Deploying Clinical Language and Vision-Language Models Where the Data Lives: An Energy-Aware, Cross-Vendor Protocol for On-Premises Healthcare AI

Manpreet Singh
Embedded LLM , Singapore
manpreet.singh@embeddedllm.com
COLM 2026 Workshop on Deploying AI in Healthcare (DAIH)

Abstract

Language and vision-language foundation models are rapidly entering clinical workflows: pathology vision-language encoders analyze whole-slide images, protein and genomic language models support precision oncology, and sequence models continuously monitor intensive-care streams. While benchmark leaderboards emphasize accuracy and AUC, deploying health systems face a different set of questions. Which accelerator will the model run on? How much energy does each inference consume on power-constrained on-premises infrastructure? Does protected health information (PHI) remain within institutional boundaries? And will results measured on one GPU vendor reproduce on the hardware a hospital actually deploys?

We argue that *deployment is itself an evaluation problem* and present an evidence-consistent protocol for measuring it. Our approach uses a single fused-kernel substrate, authored once in OpenAI Triton and executed unmodified across NVIDIA and AMD GPUs, while keeping model weights, numerics, and algorithms fixed. This isolates hardware and implementation effects, enabling consistent deployment evaluation across the accelerators institutions own.

Using whole-slide pathology encoders (UNI and Virchow) as primary case studies, and extending to protein and genomic language models and real-time clinical time-series models, the protocol reports deployment-centric metrics rather than predictive performance alone. Across workloads, we observe approximately $2\times$ lower on-premises encoding energy per slide at fixed diagnostic quality (mean $|\Delta\text{AUC}| \leq 0.001$), fidelity-gated speedups ranging from $1.51\times$ to $43.1\times$ at cosine similarity ≥ 0.99999, and an end-to-end latency reduction from 805 ms to 23 ms, enabling sub-50 ms bedside inference. The protocol also exposes a governance-relevant reproducibility hazard that conventional accuracy benchmarks overlook: a latent host-device round-trip in an upstream reference implementation changes one model's measured speedup from $4.22\times$ on NVIDIA to $38.8\times$ on AMD, showing that single-vendor benchmarking can hide implementation artifacts that surface very differently on the hardware a clinical site actually deploys.

These findings suggest that responsible deployment of clinical LLMs and VLMs requires evaluating *how* and *where* models are served, not merely how accurately they predict.

1 Introduction

Large language and vision-language models are entering clinical practice (Singhal et al., 2023; Moor et al., 2023; Thirunavukarasu et al., 2023). In computational pathology a frozen vision-language encoder such as UNI or Virchow (Chen et al., 2024; Vorontsov et al., 2024; Lu et al., 2024) turns a gigapixel whole-slide image into tile embeddings that drive downstream diagnosis; protein and genomic language models (Lin et al., 2023; Elnaggar et al., 2021;

Avsec et al., 2026) inform precision-oncology pipelines; and sequence models monitor intensive-care streams (Gu & Dao, 2024). The community has invested heavily in measuring how *good* these models are: classification accuracy, detection and segmentation scores, AUC on held-out cohorts. What it measures far less is whether a model that scores well can actually be *deployed* in the setting where the data lives, which for high-stakes clinical AI is on-premises, on whatever accelerator the institution owns, under real energy, privacy, and consistency constraints.

Why benchmark conditions do not match clinical reality. Most reported clinical AI results are obtained under a common benchmark setting: dense, large-batch transformer inference on NVIDIA GPUs, reflecting the hardware and workloads for which general AI infrastructure is optimized. Real clinical deployments rarely match these conditions. A single whole-slide image produces 10^4 to 10^5 tiles that must be processed by a vision encoder, while both raw slides and intermediate embeddings contain protected health information (PHI). In practice, pathology laboratories typically operate one or a few mid-range GPUs rather than large cloud clusters (Hanna et al., 2022). The same mismatch extends to the other clinical foundation models considered in this work, including protein and genomic language models (Lin et al., 2023; Elnaggar et al., 2021; Dauparas et al., 2022; Avsec et al., 2021; 2026; Liu & Paliwal, 2024) and state-space models for intensive-care monitoring (Gu & Dao, 2024; Gu et al., 2022). Their reference implementations were primarily developed and optimized for NVIDIA GPUs, yet they are increasingly deployed on AMD MI300X accelerators because of their 192 GB unified HBM3 memory. As a result, accuracy benchmarks obtained under the assumed experimental setup provide little insight into the deployment questions hospitals must answer first.

This paper: deployment as a benchmarking problem. We take the position that deployment is not a downstream engineering detail but a property an evaluation protocol should diagnose directly, alongside robustness and evidence consistency. We propose a protocol that measures, for a fixed model, the quantities that govern whether a clinical LLM/VLM can be served on-site: throughput and energy per inference, on-site retention of PHI, and cross-vendor reproducibility of the reported result. The instrument that makes this measurable is a single fused-kernel substrate, authored once in OpenAI Triton (Tillet et al., 2019) and compiled unmodified to NVIDIA (PTX) and AMD (ROCm). Because the same source runs on both vendors with weights and numerics held fixed, we vary only hardware and implementation and read off deployment-facing differences a single-vendor accuracy benchmark cannot expose. We anchor on whole-slide pathology, the most vision-native and most PHI-sensitive of these workloads, and show the protocol generalizes to the protein, genomic, and clinical time-series settings.

Contributions.

1. **A deployment-oriented evaluation protocol** (Sec. 2): we define the metrics that an evidence-consistent deployment benchmark should report, including fidelity-gated speedup, energy per inference, on-site PHI retention, and cross-vendor reproducibility, together with a controlled evaluation framework that holds the model fixed while varying only hardware and implementation.
2. **An on-premises whole-slide pathology benchmark** (Sec. 3): for UNI and Virchow, we reduce on-premises encoding energy per slide by approximately $2\times$ while preserving diagnostic quality (mean $|\Delta\text{AUC}| \leq 0.001$ across four tasks), keeping PHI on site, reducing peak memory to fit within a 24 GB GPU, and obtaining consistent results on AMD using the same source code.
3. **A vendor-portable benchmark across additional clinical models** (Sec. 4): the same Triton substrate accelerates six protein and genomic foundation models by $1.51\times$ to $43.1\times$ at cosine similarity ≥ 0.99999 on NVIDIA L4, H100, and AMD MI300X using a single source implementation.
4. **A measured cross-vendor reproducibility hazard** (Sec. 5): identical kernel implementations achieve $38.8\times$ speedup on AMD but $4.22\times$ on NVIDIA because a

latent host-device round-trip is hidden on one platform and exposed on the other, illustrating a governance-relevant limitation of single-vendor evaluation.

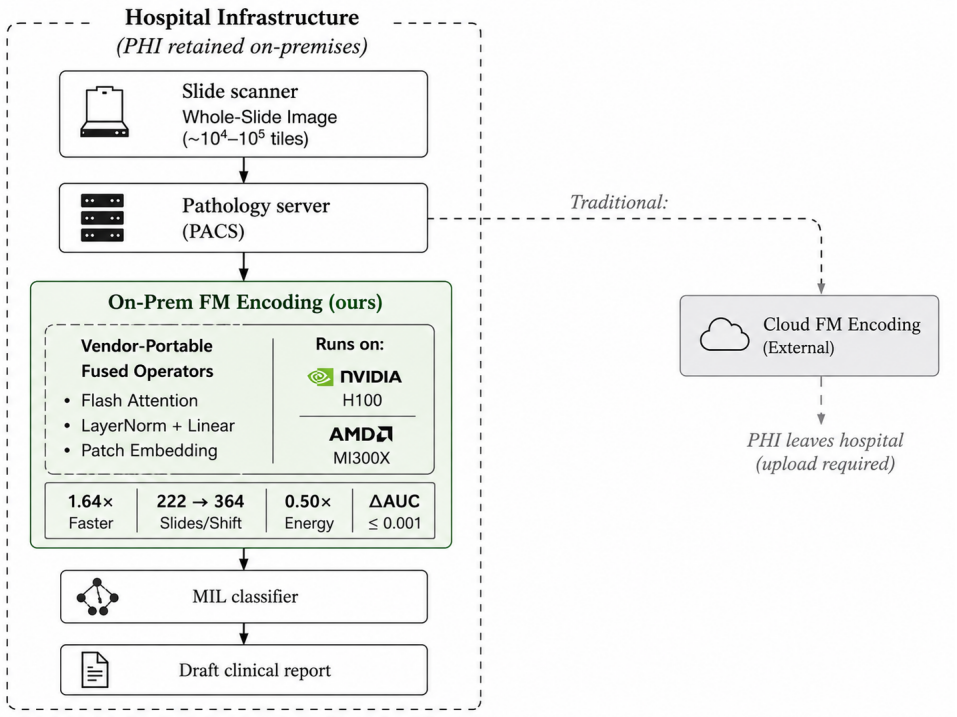

Figure 1: On-premises whole-slide pathology deployment. The slide scanner and pathology server keep PHI inside the hospital; vendor-portable fused operators encode the slide on the institution's own NVIDIA or AMD card before a downstream MIL classifier drafts a report. The traditional path (right) ships PHI to an external cloud encoder. The protocol scores the on-premises path on speed, slides per shift, energy, and a diagnostic-quality gate ($|\Delta\text{AUC}| \leq 0.001$).

2 A Deployment-Oriented Evaluation Protocol

We first state what a deployment benchmark should measure, give a plain-language account of the measurement instrument, and explain why a portable substrate is the right one.

What to measure. Accuracy and AUC answer whether a model is good, not whether it can be served on-site. A deployment benchmark should add four metrics, each mapped to a DAIH question. *Fidelity-gated speedup* (**does it still work?**): throughput improvement admissible only when outputs match a reference to a stated numerical gate, so speed is never bought with silent quality loss. *Energy per inference* (**is it equitable?**): on-premises clinical hardware is power and cooling constrained, and energy is a first-class cost (Strubell et al., 2019; Lacoste et al., 2019). *On-site PHI retention* (**is it private?**): whether the raw slide, the encoder, and its intermediate embeddings ever leave the building. We do not claim cloud storage of structured EHR text is unsafe in general, and many institutions already run EHRs under a cloud business-associate agreement; the constraint we measure is narrower and specific to the workloads in this paper, where a gigapixel image or a raw genomic sequence is shipped whole to an external encoder rather than staying on infrastructure the hospital controls, which is a materially different exposure than a covered cloud EHR deployment. *Cross-vendor reproducibility* (**is it safe across sites?**): whether a reported result holds when the same model runs on a different institution's accelerator. Accuracy alone answers none of these.

What the measurement instrument does (a primer). A modern vision or language encoder executes on a GPU as a sequence of small operations, including patch or token projection, attention, normalization, and MLPs. This section, and the fusion discussion throughout Secs. 3 and 4, concerns three tiers of memory that we name once here to keep them distinct: (i) on-die SRAM (registers and shared memory, private to a streaming multiprocessor), (ii) on-package HBM (the GPU's own global/device memory, off-die but still on the accelerator), and (iii) off-package host RAM (CPU-side DRAM, reached only over PCIe). Fusion, discussed next, moves traffic between tiers (i) and (ii) and never leaves the GPU; the host-device round-trip discussed in Sec. 5 is a distinct phenomenon that moves data between tiers (ii) and (iii).

Absent fusion, each operation reads its inputs from HBM and writes its outputs back to HBM before the next operation begins. For the clinical models considered here, this HBM traffic, rather than arithmetic, dominates execution time because the tensors are large while the computation performed by each operation is relatively modest.

Kernel fusion combines multiple consecutive operations into a single kernel that keeps intermediate values in on-die SRAM and writes to HBM only once, reducing both latency and energy while preserving the same numerical result. We implement these fused kernels in OpenAI Triton (Tillet et al., 2019), where a single source file compiles to both NVIDIA (PTX) and AMD (ROCm) without vendor-specific rewrites. Model weights and numerical computations remain unchanged, making the fused kernels drop-in replacements that produce identical embeddings more efficiently. The substrate therefore changes *how* the model is computed without changing *what* it computes, and it never involves tier (iii): all fusion in this paper stays GPU-resident.

Why a portable substrate is the instrument. Measuring the four metrics fairly requires holding the model constant while changing only hardware and implementation. Standard PyTorch models already execute on AMD through ROCm/HIP with little or no code change: matmul, LayerNorm, softmax, and SDPA-based attention all dispatch to ROCm kernels transparently, so the realistic AMD fallback for these workloads is the unfused ROCm GPU path, not a CPU path. The gap this protocol targets is narrower and specific to the *hand-fused* operators that dominate clinical-model latency (Sec. 3–5): those have historically been authored and tuned against CUDA alone, so a hospital that buys AMD cannot benchmark the fused, deployment-grade path without a vendor-specific rewrite. A single Triton source compiled unmodified to both vendors removes that confound at the level where it actually applies: weights, numerics, and downstream heads are unchanged, so any measured difference in the fused path between vendors is a property of the deployment stack rather than of the model. This turns portability of the fusion layer, not of the baseline model, into a measurement instrument.

Why the workloads stress the instrument. Profiling the reference PyTorch implementations on both vendor stacks shows three recurring properties that make LLM-tuned infrastructure underserve clinical foundation models. Whole-slide pathology encoders run a vision transformer over 10^4 to 10^5 tiles per slide, so per-tile attention and normalization dominate. Protein and genomic models tokenize over 4 to 20 symbols, yet stock embedding kernels are matrix multiplications tuned for 50K-plus LLM vocabularies, wasting forward-pass time on trivial lookups. Long-context genomic models accept up to 196K bases at batch one, so attention is bandwidth-bound and standard FlashAttention tiles do not fit (Dao et al., 2022; Dao, 2023); binding models interleave an energy computation between transformer passes; and clinical time-series pipelines spend most of their time in irregular-sampling interpolation. A block-structured kernel language that fuses these paths and compiles to both vendors lets the protocol measure all of them under one controlled instrument.

3 On-Premises Whole-Slide Pathology: An Energy and PHI Benchmark

We open with the most vision-native deployment problem, where portability, energy, and PHI retention are the binding constraints. A single gigapixel slide yields 10^4 to 10^5 tiles, and

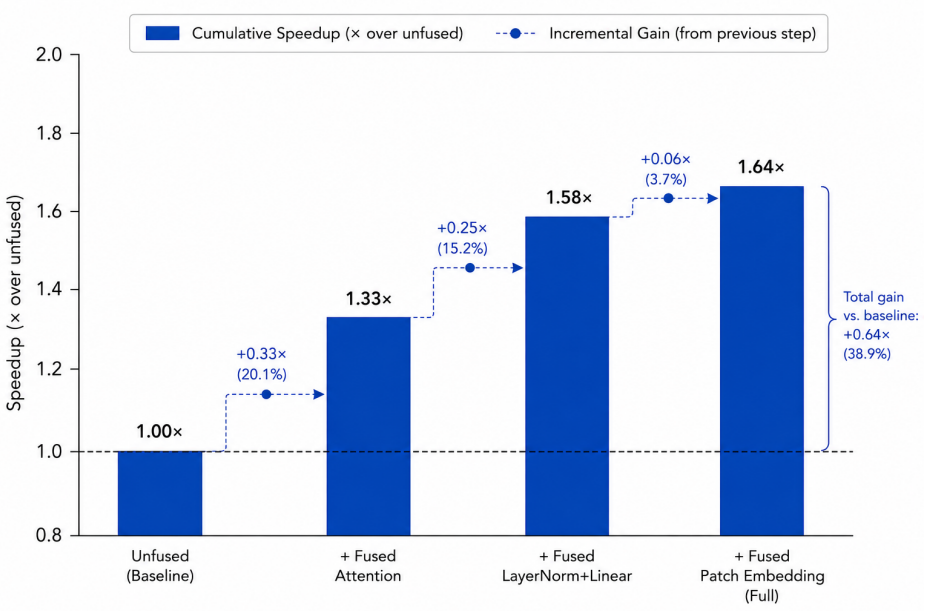

Figure 2: Operator-level attribution on the UNI encoder. Adding fused operators in sequence (attention, then LayerNorm-plus-Linear, then patch embedding) composes to a 1.64× end-to-end speedup over the unfused baseline, with the incremental gain from each operator annotated. The order of gains matches the profiled per-operator time share, so the speedup is explained by where the encoder actually spends memory traffic rather than by a single lucky kernel.

a busy department generates hundreds of slides per day. Raw slides and their embeddings carry PHI, and many institutions are reluctant or legally unable to ship them to a cloud endpoint, yet a department typically owns only one or a small number of mid-range cards, often with no more than 24 GB (Hanna et al., 2022). Encoder throughput, not model quality, is the wall that decides whether a vision foundation-model pathology pipeline can run inside the hospital at all (Fig. 1).

We replace the encoder's dominant per-tile operations with fused operators from the same Triton substrate, JIT-specialized for each device: FlashAttention (Dao et al., 2022), fused LayerNorm followed by Linear, and a fused patch-embedding operator. Model weights, numerics, and downstream classification heads remain unchanged, ensuring that the model under evaluation is held fixed. Profiling shows that encoder computation is dominated by attention ($\approx$ 46%), followed by MLP and normalization ($\approx$ 41%), patch embedding ($\approx$ 6%), and other operations ($\approx$ 7%). This breakdown suggests that end-to-end speedup should increase with backbone size as fixed I/O overhead becomes a smaller fraction of total runtime. Figure 2 validates this hypothesis through an incremental ablation that introduces each fused operator individually on UNI.

On a representative whole-slide image containing approximately 12,400 tiles, the speedup increases from 1.42× for a ViT-S backbone to 1.64× for UNI and 1.71× for Virchow (median wall-clock time: 129.6 $\rightarrow$ 79.1 s for UNI), confirming that larger backbones benefit more as the fixed I/O overhead becomes a smaller fraction of total runtime. The same optimized implementation delivers consistent results on AMD MI300X, with all measurements remaining within approximately 3–5% of their NVIDIA counterparts. From a deployment perspective (Table 1), the optimized operators raise throughput from 222 to 364 slides per 8-hour shift on NVIDIA and 224 to 348 on AMD, halve energy consumption per slide (20.2 $\rightarrow$ 10.0 Wh on NVIDIA, 20.4 $\rightarrow$ 10.6 Wh on AMD), and reduce peak memory usage to fit within the 24 GB capacity of a mid-range GPU. The two vendors sit within about 5% of each other in

Table 1: On-premises deployment metrics, UNI backbone, single on-site GPU, per slide (about 12,400 tiles). Cost is the optimized-vs-baseline ratio to stay robust to GPU price volatility; energy-to-cost conversion uses a \$0.12/kWh rate (U.S. Energy Information Administration, 2026). PHI never leaves the building.

Configuration	Energy/slide	Slides/shift	Cost/slide (rel.)
Local NVIDIA, baseline	20.2 Wh	222	1.00×
Local NVIDIA, Our optimized	10.0 Wh	364	0.56×
Local AMD, baseline	20.4 Wh	224	1.01×
Local AMD, Our optimized	10.6 Wh	348	0.58×

both baseline and optimized energy per slide, while all on-premises deployments keep PHI entirely on site, directly satisfying the data-retention requirement.

Energy measurement methodology. Energy is integrated board power sampled from on-device telemetry (pynvml on NVIDIA, rocm-smi on AMD, sampled at 100 ms) over the full end-to-end slide encode, so tile decode and host-to-device transfer fall inside the integration window. Idle board power is not subtracted, so the reported figure is the one a facilities budget would see; we report board energy only, excluding host CPU, storage, and cooling, which makes each Wh/slide figure a lower bound on total facility energy.

Diagnostic quality is held fixed. Acceleration is meaningful only if diagnostic performance is preserved. Accordingly, the protocol pairs every energy measurement with a quality gate. Across four diagnostic tasks, including tumor detection on CAMELYON16 (Bejnordi et al., 2017), subtyping on TCGA-NSCLC, biomarker prediction on TCGA-CRC, and grading on TCGA-RCC, each evaluated using an attention-based MIL head (Ilse et al., 2018; Lu et al., 2021) over five random seeds, every $|\Delta\text{AUC}|$ remains within the across-seed noise floor (≤ 0.001), while the mean embedding cosine similarity to the baseline is 0.9998. The gains in energy efficiency, hardware portability, and PHI retention therefore come with no measurable loss in diagnostic quality, providing the evidence-consistency guarantee required of a deployment benchmark.

4 The Protocol Generalizes: Protein and Genomic Foundation Models

The same instrument, unchanged, applies beyond pathology. We port six widely used protein and genomic foundation models to fused Triton kernels from a single source tree with no per-vendor forks, and measure them on NVIDIA L4, NVIDIA H100, and AMD MI300X. Mixed precision (FP16/BF16) is used with selective FP32 accumulators for softmax and normalization statistics. Every kernel must pass the fidelity gate (cosine similarity ≥ 0.99999, exact-match top-1 discrete predictions, or zero numerical difference to six decimal places); kernels that fail are excluded, so every reported speedup is quality-preserving by construction.

Table 2 reports the headline measurements. For the transformer-heavy models (ESM-2, ProtBERT, AlphaGenome) the wins come from fusing QKV projection, bias, and post-attention normalization into one kernel plus an IO-aware fused-attention path, the same recipe used for the pathology encoder; the largest ESM-2 win coincides with the long-sequence regime where the baseline activation no longer fits in cache. ProteinMPNN's 1.51× is the smallest and most informative win: its time is spent in geometric operations with no vendor-library equivalent, so the headroom for general infrastructure shrinks as models move further from the transformer template, which is precisely the regime a clinical-model deployment benchmark must cover.

Cross-vendor reproducibility, measured directly. The Enformer experiments provide the clearest evidence of cross-vendor reproducibility: the same Triton source executes unmodified on both H100 and MI300X at the full 196K-base sequence length. The difference in observed speedup (1.82× on H100 versus 1.43× on MI300X) reflects differences in the

Table 2: Fidelity-gated measurement across protein and genomic foundation models. Speedup is over the reference implementation on the listed GPU; fidelity is the gate used to admit each kernel. A single Triton source runs unmodified on all three devices, so the result holds across vendors.

Model	Family	Hardware	Speedup	Fidelity
Meta ESM-2 (8M)	Protein LM	NVIDIA L4	43.1×	cos. > 0.99999
ProtBERT	Protein LM	AMD MI300X	3.6×	top-1 exact (12/12)
ProteinMPNN	Structure-to-seq	NVIDIA L4	1.51×	exact sequence
DeepMind AlphaGenome	Genomic attention	NVIDIA H100	5.05×	cos. 0.99999997
DeepMind Enformer	Genomic attention	NVIDIA H100	1.82×	cos. 0.99999825
DeepMind Enformer	Genomic attention	AMD MI300X	1.43×	cos. 0.99999967
Nvidia DualBind	Binding affinity	AMD MI300X	38.8×	exact (6-dec)
Nvidia DualBind	Binding affinity	NVIDIA H100	4.22×	exact (6-dec)

baseline implementations rather than the optimized kernels. Although the H100 baseline is slower in absolute terms, it contains less unfused overhead to eliminate. These results demonstrate that a single implementation can deliver consistent performance improvements across both hardware vendors without requiring vendor-specific kernels or code changes. The optimized ESM-2 implementation further achieves speedups of up to 74.7× on the longest input sequences, reduces peak memory usage by up to 68.9%, and lowers GPU execution time by 97.7% relative to the Hugging Face baseline while preserving embedding fidelity.

5 A Cross-Vendor Reproducibility Hazard the Protocol Surfaces

The cross-vendor claim above has a sharp edge any institution adopting a second vendor must benchmark for, and it is a deployment-safety issue, not just an engineering one. During profiling of the DualBind binding-affinity model we found the upstream reference was not GPU-resident end-to-end: after the transformer pass, the reference implementation moved per-sample logits back to host memory to evaluate the binding-energy scoring function in NumPy before returning to the GPU for the next stage, rather than keeping that computation GPU-resident. The kernel files are identical between vendors, yet the measured end-to-end speedup is 38.8× on AMD MI300X and only 4.22× on NVIDIA H100. The reason is not a near-10× hardware gap. On H100, NVIDIA's PCIe controller, unified-memory prefetcher, and CUDA runtime mask most of the round-trip cost; on MI300X the round-trip is exposed, and 800 samples take 41.3 s instead of 1.06 s once a fused, GPU-resident kernel removes it. We fixed this specific hazard by rewriting the scoring function as a Triton kernel, which was necessary here because the scoring step needed to fuse into the surrounding kernel to be GPU-resident at all; a smaller, non-fused GPU-resident rewrite of just the scoring step (leaving the rest of the pipeline unfused) would likely have removed most of the round-trip cost too, so fusion was sufficient but not shown to be necessary for closing this particular gap.

We want to be precise about what this example does and does not establish. It is one exemplar in one upstream implementation, not a characterized population of such hazards; we have not systematically audited other reference implementations for the same pattern, so we do not claim this is common, only that it is possible and, when present, invisible to single-vendor benchmarking. The benchmarking lesson is general and governance-relevant regardless of prevalence: a result reported on the vendor a model was developed on can carry a latent systems behavior that stays invisible there and becomes first-order on another, so a single-vendor number does not transfer across sites without being checked; a write-once, GPU-resident fused kernel masks the asymmetry on both, bringing the vendors within about 1.4× of each other in absolute throughput. This is exactly the risk a hospital faces when it buys the accelerator its budget allows rather than the one the code was written on, which is why we treat cross-vendor reproducibility as a measured property rather than an assumption. It also reframes the AMD number correctly: the 38.8× is not evidence that

MI300X is intrinsically faster, but that a latent reproducibility hazard was real for this model and an evidence-consistent benchmark must report it rather than average it away.

6 Real-Time Intensive-Care Monitoring: A Latency Benchmark

The protocol extends to a modality where the bottleneck is not the model at all. Real-time intensive-care early-warning systems require sub-50 ms inference for bedside updates at 20 Hz (Henry et al., 2015), but the physiological streams they consume are irregularly sampled with more than 30% missing values (Silva et al., 2012), forcing deep pipelines into sequential preprocessing that dominates wall-clock time. Profiling a state-space-model pipeline (Gu & Dao, 2024; Gu et al., 2022) on the PhysioNet 2012 challenge attributes more than 85% of end-to-end latency to time-aware interpolation of the irregular input, not the model forward pass, so a benchmark that only times the forward pass misses the real bottleneck.

We fuse the entire interpolation into a single Triton kernel that launches one block per (batch, time, feature-tile) combination and performs a bounded neighbor search with early-exit and coalesced memory access, written once and run on both vendors. This is the paper's deployment contribution for this section, and it is strictly fidelity-preserving: outputs match the unfused interpolation to within 5×10^{-7}, so by construction it cannot change what the downstream model predicts. Table 3 reports its effect: a 53.6 to 84.6× kernel-level speedup cuts end-to-end latency from 805 ms to 23 ms (35.7×) at batch 32 and lifts peak throughput 82.8×. The kernel is bandwidth-bound and near the roofline (Williams et al., 2009) on both vendors, and energy per inference drops from 52.0 to 5.6 mJ on NVIDIA and 60.1 to 7.2 mJ on AMD, so latency, throughput, and energy all hold across vendors.

We report one further, kernel-*independent* observation alongside these deployment numbers because it uses the same pipeline: swapping the sequence model itself from GRU-D to a state-space model (Gu & Dao, 2024; Gu et al., 2022) trains 2.29× faster and, across five seeds, improves test AUROC by $+0.037$ (paired bootstrap 95% CI $[0.018, 0.058]$, $p < 0.01$) on PhysioNet 2012, with $+0.024$ macro-AUROC on MIMIC-III 25-task phenotyping (Harutyunyan et al., 2019). This predictive gain comes from the choice of sequence model, not from the interpolation kernel, which leaves the model's inputs numerically unchanged; we report it here only because it uses the same evaluation pipeline, and we do not count it as a deployment-efficiency result.

Table 3: Real-time ICU pipeline on PhysioNet 2012 (batch 32 unless noted). The same Triton kernels run on NVIDIA and AMD without code change. The first four rows are the fidelity-preserving deployment result (interpolation kernel only, model held fixed); the last two rows are a separate, kernel-independent modeling comparison (state-space model vs. GRU-D) included because it shares the same pipeline, not because the kernel caused it.

Quantity	Baseline	Fused Triton	Improvement
End-to-end latency	805 ms (PyTorch)	23 ms	35.7× lower
Interpolation kernel	1.74–2.94 ms	0.02–0.05 ms	53.6–84.6×
Peak throughput	125 samp/s	10,316 samp/s	82.8× higher
Tail latency (p99)	> 800 ms	44.6 ms	< 50 ms met
Training time	GRU-D	2.29× faster	–
Test AUROC (PhysioNet)	0.622 (GRU-D)	0.659	$+0.037$ $[0.018, 0.058]$
Macro-AUROC (MIMIC-III)	0.738 (GRU-D)	0.762	$+0.024$ $[0.014, 0.036]$

7 Related Work

Existing benchmarks for clinical foundation models primarily emphasize predictive performance, reporting metrics such as accuracy, AUC, and Dice on fixed evaluation cohorts, with more recent work incorporating robustness and cross-site generalization (Chen et al., 2024; Vorontsov et al., 2024; Singhal et al., 2023). These benchmarks establish whether a

model is clinically effective, but not whether it can be deployed where the data reside. We instead focus on deployment-oriented evaluation, introducing metrics for energy per inference, end-to-end and tail latency, cross-vendor reproducibility, on-site PHI retention, and fidelity-preserving acceleration, all evaluated under explicit numerical fidelity constraints. Although energy-aware evaluation has been advocated as a first-class consideration (Strubell et al., 2019; Lacoste et al., 2019), prior work has not combined these deployment dimensions within a single protocol that keeps model weights and numerics fixed while evaluating across hardware vendors.

From a systems perspective, FlashAttention demonstrated the benefits of IO-aware attention on NVIDIA GPUs (Dao et al., 2022; Dao, 2023), while Triton provides a portable programming model for high-performance GPU kernels (Tillet et al., 2019). Existing optimization efforts for clinical foundation models have largely relied on CUDA-specific, single-vendor implementations. In contrast, we present a deployment evaluation protocol based on a single Triton implementation that executes unmodified across NVIDIA and AMD GPUs, enabling controlled, cross-vendor measurements of deployment efficiency and reproducibility.

8 Discussion and Conclusion

We argued that benchmarking clinical language and vision-language models for high-stakes use must measure deployment, not accuracy alone, and proposed a protocol reporting fidelity-gated speedup, energy per inference, on-site PHI retention, and cross-vendor reproducibility under a single fused-kernel substrate authored once in Triton and run unmodified on NVIDIA and AMD. Across whole-slide pathology, protein and genomic models, and real-time intensive-care monitoring, it measured a roughly halved on-premises encoding energy at fixed diagnostic quality, speedups of $1.51\times$ to $43.1\times$ at high fidelity, a latency collapse from 805 ms to 23 ms, and a cross-vendor hazard in which a latent host-device round-trip turns one model's speedup from $4.22\times$ into $38.8\times$, so a single-vendor number does not transfer across sites. Evidence-consistent evaluation of clinical AI must therefore diagnose *where* and *how* a model is served. Limitations include single-GPU inference-time measurement, a single AMD SKU, and public-cohort validation needing site-specific confirmation before clinical use.

Acknowledgments

The author thanks Ghee Leng Ooi and Pin Siang Tan, co-founders of Embedded LLM, for their support and guidance, and the Embedded LLM team for helpful discussions on cross-vendor GPU kernel deployment.

References

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R. Ledsam, Agnieszka Grabska-Barwinska, Kyle R. Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R. Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature Methods*, 18(10):1196–1203, 2021.

Žiga Avsec, Natasha Latysheva, Jun Cheng, Guido Novati, Kyle R. Taylor, Tom Ward, et al. Advancing regulatory variant effect prediction with alphagenome. *Nature*, 649(8099): 1206–1218, 2026. doi: 10.1038/s41586-025-10014-0.

Babak Ehteshami Bejnordi, Mitko Veta, Paul Johannes van Diest, et al. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *JAMA*, 318(22):2199–2210, 2017. doi: 10.1001/jama.2017.14585.

Richard J. Chen, Tong Ding, Ming Y. Lu, Drew F. K. Williamson, Guillaume Jaume, Andrew H. Song, Bowen Chen, Andrew Zhang, Daniel Shao, Muhammad Shaban, et al. Towards a general-purpose foundation model for computational pathology. *Nature Medicine*, 30(3):850–862, 2024. doi: 10.1038/s41591-024-02857-3.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Alexis Courbet, Rob J. de Haas, Neville Bethel, et al. Robust deep learning-based protein sequence design using proteinmpnn. *Science*, 378(6615): 49–56, 2022. doi: 10.1126/science.add2187.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. doi: 10.1109/TPAMI.2021.3095381.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2024.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

Matthew G. Hanna, Orly Ardon, Victor E. Reuter, S. Joseph Sirintrapun, Christine England, David S. Klimstra, and Meera R. Hameed. Integrating digital pathology into clinical practice. *Modern Pathology*, 35(2):152–164, 2022. doi: 10.1038/s41379-021-00929-0.

Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6(1): 96, 2019.

Katharine E. Henry, David N. Hager, Peter J. Pronovost, and Suchi Saria. A targeted real-time early warning score (trewscore) for septic shock. *Science Translational Medicine*, 7 (299):299ra122, 2015.

Maximilian Ilse, Jakub M. Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 2127–2136, 2018.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*, 2019.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023. doi: 10.1126/science.ade2574.

Meng Liu and Saurabh G. Paliwal. Dualbind: A dual-loss framework for protein-ligand binding affinity prediction. *arXiv preprint arXiv:2406.07770*, 2024.

Ming Y. Lu, Drew F. K. Williamson, Tiffany Y. Chen, Richard J. Chen, Maha Barbieri, and Faisal Mahmood. Data-efficient and weakly supervised computational pathology on whole-slide images. *Nature Biomedical Engineering*, 5(6):555–570, 2021. doi: 10.1038/s41551-020-00682-w.

Ming Y. Lu, Bowen Chen, Drew F. K. Williamson, Richard J. Chen, Ivy Liang, Tong Ding, Guillaume Jaume, Igor Odintsov, Long Phi Le, Georg Gerber, et al. A visual-language foundation model for computational pathology. *Nature Medicine*, 30(3):863–874, 2024. doi: 10.1038/s41591-024-02856-4.

Michael Moor, Oishi Banerjee, Zahra Shakeri Hossein Abad, Harlan M. Krumholz, Jure Leskovec, Eric J. Topol, and Pranav Rajpurkar. Foundation models for generalist medical artificial intelligence. *Nature*, 616(7956):259–265, 2023. doi: 10.1038/s41586-023-05881-4.

Ikaro Silva, George Moody, Daniel J. Scott, Leo A. Celi, and Roger G. Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. *Computing in Cardiology*, 39:245–248, 2012.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, 2023. doi: 10.1038/s41586-023-06291-2.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 3645–3650, 2019.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature Medicine*, 29(8):1930–1940, 2023. doi: 10.1038/s41591-023-02448-8.

Philippe Tillet, H. T. Kung, and David Cox. Triton: An intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages (MAPL)*, pp. 10–19, 2019. doi: 10.1145/3315508.3329973.

U.S. Energy Information Administration. Electric power monthly, table 5.6: Average price of electricity to ultimate customers by end-use sector. 2026. Accessed June 2026.

Eugene Vorontsov, Alican Bozkurt, Adam Casson, George Shaikovski, Michal Zelechowski, Kristen Severson, et al. A foundation model for clinical-grade computational pathology and rare cancers detection. *Nature Medicine*, 30(10):2924–2935, 2024. doi: 10.1038/s41591-024-03141-0.

Samuel Williams, Andrew Waterman, and David Patterson. Roofline: An insightful visual performance model for multicore architectures. *Communications of the ACM*, 52:65–76, 2009. doi: 10.1145/1498765.1498785.