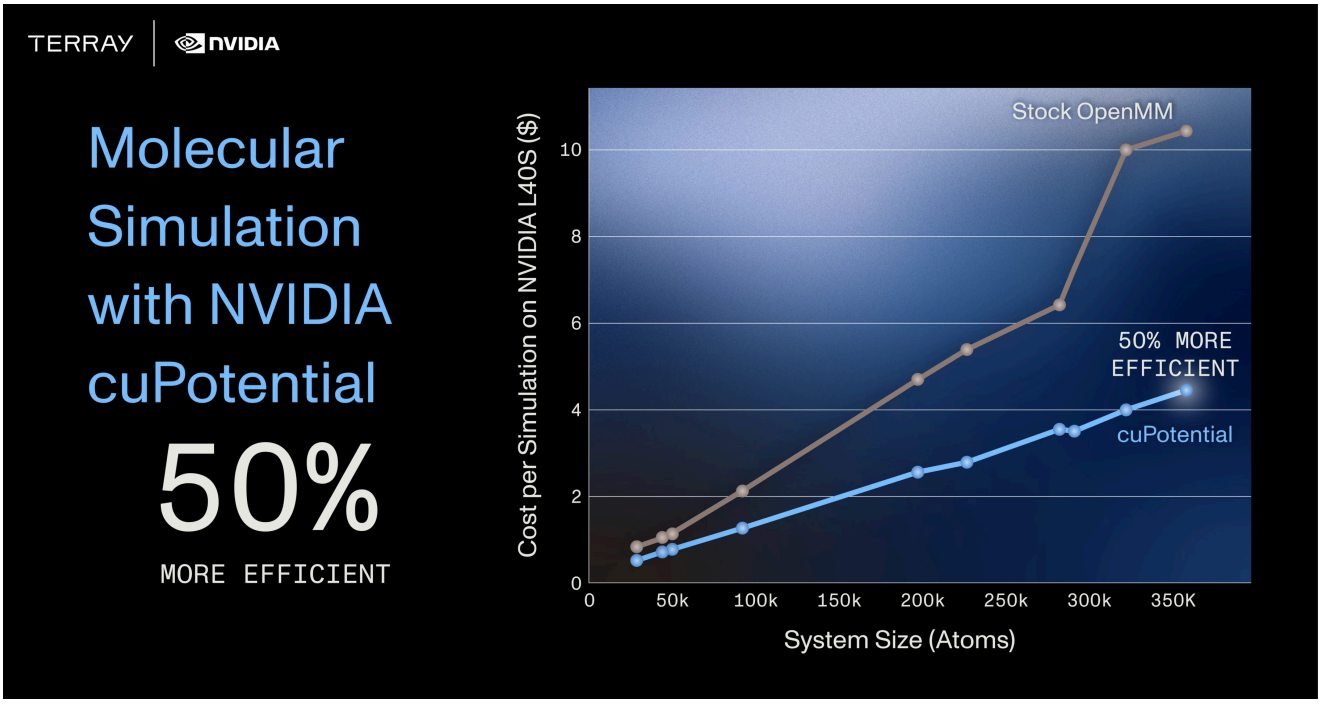
Terray Therapeutics • 2026
Source: https://www.terraytx.ai/news-and-research/terrabind-generating-synthetic-data-NVIDIA-BioNeMo

Accelerating TerraBind and Generating Synthetic Training Data Economically with NVIDIA BioNeMo

How NVIDIA software is making Terray's AI models faster and synthetic training data generation cheaper

Nov 16, 2026
Authors
Gavin Bascom
Dominic Rufa
Narbe Mardirossian, Ph.D.



Drug design is a complex multi-parameter optimization problem, balancing target engagement and safety with essential properties like stability and solubility. Because drug-like chemical space is intractably vast, computation is essential to prioritize which molecules are actually worth synthesizing and testing. Across all parameters, potency is the non-negotiable anchor – if a molecule doesn't bind its target, it isn't worth pursuing. Yet modeling potency *in silico* has historically forced a trade-off between **speed** (how much chemical space can be evaluated) and **accuracy** (how reliably the underlying interactions are simulated or predicted).

Through our ongoing collaboration with NVIDIA, we have upgraded TerraBind, EMMI's universal potency model, to transcend this trade-off on both fronts simultaneously. **Downstream, we improved speed:** integrating NVIDIA's CuTeDSL kernels into production slashed memory overhead—eliminating out-of-memory bottlenecks on massive target complexes—while accelerating inference by up to 7x to unlock broad chemical space exploration. **Upstream, we boosted accuracy:** while the unmatched stream of iterative wet-lab data from Terray's EMMI platform drives TerraBind's industry-leading performance, we can push generalizability and precision even further by augmenting it with atomistic simulation via Molecular Dynamics. Because synthetic *in silico* data must be generated at massive scale to meaningfully complement high-value wet-lab measurements, ultra-low compute cost is essential. Using the new NVIDIA cuPotential library to reduce synthetic data generation costs by up to 2.5x, we built an on-the-fly simulation pipeline that continuously trains TerraBind on rich conformational dynamics, sharpening potency prediction accuracy. By uniting high-throughput inference with physically-grounded training, we are transforming how candidate space is navigated from the ground up.

Part 1: Faster Triangle Attention in TerraBind

Triangle attention is a key component of AlphaFold-class models and the dominant cost in TerraBind, both during training and inference. It updates every residue-pair representation by attending over a third residue, so its cost grows as $O(n^3)$ in the number of tokens. For protein-ligand complexes, where we want large crops that capture the whole binding site and its surroundings, that cubic term dominates.

We benchmarked a 48-layer Pairformer trunk (pair dimension 128, bf16, three recycles) on NVIDIA H100 GPUs. We swapped the triangle attention modules among three implementations: vanilla PyTorch, cuEquivariance (our current optimized stack), and NVIDIA's CuTeDSL kernels from NVIDIA BioNeMo™ Inference Runtime.

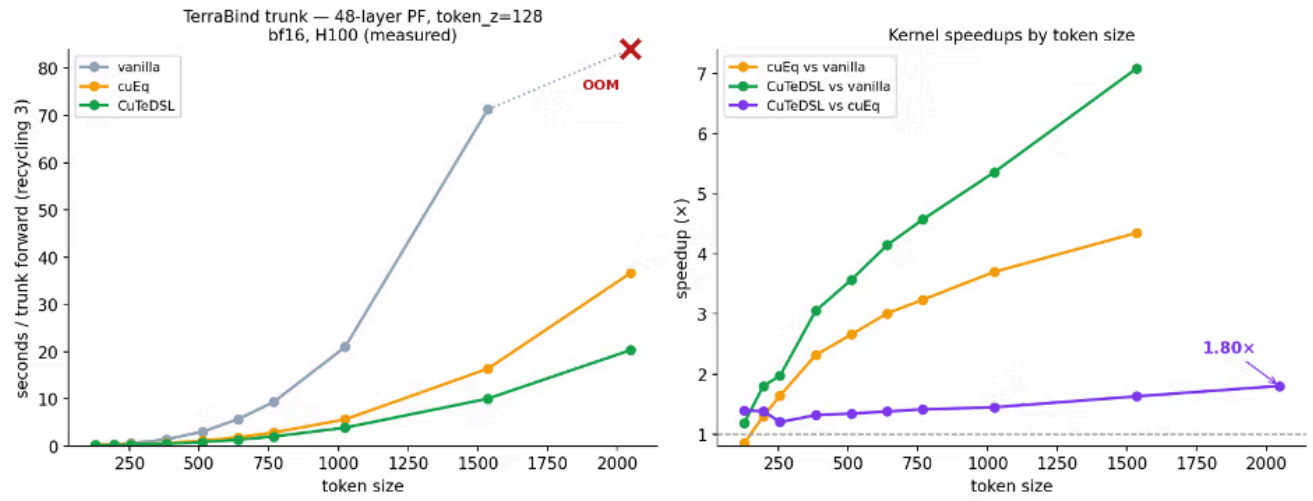


Figure 1. Left: TerraBind trunk forward time versus token count. Vanilla PyTorch runs out of memory above ~1,500 tokens. Right: Speedup of each kernel stack relative to the others. CuTeDSL reaches 7.1× over vanilla and 1.80× over cuEquivariance at the largest crop measured, and the CuTeDSL-vs-cuEquivariance advantage rises steadily with token count.

At small crops, the three stacks are close, since fixed overheads dominate. As crops grow, the optimized kernels pull away: at 2,048 tokens, CuTeDSL completes a trunk forward pass in roughly half the time of cuEquivariance. Memory tells a similar story: at 1,536 tokens, standard PyTorch attention peaks above 62 GB where the fused kernels use ~10 GB (a 6× reduction) and at 2,048 tokens PyTorch no longer fits on an 80 GB H100 at all, while cuEquivariance and CuTeDSL run comfortably in ~18 GB, letting us reach system sizes that were previously out of reach on a single GPU. Integration was straightforward: we had the kernels running and benchmarked within a day of getting access. These were module-level swaps, with no pipeline rewrites.

Part 2: Cheaper Simulation Data with cuPotential

Our co-folding structural model, TerraBind, is trained on hundreds of thousands [NM1] (#_msocom_1) [MR2] (#_msocom_2) of experimental crystal structures. However, a crystal structure is a single static snapshot of a protein-ligand complex captured under non-physiological conditions. This limitation obscures the true thermodynamically accessible conformational space and dynamic interactions that govern binding strength and experimental assay readouts.

To capture these important protein-ligand conformational ensembles, we augment our structural data with Molecular Dynamics (MD) simulation data. To achieve scale and ensure reliably converged simulations, each system is simulated using **Hamiltonian replica exchange**: six replicas progressively soften the ligand's torsional barriers and swap configurations every 2 ps. This protocol helps the ligand cross conformational barriers that plain MD would take far longer to traverse. A standard run consists of 2,000 exchanges (4 ns per replica) saving 2,000 physical frames. From these frames, we collate complex features that TerraBind can leverage for increased accuracy and precision.

The catch is scale. Every additional system costs GPU-hours; consequently, simulation throughput directly sets how much physics the model gets to see.

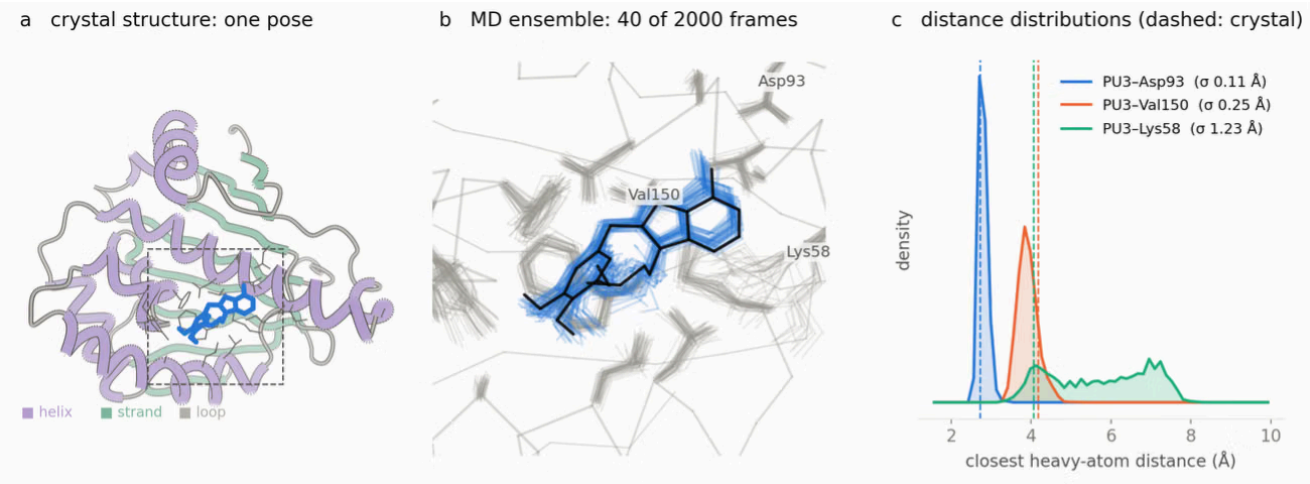


Figure 2. One crystal pose versus its MD ensemble: the N-terminal domain of HSP90 bound to the purine-scaffold inhibitor PU3 (PDB 1UY6). (a) The crystal structure gives a single pose (backbone colored by secondary structure; pocket side chains in gray; dashed box marks the region enlarged in b). (b) Forty of the 2,000 saved frames, overlaid in the pocket. The ligand (blue; black is the prepared crystal pose) and the surrounding side chains (gray) move around the crystal geometry. (c) Per-frame closest heavy-atom distance between PU3 and three pocket residues. The hydrogen bond to Asp93

barely moves (σ 0.11 Å), Val150 packs against the ligand with modest motion (σ 0.25 Å), and Lys58 swings between contact and a solvent-exposed state 7 Å away (σ 1.23 Å), information a single structure cannot carry.

The bottleneck in stock OpenMM

We run simulations with OpenMM, an open-source, GPU-accelerated molecular dynamics engine. A primary performance bottleneck in MD stems from computing nonbonded interactions, specifically the kernels required to compute **repulsion, electrostatics, and dispersion forces every integration timestep**. A key limitation in stock OpenMM is that as system sizes grow, simulation efficiency deteriorates, a consequence of core saturation; this becomes a particularly frustrating aspect of generating data for large systems, which pervades the PDB (see Fig.3).

cuPotential improves simulation efficiency

cuPotential, NVIDIA's drop-in plugin for OpenMM, directly tackles these scaling bottlenecks. First, cuPotential directly improves the execution speed of the nonbonded kernels, dramatically reducing the simulation cost and wall-clock times for large systems. As a direct consequence, the cost-per-simulation becomes nearly linear in system size. Second, cuPotential reduces the kernel execution overhead, yielding a roughly constant increase in simulation speed, independent of system size. Finally, cuPotential serves as a near zero-effort drop-in replacement for our MD workflows. Not only does it integrate seamlessly with our replica exchange-based sampling methodology and software, but it does so while matching OpenMM-based nonbonded energies with relative errors less than 1 part in ten million, ensuring consistency and simulation reliability.

Benchmark setup

To isolate the plugin's effect, both arms run in the same container image, with the same systems, the same preparation pipeline, and the same 2,000-exchange protocol. The only difference is whether cuPotential is enabled. We selected 11 protein-ligand systems from our dataset, spanning 29,000 to 359,000 atoms after solvation (a 12-fold size range), and ran each to completion on an NVIDIA L40S GPU (AWS g6e.2xlarge, on-demand pricing).

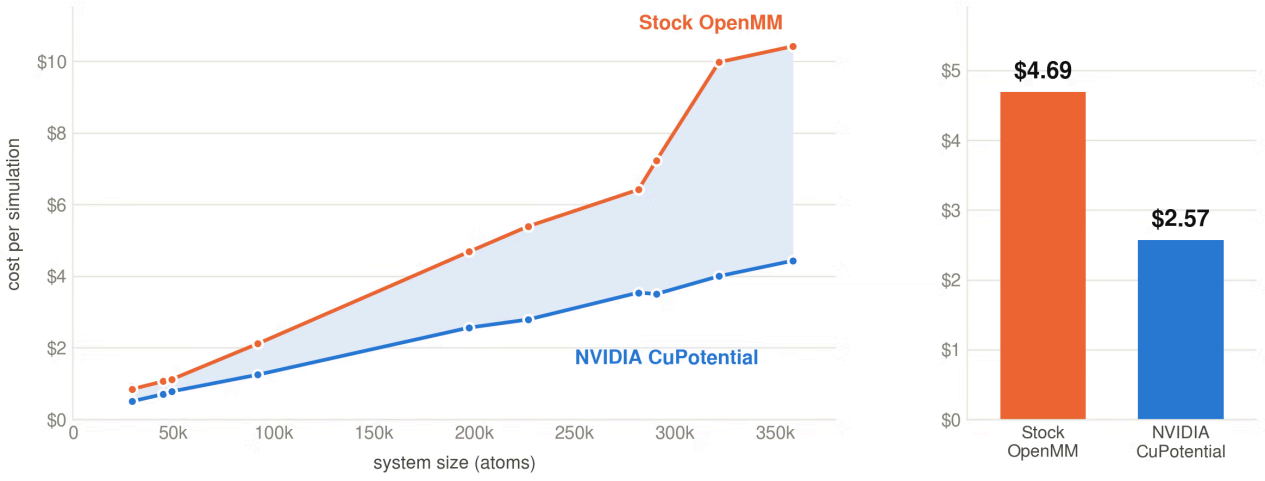


Figure 3. Cost per simulation on NVIDIA L40S, stock OpenMM versus cuPotential, across the 11

measured systems. Left: cost per 2,000-exchange simulation versus system size; the shaded region is the saving. Right: median cost across the 11 systems, $4.69 with stock OpenMM versus $2.57 with cuPotential..

System	Atoms	Stock OpenMM		cuPo		
		Time (h)	Cost	Time (h)	Cost	Speedup
4ku7	29,110	0.42	$0.93	0.23	$0.52	1.80×
5z02	29,271	0.38	$0.85	0.23	$0.51	1.66×
1bx1	44,875	0.48	$1.07	0.32	$0.71	1.50×
3rx3	49,122	0.50	$1.13	0.35	$0.79	1.43×
3sm8	92,035	0.95	$2.12	0.56	$1.26	1.69×
8gti	197,316	2.09	$4.69	1.15	$2.57	1.82×
3bt9	226,717	2.41	$5.40	1.25	$2.80	1.93×
4c50	281,666	2.86	$6.42	1.58	$3.55	1.81×
9jrn	290,608	3.23	$7.23	1.56	$3.51	2.06×
7bnu	321,806	4.45	$9.98	1.79	$4.01	2.49×
4s1d	358,638	4.65	$10.43	1.98	$4.44	2.35×

Table 1. Wall time and on-demand cost per 2,000-exchange simulation on a single L40S.

Results & What This Means for Synthetic Data

cuPotential is faster on every system we measured, by 1.43× to 2.49×. Three things stand out:

The advantage grows with system size. The speedup is around 1.5× for systems in the 30-50k-atom range, 1.7× at 92k, and 1.8-2.5× above 200k. Across all 11 systems, cuPotential cuts total simulation cost by 51%.

cuPotential’s per-atom cost falls and then flattens. Stock OpenMM's does not. cuPotential's cost per thousand atoms drops from 0.008 to 0.0055 GPU-hours as systems grow and then holds flat from 200k to 359k atoms. Stock OpenMM bottoms out around 0.010 in the middle of the range and climbs back to 0.013 for the largest systems. The large complexes, often multi-domain proteins or oligomeric assemblies, are where cuPotential saves the most, and they are also the most expensive systems to simulate.

Typical systems still save more than a third. Most of our dataset is smaller than the median system in this benchmark: the typical system in our dataset is around 37,000 atoms. At that size, a simulation costs $0.96 with stock OpenMM and $0.61 with cuPotential, a 37% reduction.

What this means for synthetic data

Holding the budget fixed, a 37-51% cost reduction translates into roughly 1.6-2× as many simulated complexes, or the same number of complexes with longer trajectories. Because the savings are largest on big systems, cuPotential also makes it cheaper to include the large, multi-chain complexes that we

would otherwise be tempted to skip on cost grounds, which reduces a size bias in the training data.

Looking Forward

Structure-based AI models have come a long way in a short time. The first generation were co-folding models: given a protein and a molecule, they predicted how the two fit together. Only recently have they grown into potency models, predicting not just where a molecule binds but how strongly. Now TerraBind is becoming a foundation for much more. Used as context for downstream predictions, it is helping us model cellular activity, read across assay panels, and anticipate efficacy in a broader biological setting.

Each step up that ladder demands more, and richer data. That's why these two results matter together. Faster model computation lets TerraBind train and run at the scale that protein-ligand complexes demand, and cheaper simulation lets us feed it far more physics at the same budget. In both cases the gains are largest on the big, complex problems that matter most in drug discovery.

We're excited to keep working with NVIDIA to write the next chapter of drug development, one driven by these game-changing advances in computational technology.