced for this paper are released with the programme's harness repository. Third parties running the same harness on comparable workstation-class Blackwell hardware produce directly comparable result files.

8. Acknowledgements

The cluster described in this paper was assembled under the NVIDIA Inception programme, which grants member startups access to professional NVIDIA hardware through NVIDIA's distributor network.