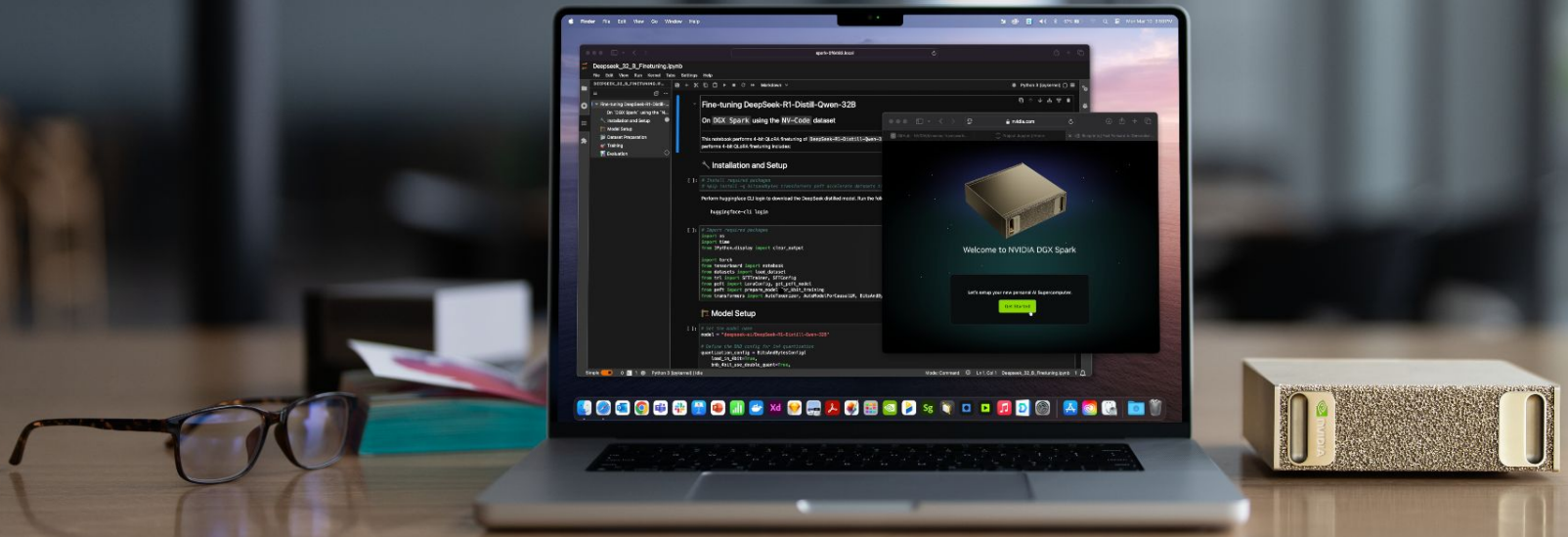
AI Prototyping on DGX Spark (aka Sparky)



Problem to solve

- Why is AI engineering or data science so hard?
- Big workstation vs. cloud
- Easy access to latest technologies

Motivation for Sparky?

- Few things I wanted to learn
- I asked myself: What is the best way to learn about GPU infrastructure design, and AI plumbing?

Knowledge check

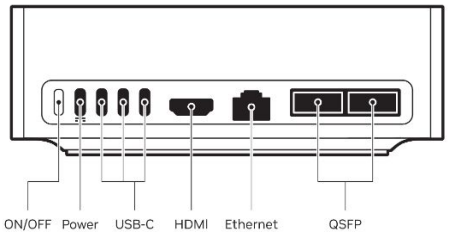
- AI for work
- AI plumbing
- Data science
- Python or Linux

Why a local GPU Computer?

- Powerful laptops are not optimized for AI workloads.
- No complexity from cloud wrapper.
- No extra cloud costs.
- Fast feedback loop.
- Ability to run latest releases at anytime.
- Access to the market-leading ecosystem and tooling stories.
- Full access for troubleshooting and metrics.

Box design

GPU	NVIDIA Blackwell Architecture - 6,144 CUDA cores - general parallel processing - 192 Tensor Cores (5th Gen) - AI matrix math and sparse FP4 operations - 48 RT Cores (4th Gen) - ray-tracing workloads and synthetic data generation
CPU	20 core ARM processor - 10 Cortex-X925 cores - performance of compute heavy and serial work - 10 Cortex-A725 cores - efficiency for background and system tasks - Shared cache optimized for mixed workloads
Memory	128 GB LPDDR5x unified system memory, 256-bit interface, 4266 MHz, 273 GB/s bandwidth - Fully coherent CPU and GPU memory with zero copy access
Storage	4 TB NVMe M.2 with self-encryption
Network	1x RJ-45 (10 GbE), ConnectX-7 Smart NIC, Wi-Fi 7, Bluetooth 5.4
Connectivity	4x USB Type-C, 1x HDMI 2.1a



DGX comparison

	DGX-1 (2016)	DGX Spark (2025)	DGX B200 (2025)
GPU	NVIDIA Pascal	NVIDIA Grace Blackwell	8x NVIDIA B200
GPU memory	128GB (16GB per GPU)	128GB unified system	1,440 GB total GPU
AI Performance	170 TFLOPS (FP16)	1 petaFLOPS (FP4)	144 petaFLOPS (FP4)
System Power	3,200 W	240 W	14.3 kW
Dimensions	34.1 in. x 17.48 in. x 5.16 in.	5.91 in. 5.91 in. x 1.99 in.	17.5 in. x 19.0 in. x 35.3 in.
System Weight	134 lb.	2.65 lb.	313.9 lb
Price	$129.000	$3.999	~$500k

NVIDIA AI software stack on DGX Spark

- NVIDIA DGX OS
- NVIDIA Sync tool
- DGX dashboard
- AI Workbench
- NIM = NVIDIA Inference Microservice
- NVIDIA Brev - One click deployments with Launchables.
- NGC = NVIDIA GPU Cloud (plus container registry)
- JupyterLab
- Playbooks, Blueprints, Forum, etc.

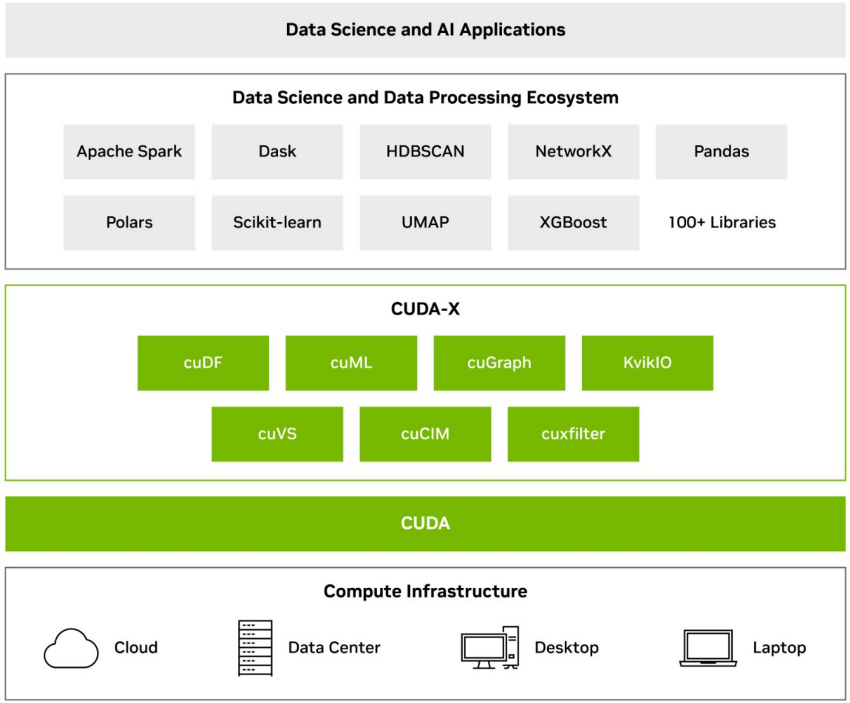
Demo: Tooling

Prototyping scenarios

- Inference experiments
- RAG ingestion pipelines
- Synthetics
- Fine-tuning models
- Multimodal
- AI plumbing

GPU-accelerated libraries

- CUDA Math libraries
- Parallel Algorithm libraries
- Data Processing libraries
- Communication libraries
- Quantum libraries
- Partner libraries
- Computational Lithography
- Image and Video libraries
- Deep Learning Core



Real performance

- Fine-tuning up to 70b parameters
- Inference up to 200B, 400B with 2 boxes
- TensorRT-LLM inference, best optimized path
- nvidia/Llama-3.3-70B-Instruct-NVFP4, optimized models

Demo: Inference Benchmark

~/demos/**machine_compare** 14:40:46
> llama-bench -m /Users/chris/models/Qwen2.5-7B-Instruct-GGUF/qwen2.5-7b-instruct-q4_k_m-00001-of-00002.gguf -p 512 -n 256 -r 5
ggml_metal_device_init: tensor API disabled for pre-M5 and pre-A19 devices
ggml_metal_library_init: using embedded metal library
ggml_metal_library_init: loaded in 0.006 sec
ggml_metal_rsets_init: creating a residency set collection (keep_alive = 180 s)
ggml_metal_device_init: GPU name: Apple M4
ggml_metal_device_init: GPU family: MTLGPUFamilyApple9 (1009)
ggml_metal_device_init: GPU family: MTLGPUFamilyCommon3 (3003)
ggml_metal_device_init: GPU family: MTLGPUFamilyMetal4 (5002)
ggml_metal_device_init: simdgroup reduction   = true
ggml_metal_device_init: simdgroup matrix mul. = true
ggml_metal_device_init: has unified memory   = true
ggml_metal_device_init: has bfloat   = true
ggml_metal_device_init: has tensor   = false
ggml_metal_device_init: use residency sets   = true
ggml_metal_device_init: use shared buffers   = true
ggml_metal_device_init: recommendedMaxWorkingSetSize = 19069.67 MB
model	size	params	backend	threads	test	t/s
qwen2 7B Q4_K - Medium	4.36 GiB	7.62 B	Metal,BLAS	4	pp512	241.76 ± 1.36
qwen2 7B Q4_K - Medium	4.36 GiB	7.62 B	Metal,BLAS	4	tg256	21.41 ± 0.74
build: 193ee38a1 (7653)

chris@spark-38fc:**~/Demos/machine-compare**$ llama-bench -m /home/chris/models/Qwen2.5-7B-Instruct-GGUF/qwen2.5-7b-instruct-q4_k_m-00001-of-00002.gguf -p 512 -n 256 -r 5
ggml_cuda_init: found 1 CUDA devices:
  Device 0: NVIDIA GB10, compute capability 12.1, VMM: yes
model	size	params	backend	ngl	test	t/s
qwen2 7B Q4_K - Medium	4.36 GiB	7.62 B	CUDA	99	pp512	3064.96 ± 17.73
qwen2 7B Q4_K - Medium	4.36 GiB	7.62 B	CUDA	99	tg256	47.92 ± 0.03
build: 193ee38a1 (7653)

Takeaway

- Comparison DGX Spark vs. MacBook Pro M4 24GB RAM
  - Inference engine lama.cpp
  - Model: Qwen2.5-7B-Instruct-GGUF
  - 5x tests each
  - Test pp512: prompt processing with 512 tokens
  - Test tg256: token generation
- Running gpt-oss-120b running by trtllm
  - MacBook Pro would collapse
  - Optimize for batching, concurrency, sustained load
  - Production grade latency consistency

Demo: Embedding Generation

(.venv) chris@spark-38fc:~/Demos/dgx-embeddings-demo$ python scripts/run_cpu.py --data data/corpus.jsonl --model BAAI/bge-large-en-v1.5 --max_samples 200000 --warmup_s 20 --measure_s 60
/home/chris/Demos/dgx-embeddings-demo/.venv/lib/python3.12/site-packages/torch/cuda/__init__.py:63: FutureWarning: The pynvml package is deprecated. Please install nvidia-ml-py instead. If you did not install pynvml directly, please report this to the maintainers of the package that installed pynvml for you.
  import pynvml # type: ignore[import]

Mode	Model	Batch policy	Rolling eps	Avg eps	p50 ms	p95 ms	GPU util	GPU mem MiB	Progress
CPU baseline	BAAI/bge-large-en-v1.5	fixed batch=32	2	2	18286.6	18408.1	0	-	128/200,000

Done. Avg eps=2 p50=18286.6ms p95=18408.1ms

(.venv) chris@spark-38fc:~/Demos/dgx-embeddings-demo$ python scripts/run_gpu_naive.py --data data/corpus.jsonl --model BAAI/bge-large-en-v1.5 --device cuda --max_samples 200000 --warmup_s 20 --measure_s 60
/home/chris/Demos/dgx-embeddings-demo/.venv/lib/python3.12/site-packages/torch/cuda/__init__.py:61: FutureWarning: The pynvml package is deprecated. Please install nvidia-ml-py instead. If you did not install pynvml directly, please report this to the maintainers of the package that installed pynvml for you.
  import pynvml # type: ignore[import]

Mode	Model	Batch policy	Rolling eps	Avg eps	p50 ms	p95 ms	GPU util	GPU mem MiB	Progress
GPU naive	BAAI/bge-large-en-v1.5	fixed batch=64 no bucketing	35	35	1847.1	1865.0	96	-	2,176/200,000

Done. Avg eps=35 p50=1847.1ms p95=1865.0ms

(.venv) chris@spark-38fc:~/Demos/dgx-embeddings-demo$ python scripts/run_gpu_tuned.py --data data/corpus.jsonl --model BAAI/bge-large-en-v1.5 --device cuda --max_samples 200000 --warmup_s 20 --measure_s 60
/home/chris/Demos/dgx-embeddings-demo/.venv/lib/python3.12/site-packages/torch/cuda/__init__.py:61: FutureWarning: The pynvml package is deprecated. Please install nvidia-ml-py instead. If you did not install pynvml directly, please report this to the maintainers of the package that installed pynvml for you.
  import pynvml # type: ignore[import]

Mode	Model	Batch policy	Rolling eps	Avg eps	p50 ms	p95 ms	GPU util	GPU mem MiB	Progress
GPU tuned	BAAI/bge-large-en-v1.5	token_budget=16384 bucketing max_batch=256	370	366	700.8	820.9	96	-	22,272/200,000

Done. Avg eps=366 p50=700.8ms p95=820.9ms

Takeaway

- BAAI bge-large-en-v1.5 is a text embedding model with ~ 300M parameters
- SentencesTransformers
  - Bucketing text of different length with higher parallelization
  - Creates vectors to capture meaning
- Avoids padding and too big batches
  - token_budget=16384
  - max_batch=256
  - buckets=(200, 400, 800, 1400, 2200, 3200)

SuperPower: Sparky

- DGX Spark is not replacing cloud, or making laptop faster
- Less friction during prototyping
- Less memory limitations
- Less iteration delays
- Less cloud overhead
- Your local AI lab

Q&A