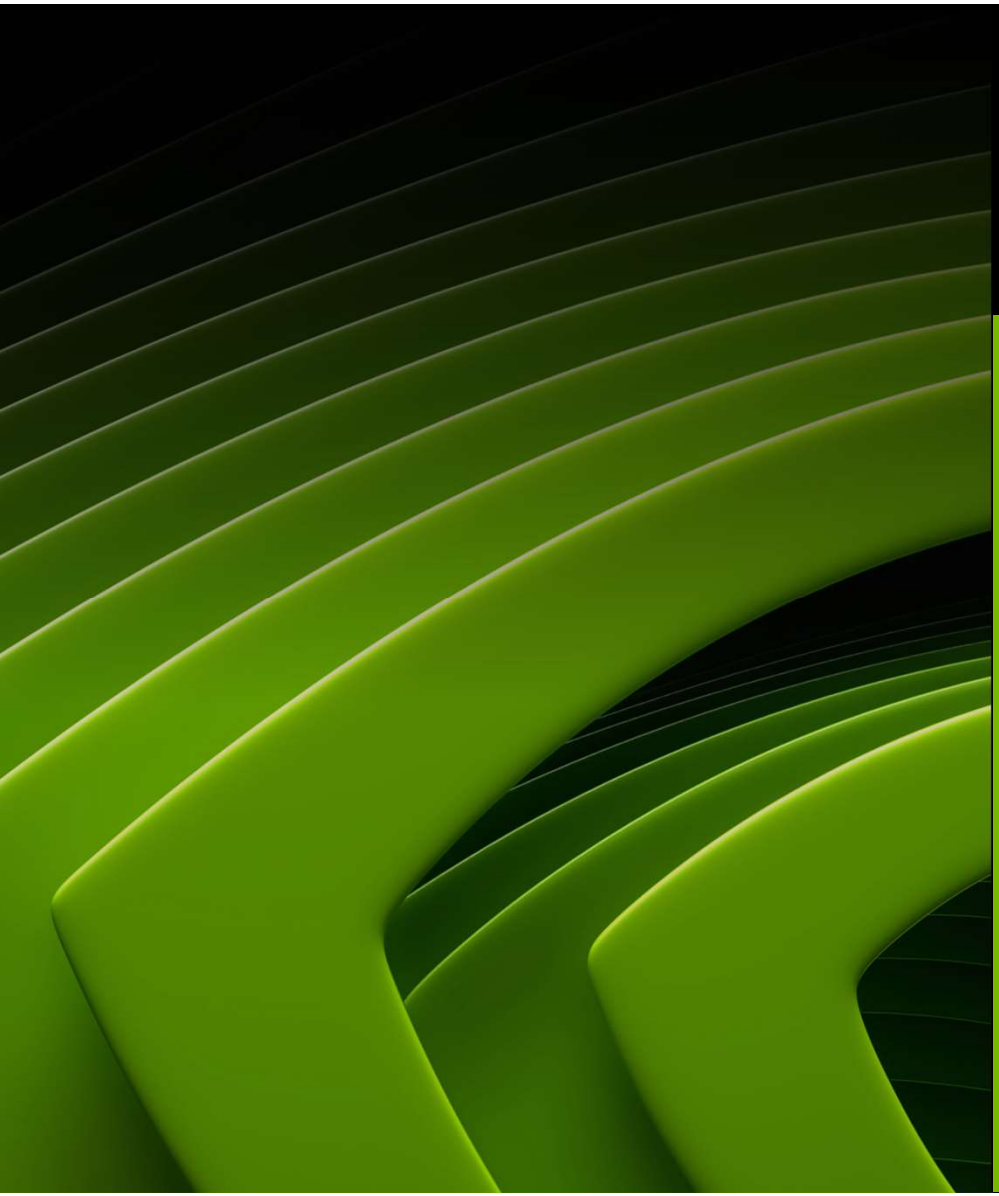
Advanced Networking & Storage For Generative AI

Kevin Deierling, NVIDIA | ADMS/Aug 2023

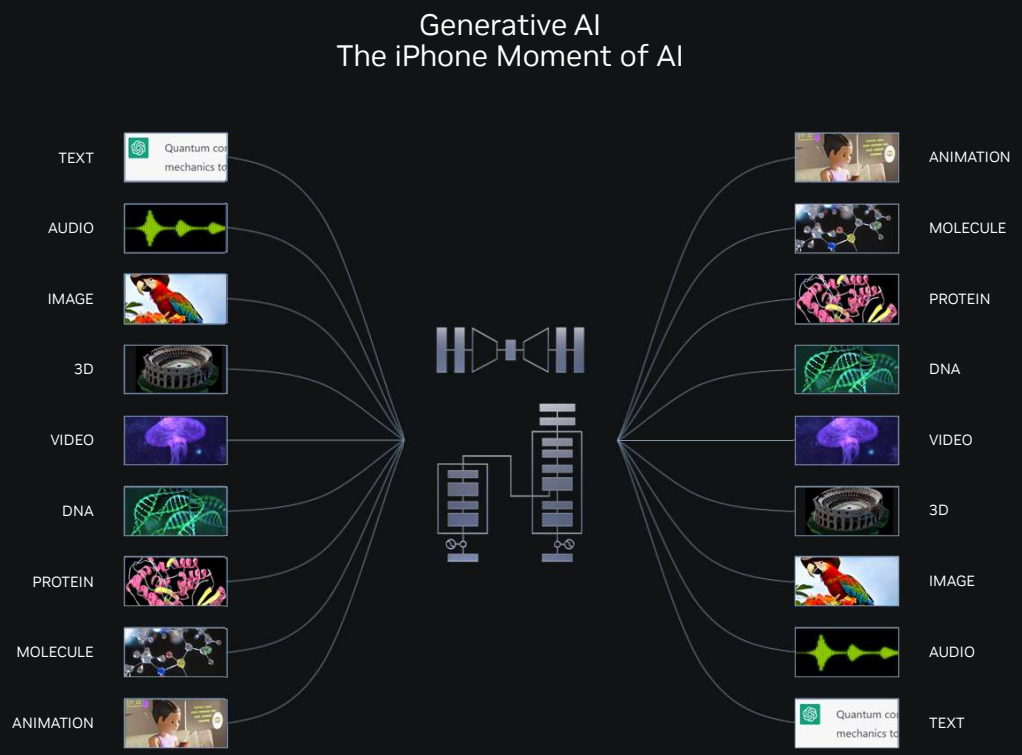


Agenda

- Natural language processing and semantic embeddings
- Attention and Generative Pre-trained Transformers (GPT)
- Vector databases and similarity search algorithms
- Implications for Networking and Storage for Generative AI

Large Language Models & Generative AI

Quantum leap to accelerate every business



What are Generative Large Language Models?

(Let's Ask an LLM)

Prompt: Explain what generative large language models are, what they can do, and the critical mechanisms that enable them to perform these tasks. Be succinct.

Generative large language models are AI models capable of understanding and generating human-like text. They utilize deep learning techniques, specifically transformer architectures, to process and generate language.

Key characteristics and mechanisms enabling their performance:

1. Transformer Architecture: The core mechanism that enables parallel processing of input text through self-attention mechanisms, allowing the model to learn contextual relationships between words and better understand context.
2. Pre-training and Fine-tuning: Models are first pre-trained on vast amounts of diverse text data to learn language patterns and general knowledge. Then, they are fine-tuned on specific tasks to adapt their knowledge to particular applications.
3. Generative Capability: These models can generate new text by sampling from learned probability distributions, enabling them to write creative stories, answer questions, or generate other forms of text.
4. Transfer Learning: Their pre-trained knowledge allows them to perform well with less task-specific training data, making them adaptable to various tasks and reducing the need for massive datasets.
5. Contextual Understanding: The models maintain memory of previous words in the sentence, enabling them to understand the context and generate coherent and contextually relevant responses.
6. Text Completion and Translation: By predicting the next word in a sentence, these models can automatically complete sentences or even translate text between different languages.

Overall, generative large language models have significantly advanced natural language processing tasks, including language understanding, text generation, translation, and even performing specific tasks like question answering, summarization, and chatbot interactions.

What are Generative Large Language Models?

(Let's Ask an LLM)

Prompt: Explain what generative large language models are, what they can do, and the critical mechanisms that enable them to perform these tasks. Be succinct.

Generative large language models are AI models capable of understanding and generating human-like text. They utilize deep learning techniques, specifically transformer architectures, to process and generate language.

Key characteristics and mechanisms enabling their performance:

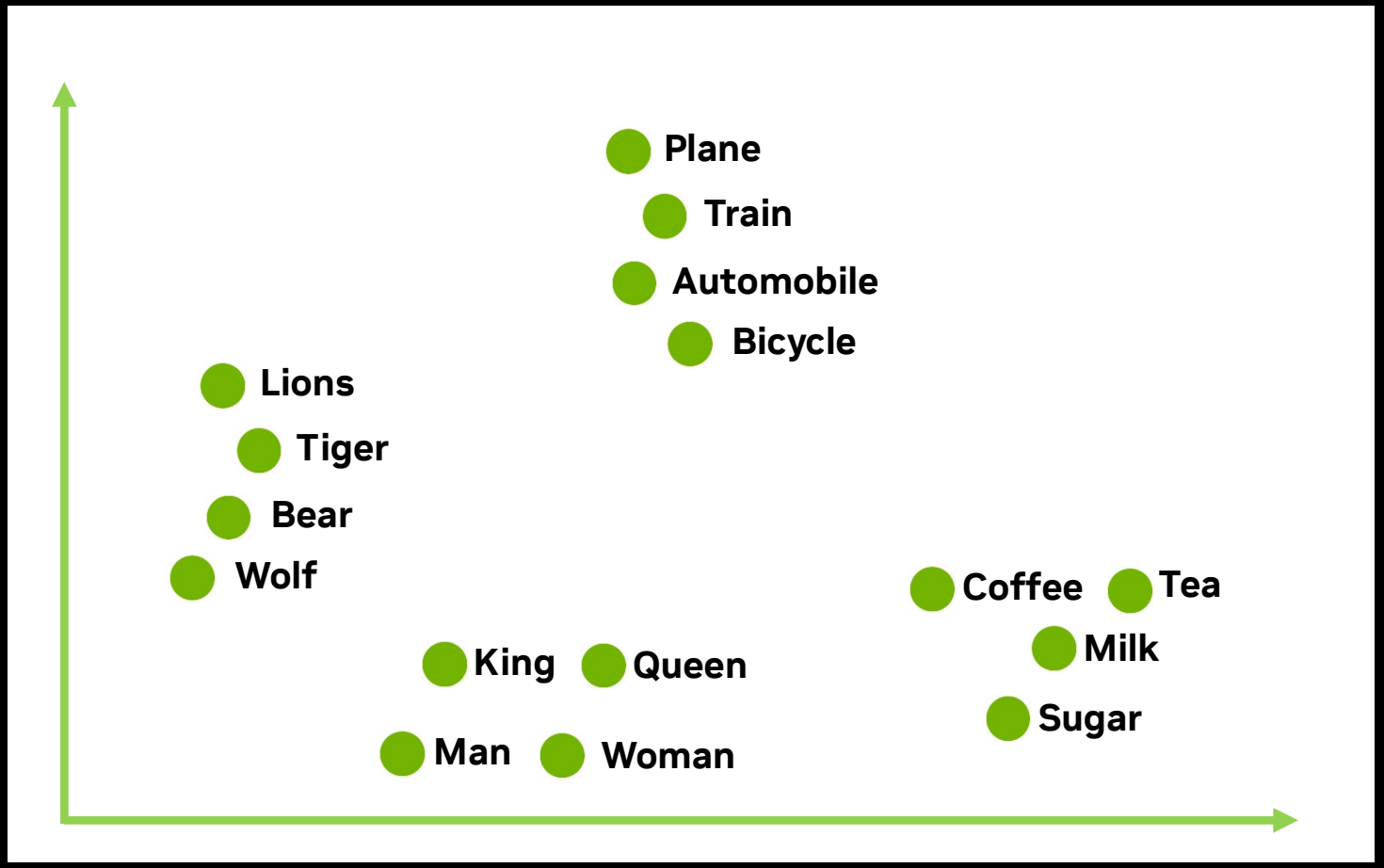
1. **Transformer Architecture: The core mechanism that enables parallel processing of input text through self-attention mechanisms, allowing the model to learn contextual relationships between words and better understand context.**
2. Pre-training and Fine-tuning: Models are first pre-trained on vast amounts of diverse text data to learn language patterns and general knowledge. Then, they are fine-tuned on specific tasks to adapt their knowledge to particular applications.
3. Generative Capability: These models can generate new text by sampling from learned probability distributions, enabling them to write creative stories, answer questions, or generate other forms of text.
4. Transfer Learning: Their pre-trained knowledge allows them to perform well with less task-specific training data, making them adaptable to various tasks and reducing the need for massive datasets.
5. **Contextual Understanding: The models maintain memory of previous words in the sentence, enabling them to understand the context and generate coherent and contextually relevant responses.**
6. Text Completion and Translation: By predicting the next word in a sentence, these models can automatically complete sentences or even translate text between different languages.

Overall, generative large language models have significantly advanced natural language processing tasks, including language understanding, text generation, translation, and even performing specific tasks like question answering, summarization, and chatbot interactions.

“You shall know a word by the company it keeps.”

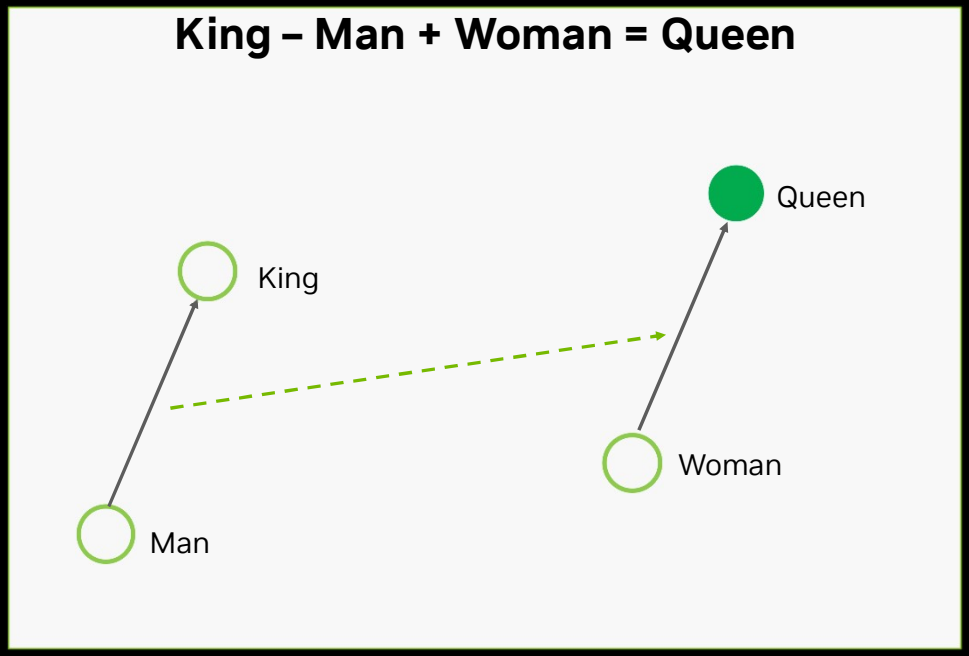
— John Rupert Firth, « A Synopsis of Linguistic Theory », 1957

Semantic Word Embeddings



Embeddings

Vector Encodings that Preserve Semantic & Syntactic Relationships & Thus Have Predictive Value



- Mikolov et al, 2013
  - Distributed Representation of Words and Phrases and their Compositionality
  - Efficient Estimation of Word Representations in Vector Space
- Skip-gram Word2Vec compute word embeddings: fundamental breakthrough in natural language processing

  Warsaw - Poland + Germany = Berlin

Why Such Huge Embedding Vector Space & Model Sizes?

Language, syntax, & grammar is vastly more complex than just words

Plurality & Countability

They We	⟷	are
She He	⟷	is
Water Sugar	⟷	some
Chickens Tractors	⟷	three

Tenses
(Pres/Past/Future) *
(Simple/Cont/Perf/Perf-Cont)

Yesterday	⟷	I farmed
Tomorrow	⟷	I will farm
Today	⟷	I have been farming

Cases
Infinitive, indicative, imperative, subjunctive

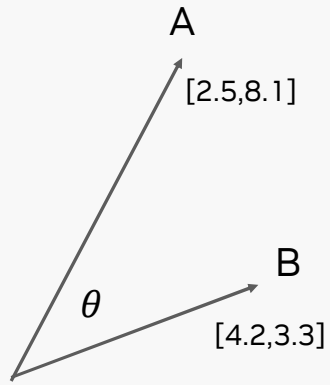
He likes	⟷	to sing
He	⟷	sings
John	⟷	empty the bin
suggests	⟷	that you sing

- Typical LLM model embeddings ~64 elements
  - English has ~100K words. If LLMs were only words - 6.4M parameters
  - LLMs are much, much larger than this: 100B's of parameters!!
- LLM embeddings are *NOT* simple word semantics
  - Embeddings include hidden neural network state connecting words, pairs of words, phrases, …
  - Grammar includes plurality, countability, tense, cases, and much more
  - And words and grammar are just the "tip of the iceberg" of language
    - idioms, symbolism, metaphors, rhymes, alliteration, irony, tone …

Similarity Within a Vector Space

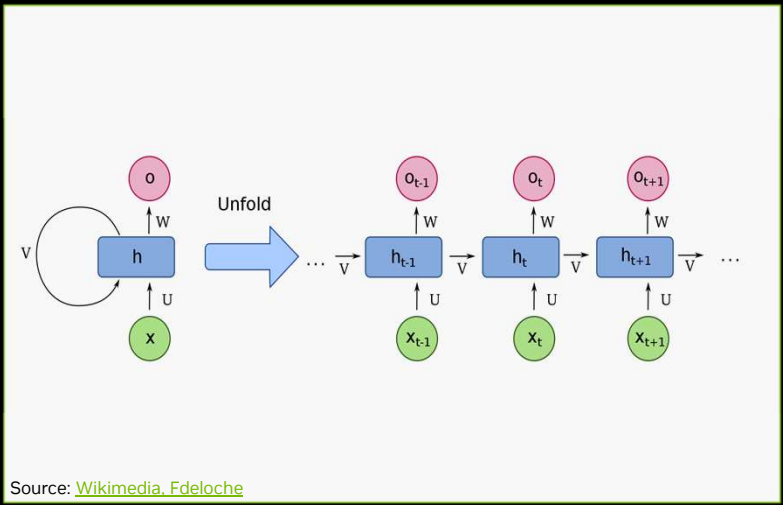
The dot product of two vectors is an indication of alignment of the vectors

$$A \bullet B = |A| * |B| * \cos(\theta) = A$$

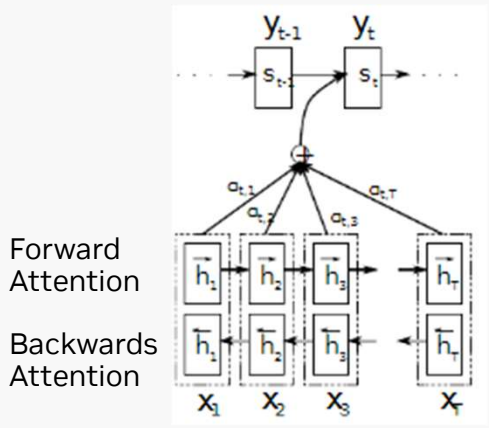


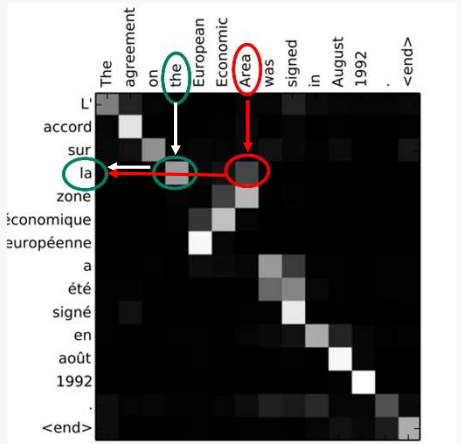
- Dot product of two vectors incorporates the magnitudes and the “similarity” of their direction
- Easy to visualize in 2 & 3 dimensions … but extends mathematically to N dimensions
- Mathematically vector dot products are performed as matrix multiplication of A and B: $A \bullet B^T$.
- Fortunately, GPUs are exceptionally good at performing layered, matrix multiplications of vectors of large dimensions

Recurrent Neural Networks + Attention



Source: Wikimedia, Fdeloche





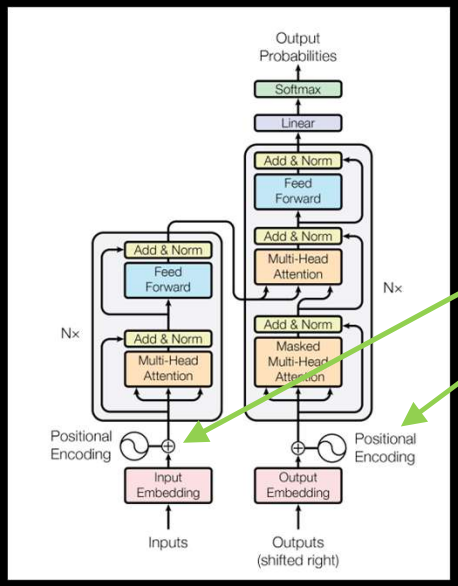
D Bahdanau: 2014, Neural Machine Translation by Jointly Learning to Align & Translate

- Recurrent neural networks (RNN) process text sequentially, because word order matters:
  - *"She only could understand RNNs"*
  - *"She could only understand RNNs"*
- RNN limitations:
  - Difficulty with long text sequences
  - Vanishing/Exploding gradient problem
  - Sequential pipeline inhibits parallelism

- Key ideas introduced:
  - Bidirectional encoder-decoder RNN pipeline
  - Conditional probability based on hidden context vectors
  - **Attention!!**
- Still suffers from RNN limitations: poor parallelism and connecting distant words

Transformers: Attention is All You Need

2017 Paper: The Big Bang Genesis of Generative AI Large Language Models

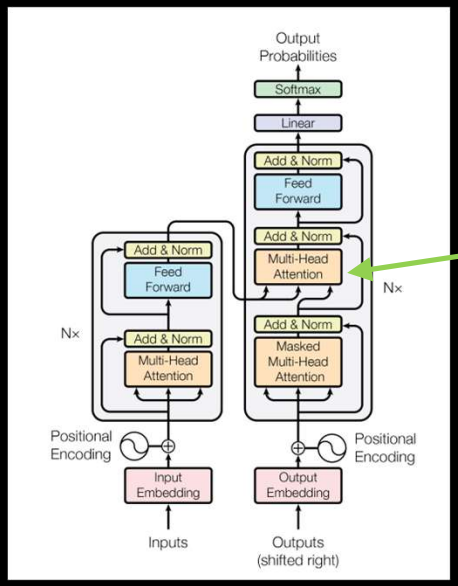


- Transformers: Four breakthroughs
  1. Positional Encoding – because word order matters:
     - Rather than word order being implied by sequential processing, it is explicitly encoded.

- Attention Is All You Need , Vaswani et al
- Transformer: Parallellizable encoder-decoder pipeline that uses multi-headed attention neural network, with positional enhanced encodings to look at all input tokens simultaneously, inferring meaning and relevance of words to each other …
- Eliminates recurrence to overcomes RNN problems

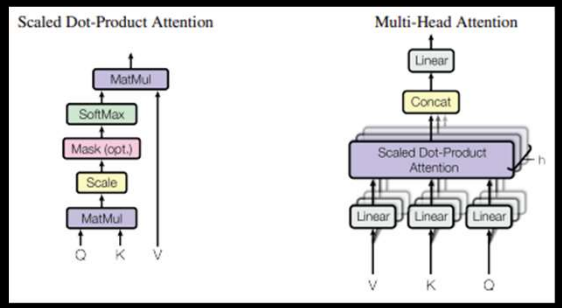
Transformers: Attention is All You Need

2017 Paper: The Big Bang Genesis of Generative AI Large Language Models



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

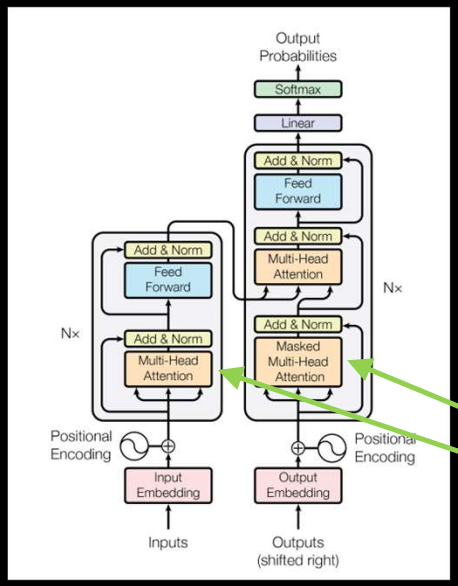
- Transformers: Four breakthroughs
  1. Positional Encoding – because word order matters:
     - Rather than word order being implied by sequential processing, it is explicitly encoded.
  2. Attention



- Attention Is All You Need , Vaswani et al
- Transformer: Parallellizable encoder-decoder pipeline that uses multi-headed attention neural network, with positional enhanced encodings to look at all input tokens simultaneously, inferring meaning and relevance of words to each other …
- Eliminates recurrence to overcomes RNN problems

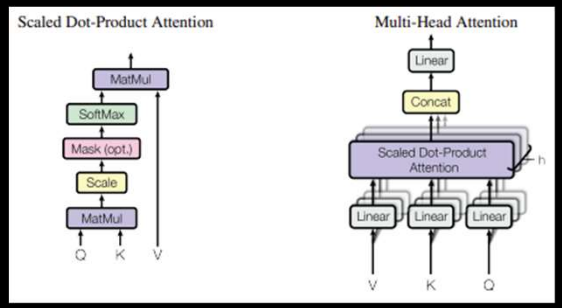
Transformers: Attention is All You Need

2017 Paper: The Big Bang Genesis of Generative AI Large Language Models



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

- Transformers: Four breakthroughs
  1. Positional Encoding – because word order matters:
     - Rather than word order being implied by sequential processing, it is explicitly encoded.
  2. Attention

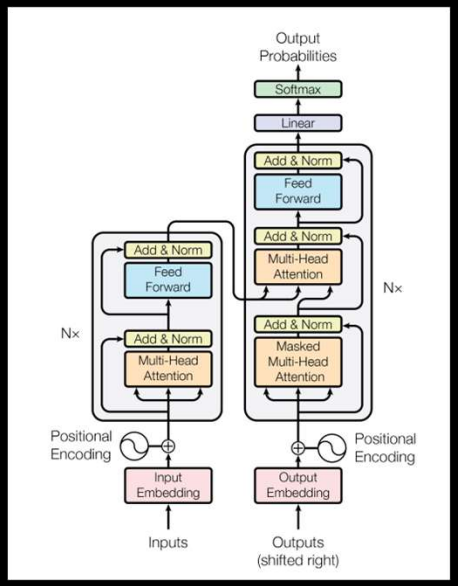


  3. Self-Attention
     - Synthesize language and meaning
       "Server, can I have the check?"
       "Looks like I just crashed the server."
     - Resolves complex language structures such as anaphora:
       *"She poured the pitcher of water into the glass, until it was **empty**."*
       *"She poured the pitcher of water into the glass, until it was **full**."*

- Attention Is All You Need , Vaswani et al
- Transformer: Parallellizable encoder-decoder pipeline that uses multi-headed attention neural network, with positional enhanced encodings to look at all input tokens simultaneously, inferring meaning and relevance of words to each other …
- Eliminates recurrence to overcomes RNN problems

Transformers: Attention is All You Need

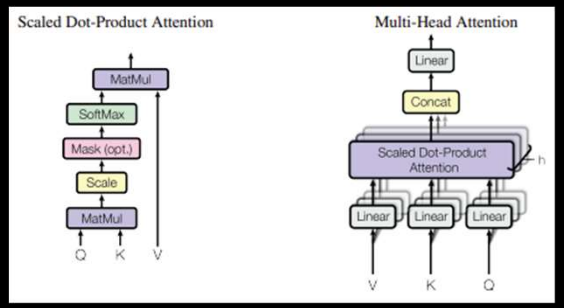
2017 Paper: The Big Bang Genesis of Generative AI Large Language Models



- Attention Is All You Need , Vaswani et al
- Transformer: Parallellizable encoder-decoder pipeline that uses multi-headed attention neural network, with positional enhanced encodings to look at all input tokens simultaneously, inferring meaning and relevance of words to each other …
- Eliminates recurrence to overcomes RNN problems

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

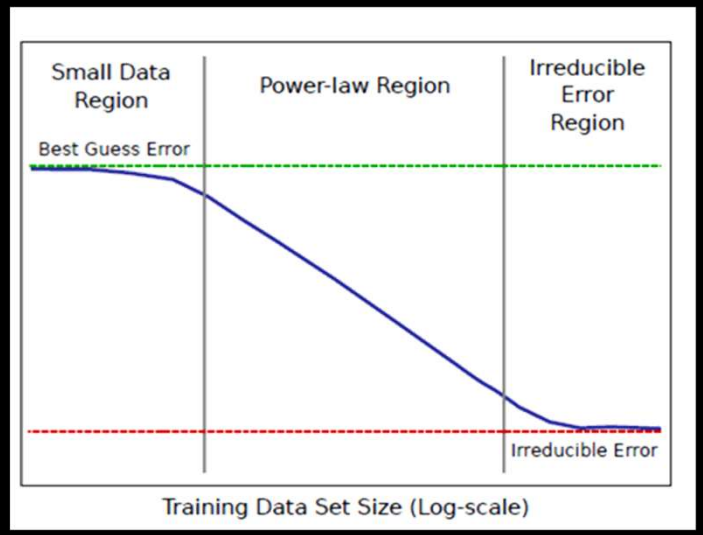
- Transformers: Four breakthroughs
  1. Positional Encoding – because word order matters:
     - Rather than word order being implied by sequential processing, it is explicitly encoded.
  2. Attention



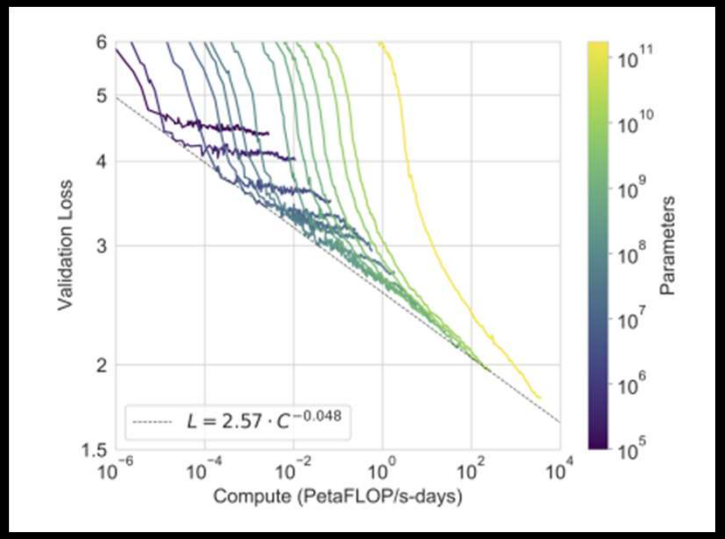
  3. Self-Attention
     - Synthesize language and meaning
       "Server, can I have the check?"
       "Looks like I just crashed the server."
     - Resolves complex language structures such as anaphora:
       *"She poured the pitcher of water into the glass, until it was **empty**."*
       *"She poured the pitcher of water into the glass, until it was **full**."*
  4. Training speed
     - *".. the Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers."*

AI Brings a New Power Law

Exponential AI Model Scale Improves Results



Deep Learning Scaling is Predictable, Empirically .Hestness et al

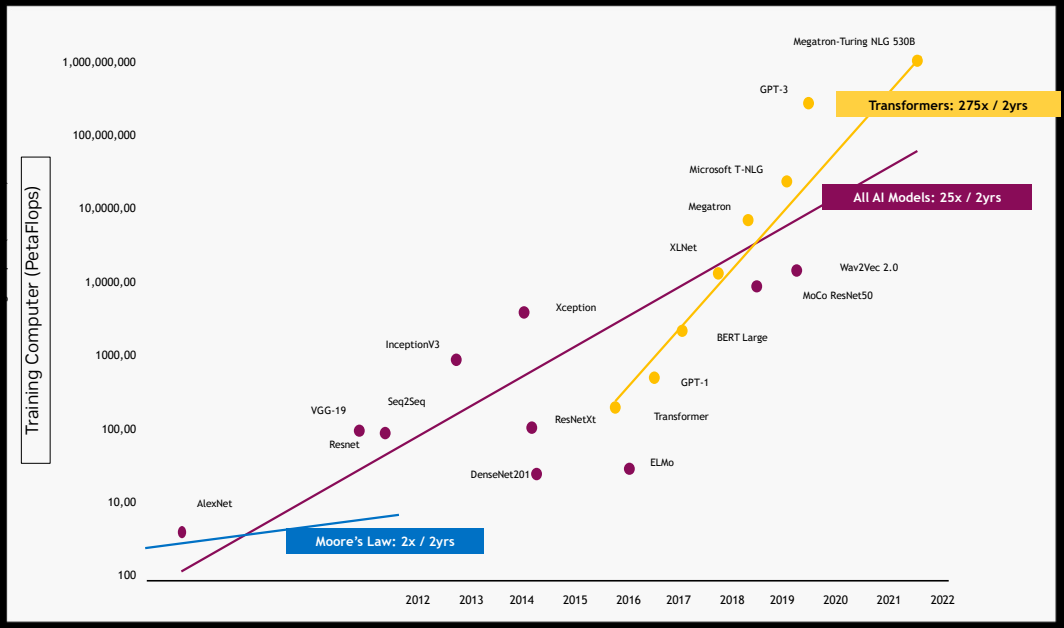


Language Models are Few Shot Learners , Brown et al

- Across a broad range of workloads model accuracy improves logarithmically with scale

Attention (and Scale) is All You Need

AI Model Scale Dramatically Improves Results



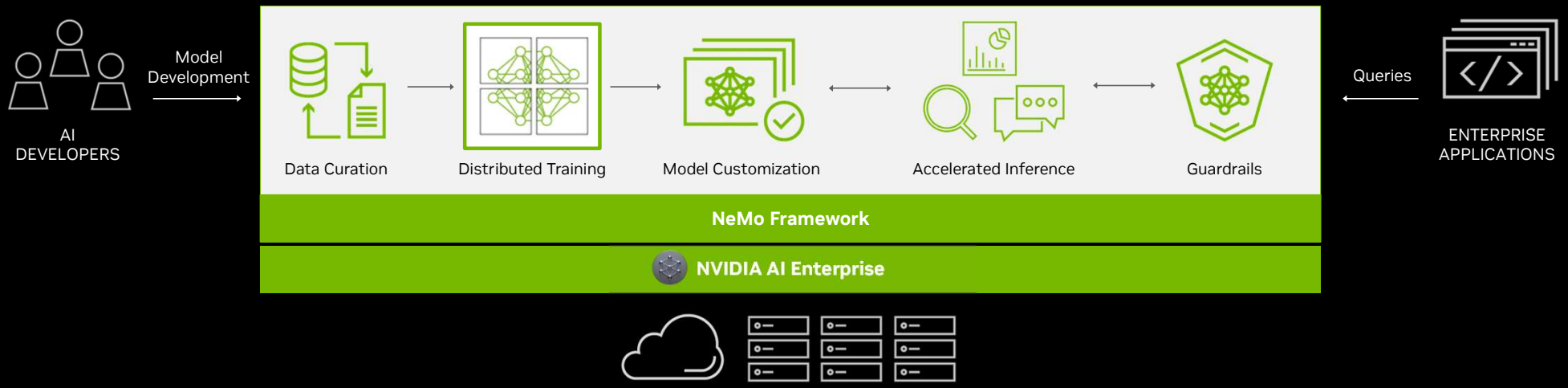
NVIDIA Developer Megatron Blog

The Data Center is the New Unit of Computing

- Compute requirements for LLM training increasing exponentially
- Models with billions of parameters require massive unsupervised training
- AI scale computing requires optimization across the entire data center stack

NVIDIA NeMo Framework for Custom LLMs

End-to-end, cloud-native framework to build, customize and deploy generative AI models



Multi-Modality

Build language, image, generative AI models

Data Curation at Scale

Extract, deduplicate, filter info from large unstructured data @ scale

Optimized Training

Accelerate training and throughput by parallelizing the model and the training data across 1,000s of nodes.

Model Customization

Easily customize with P-tuning, SFT, Adapters, RLHF, AliBi

Deploy at Scale

Run optimized inference at-scale anywhere

Guardrails

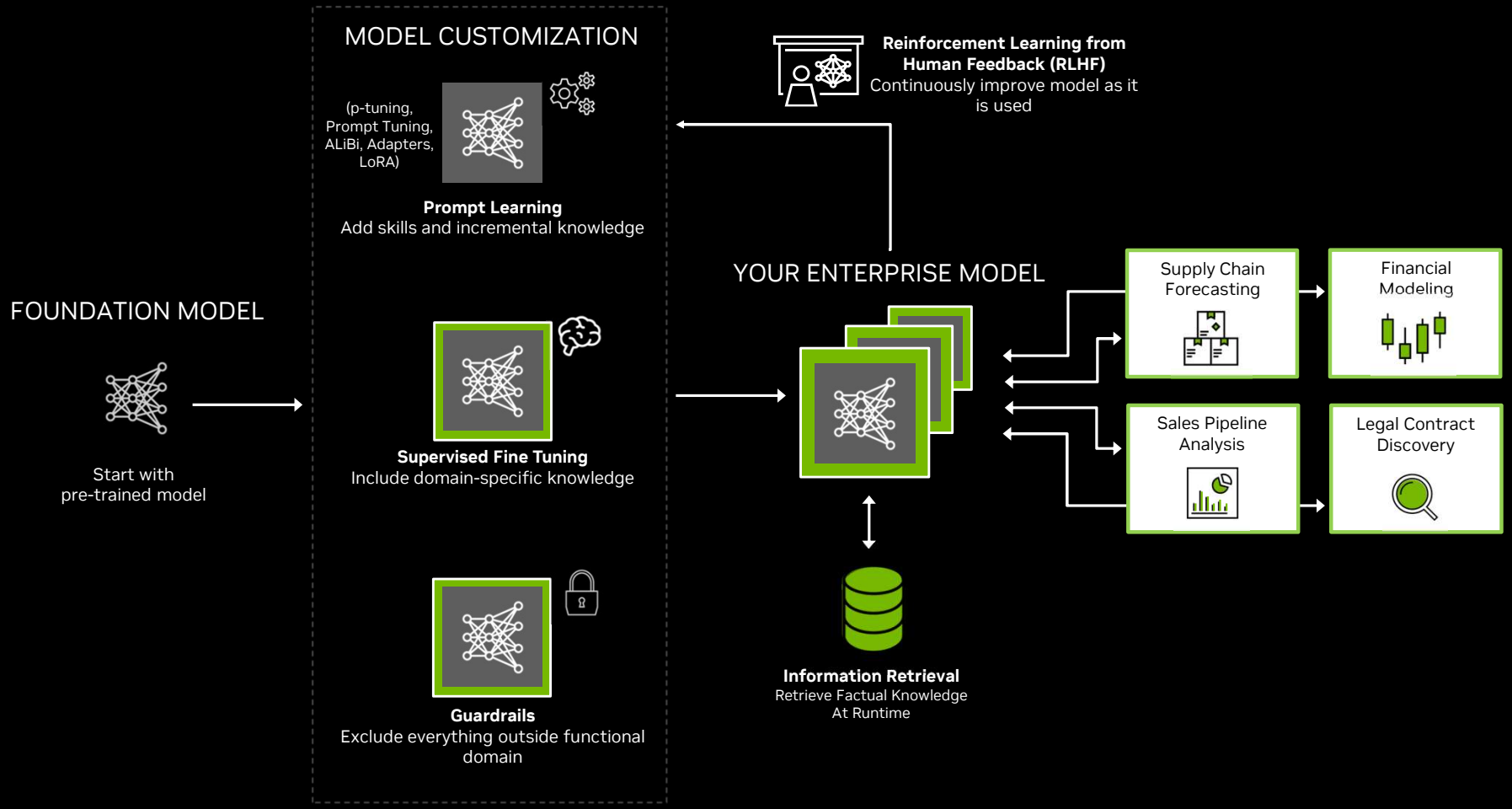
Keep applications aligned with safety and security requirements using NeMo Guardrails

Support

NVIDIA AI Enterprise and experts by your side to keep projects on track

Model Customization for Enterprise Ready LLMs

Customization techniques to overcome the challenges of using foundation models



Customization Enables the Best Responses

Zero-Shot Response

"What is the yellow part in an egg?"

LLM

"This is the part that suspended in the center of the egg."

P-Tuned Response

"What is the yellow part in an egg?"

+

Trained Prompts (Context)

LLM

Nutrition Chatbot

"The yellow part in an egg is the yolk. It contains fat, cholesterol, and protein."

Healthcare Chatbot

"The yellow part in an egg is rich in choline, which is important for fetal brain development"

Culinary Chatbot

"The yellow part in an egg is used to fortify sauces and salad dressings, and to emulsify rich, fatty, ingredients like oil and butter"

Vector Databases

What are Vector Databases?

(Let's Ask an LLM)

Prompt: Explain what a vector database is, how it differs from a traditional database, what it is useful for, and the critical mechanisms that enable it to perform these tasks. Be succinct.

A vector database is a specialized storage system designed to efficiently store and retrieve vectors (numerical representations of data). Unlike traditional databases that primarily store structured data in tables, vector databases focus on optimizing similarity search based on vector distance metrics.

Useful for:

1. Similarity Search: Quickly finding similar vectors based on distance metrics (e.g., nearest neighbors search).
2. Machine Learning: Storing vector representations for tasks like recommendation systems, image recognition, and natural language processing.

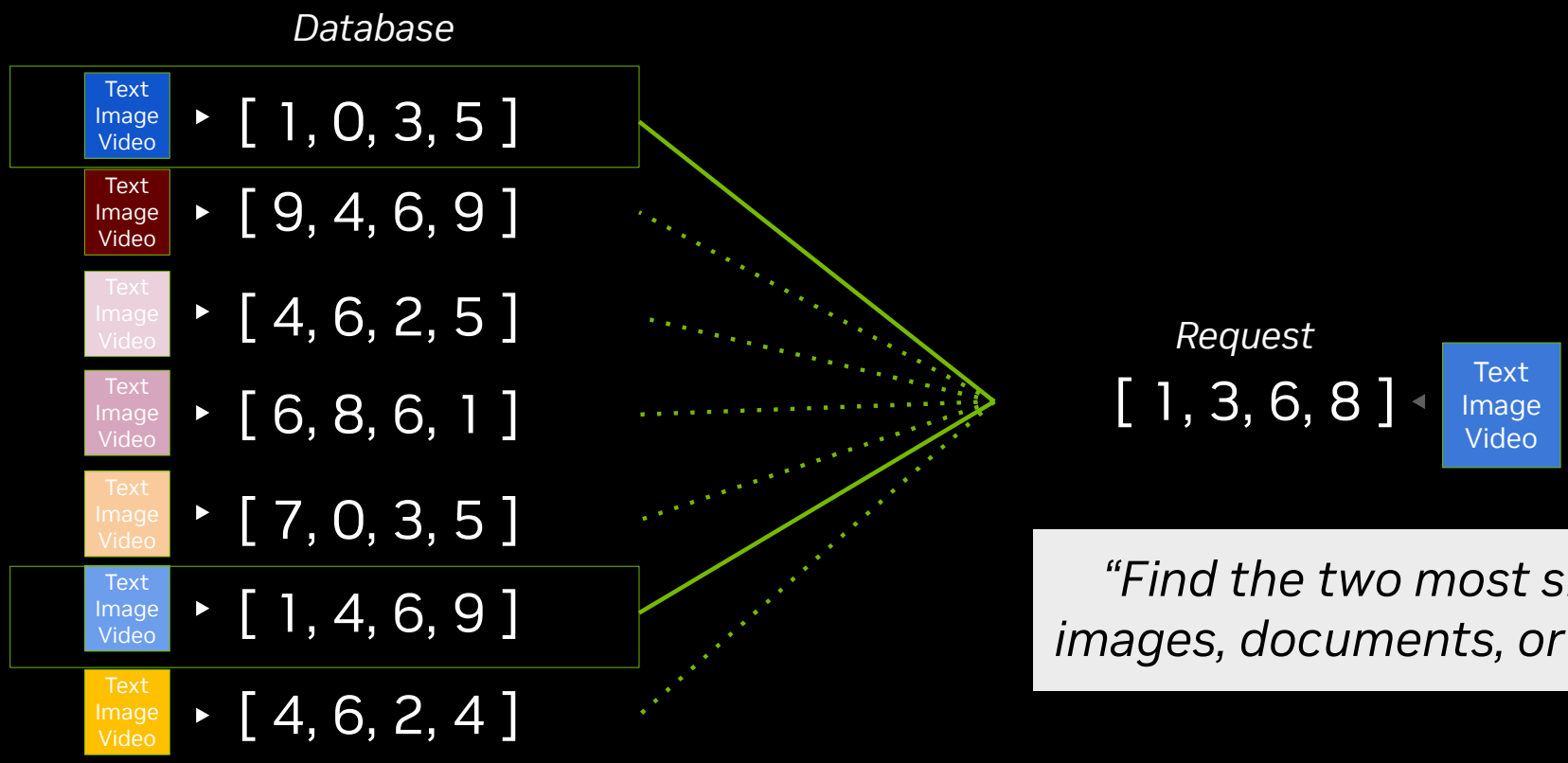
Critical mechanisms enabling its performance:

1. Indexing Techniques: Efficient data structures (e.g., KD-trees, Ball trees) organize vectors for fast search based on proximity, reducing computational complexity.
2. Distance Metrics: The database uses distance functions (e.g., Euclidean, cosine similarity) to measure similarity between vectors, enabling effective retrieval of similar items.
3. Dimension Reduction: Techniques like Principal Component Analysis (PCA) or Locality-Sensitive Hashing (LSH) reduce vector dimensions, accelerating search and storage.
4. Data Compression: Vector quantization or binary codes can be employed to reduce storage requirements while preserving essential information.
5. Parallel Processing: Leveraging parallel computing techniques allows for faster querying and retrieval of vectors from the database.

In summary, a vector database is a specialized system optimized for similarity search and storage of vector representations, differing from traditional databases in its focus and mechanisms. It is valuable for applications involving similarity-based retrieval and machine learning tasks that rely on vector representations.

Vector Similarity Search Overview

Measuring similarity and retrieving relevant embeddings

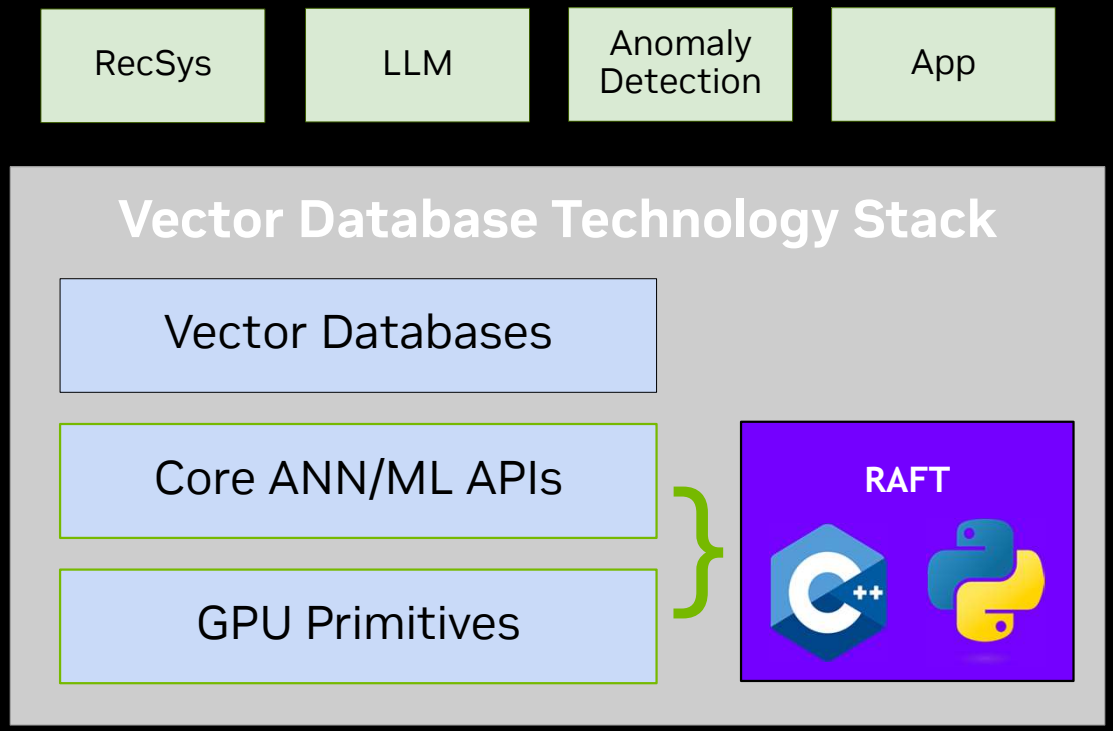


"Find the two most similar images, documents, or videos"

RAPIDS RAFT Overview

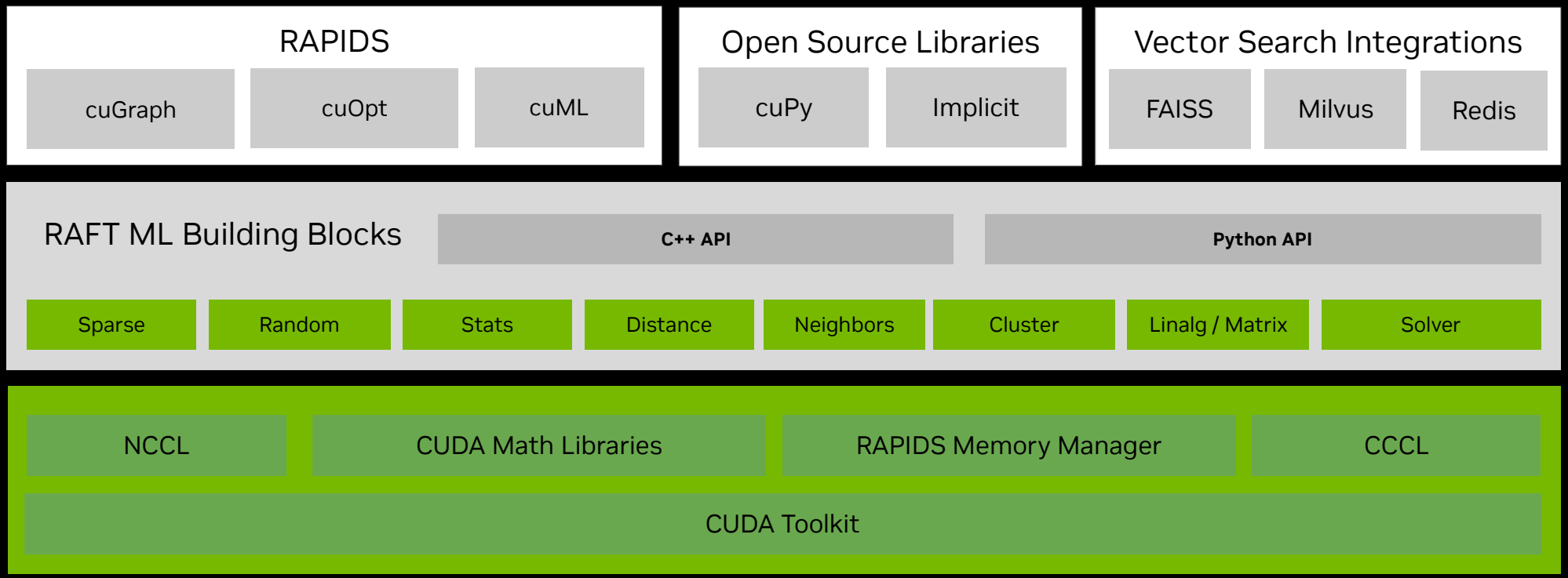
Toolbox of Accelerated, Composable Building Blocks for ML & Data Analytics

- RAFT contains ready-to-use APIs and composable building blocks
  - Sparse and dense matrix operations, nearest neighbors, clustering, iterative solvers, and more...
- Fastest Approximate and Exact Nearest Neighbors
  - Core ANN APIs: IVF-PQ, IVF-Flat
- Friendly Consistent C++17 and Python APIs with a header-only library and Apache 2.0 license



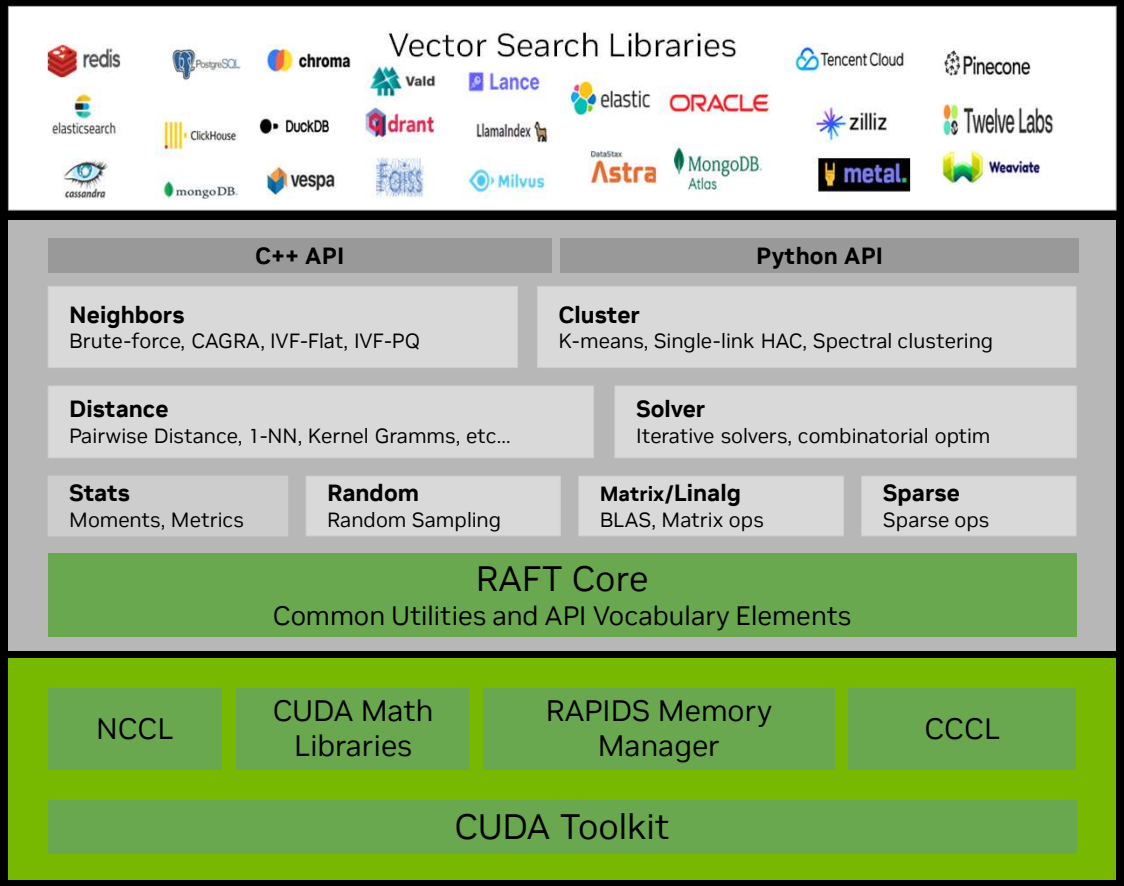
RAPIDS RAFT Overview

Toolbox of Accelerated, Composable Building Blocks for ML & Data Analytics



RAPIDS RAFT Overview

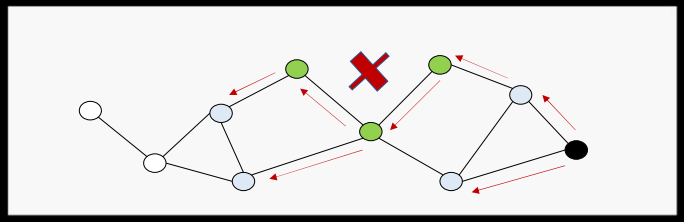
Toolbox of Accelerated, Composable Building Blocks for ML & Data Analytics

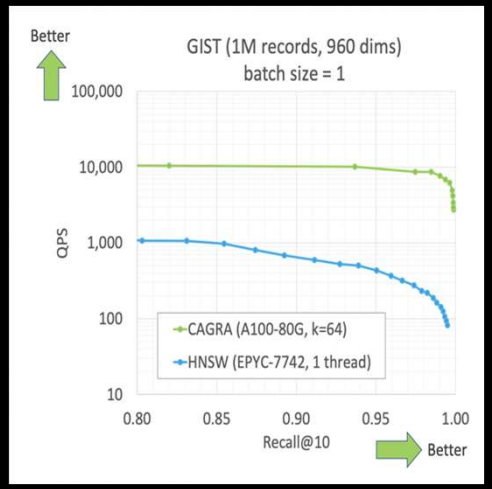


CAGRA

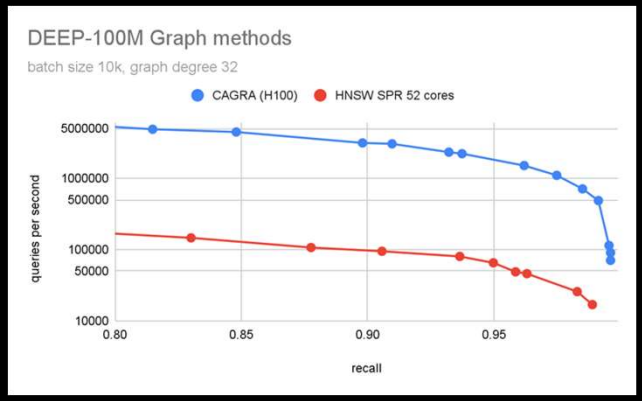
GPU-Accelerated State-of-the-Art Graph-Based ANN

- GPU-native algorithm similar to HNSW for CPU
- Setting records for both single query and large batch performance
- Higher throughput than existing GPU Graph ANNs and lower latency than SOTA CPU Graph ANNs
- Experimental implementation now available in RAFT (docs)





Note: Comparing against single thread because CPU HNSW only uses one thread at batch size 1

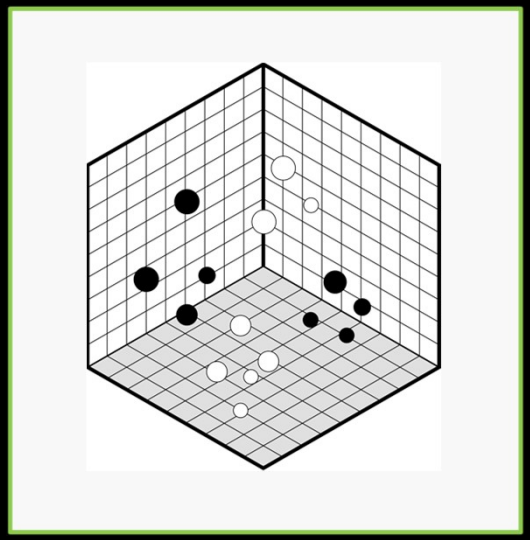


Vector Similarity Search Integrations Timeline

RAFT's ANN APIs are empowering the ecosystem

Milvus

Available now in Beta



redis

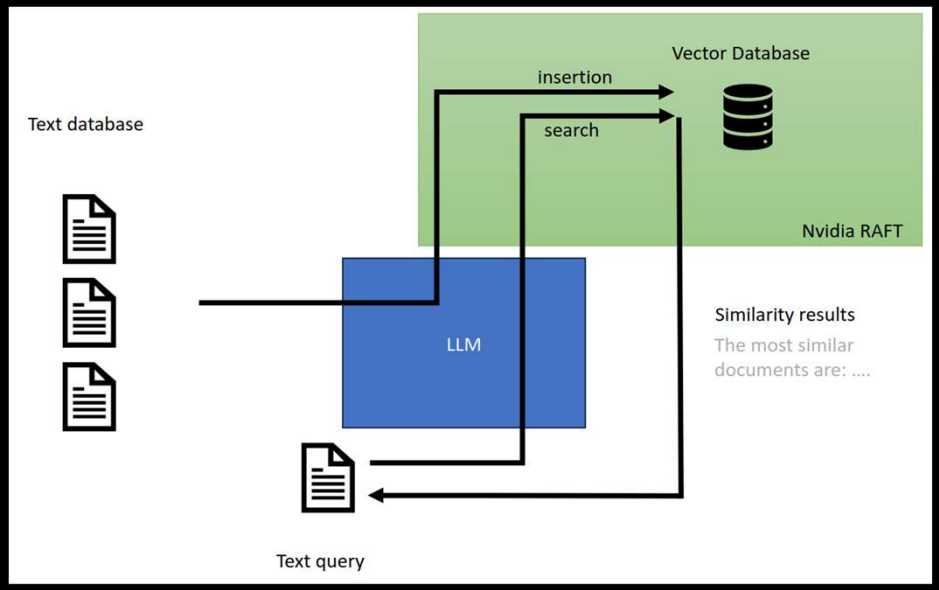
Target: Fall 2023

Target: Summer 2023

FAISS

RAG: Retrieval Augmented Generation

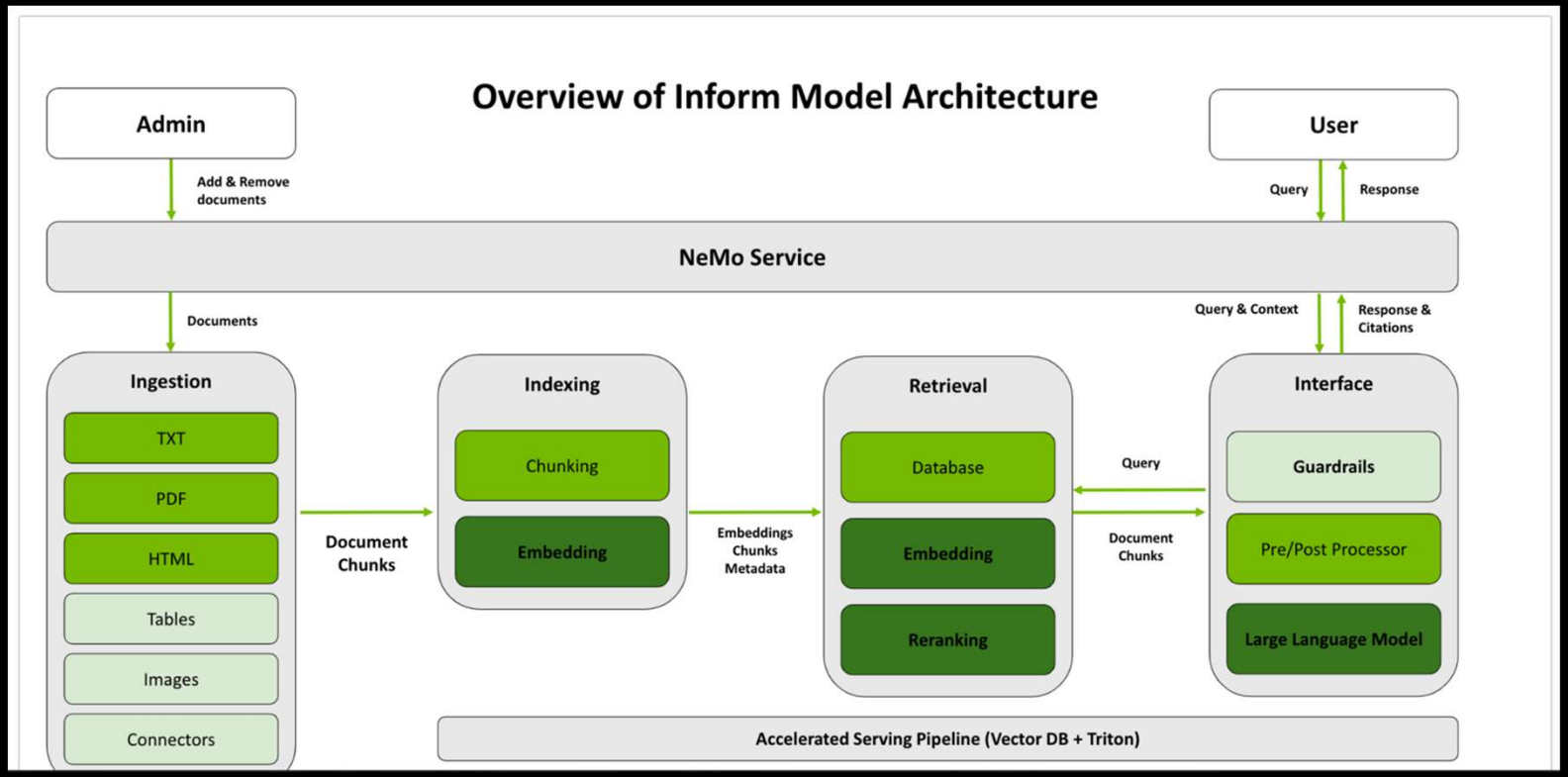
Customize pre-trained models with proprietary data



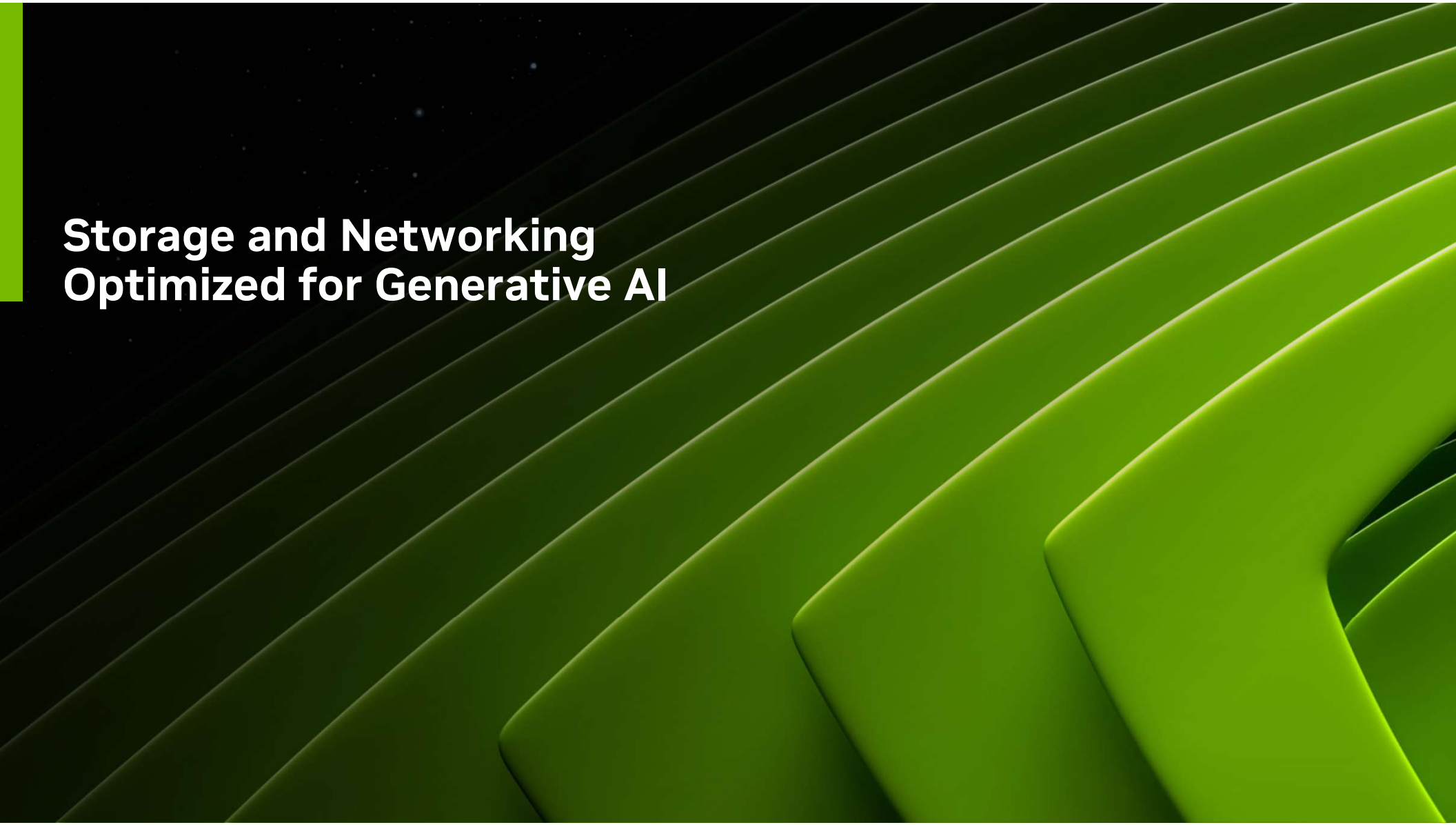
- Encoded LLM data stored in VectorDB
- Encoded queries augment similarity search of VectorDB

Information Retrieval Augmented Generation

Fine-tune both the Retriever AND the Generator



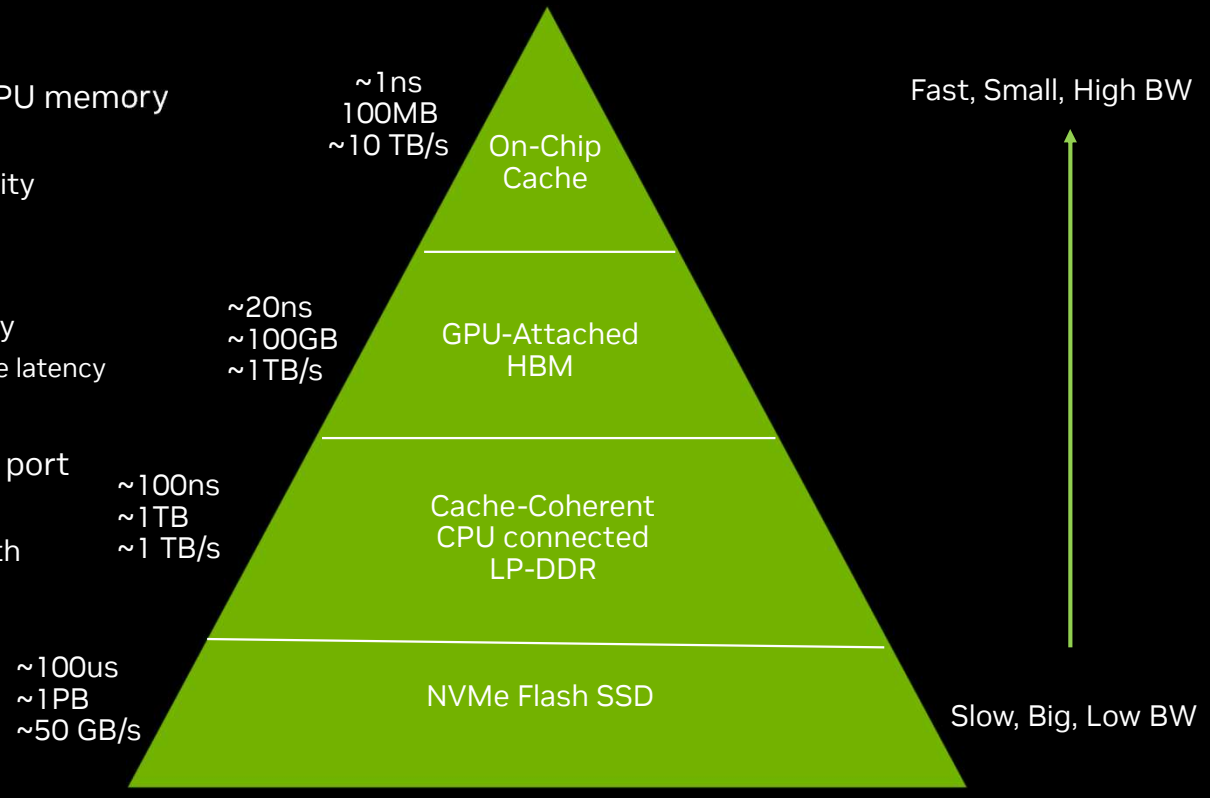
Storage and Networking Optimized for Generative AI



Generative AI is Data Intensive

Classic Memory Hierarchy Considerations Apply

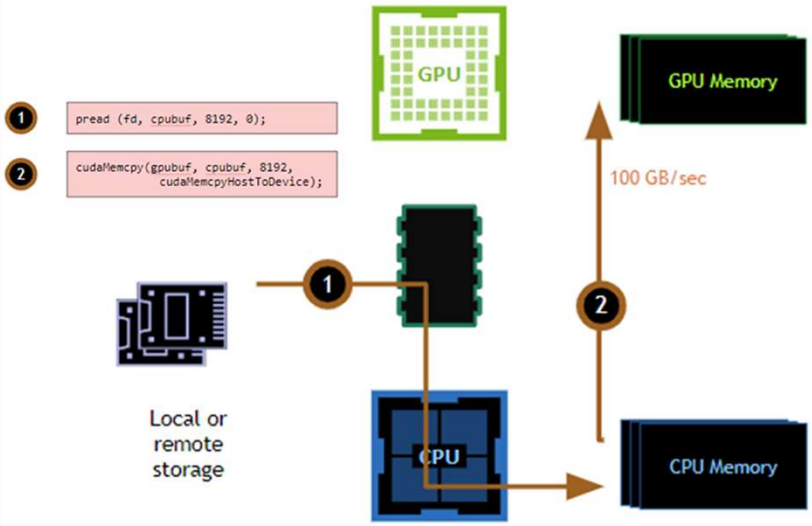
- LLM's are huge! 500B - 1T+ parameters
- Highest performance when entire model is in GPU memory
  - Clusters with 1000's GPUs only for hyperscalers
  - Train in the cloud with monster GPU-Mem capacity
- Classic memory hierarchy problem:
  - AI workloads have limited temporal locality
  - Gen-AI workloads have significant 'spatial' locality
    - Predictable accesses means pre-fetching can hide latency
  - Bandwidth is all you need!
- Mem BW ~1TB/s while networking ~50GB/s per port
  - 800Gbit/s ports (100GB/s) coming soon
  - Each GPU needs dedicated networking bandwidth
- Collective offload performs data tapering
  - Reduces data bandwidth requirements



NVIDIA Magnum IO GPUDirect™ Storage overview

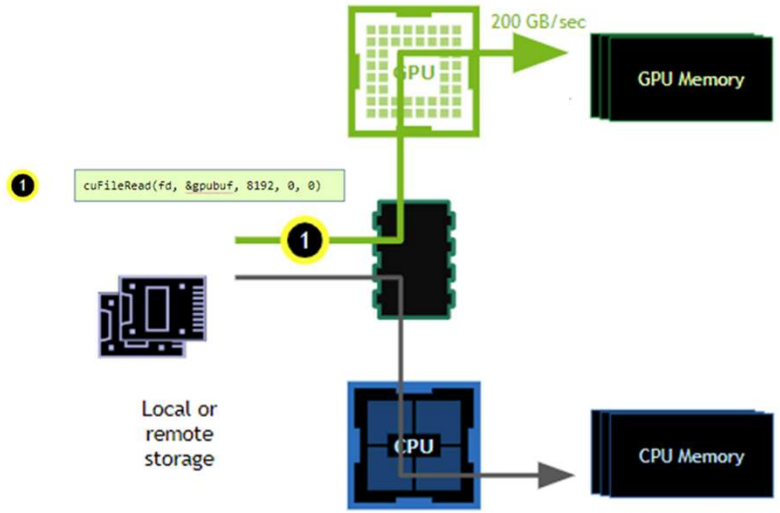
GPUDirect Storage adds File IO as part of CUDA

Traditional File System access



- 2 DMA operations
- Requires bounce buffers
- Halves CPU ⇔ PCI bandwidth

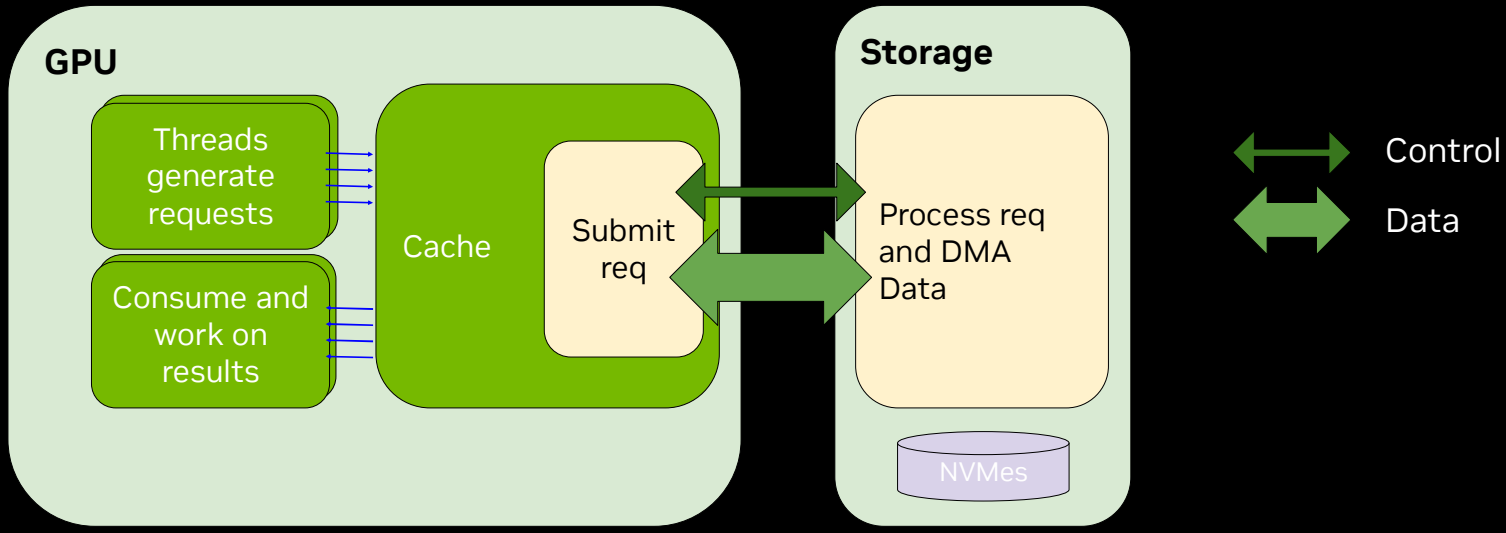
GPUDirect Storage access



- Single DMA operation
- No bounce buffer
- Independent of CPU ⇔ PCI
- Similar APIs as Posix
  - cuFileRead
  - cuFileWrite

GPU Initiated I/O Architecture

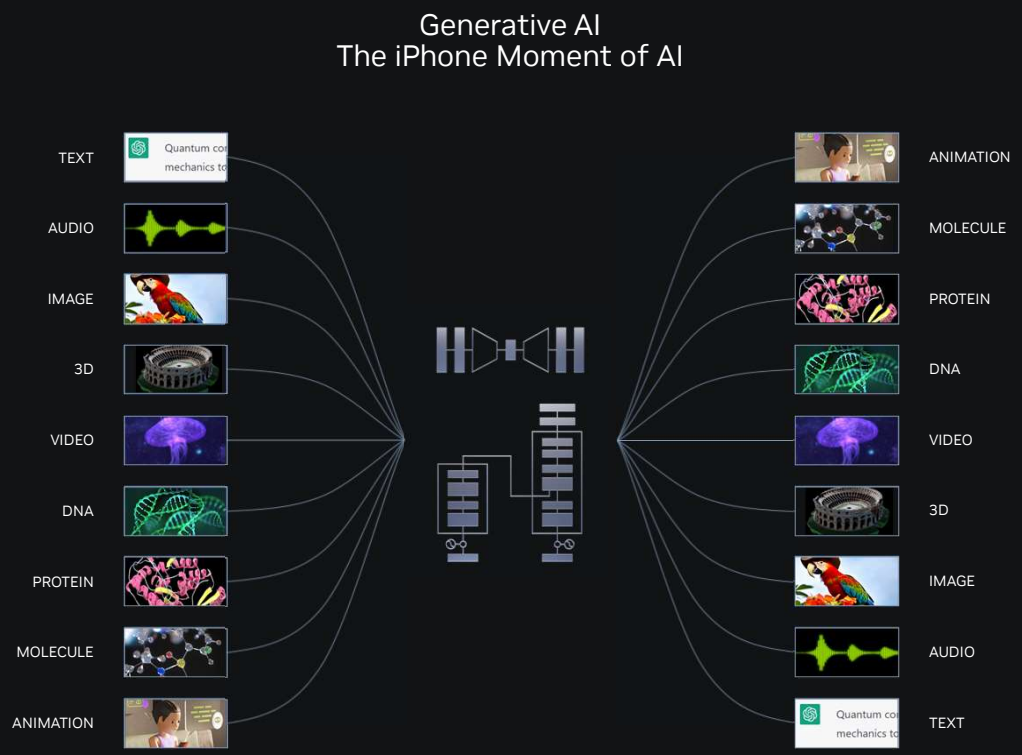
Eliminate CPU Bottleneck for Storage



- Often CPU has limited value in AI data processing
- In such cases moving both control and data path to GPU makes sense
  - Request, initiation, service, consumption all happen on the GPU
- GPU initiated networking & storage enables IO accesses that are initiated and triggered by GPU

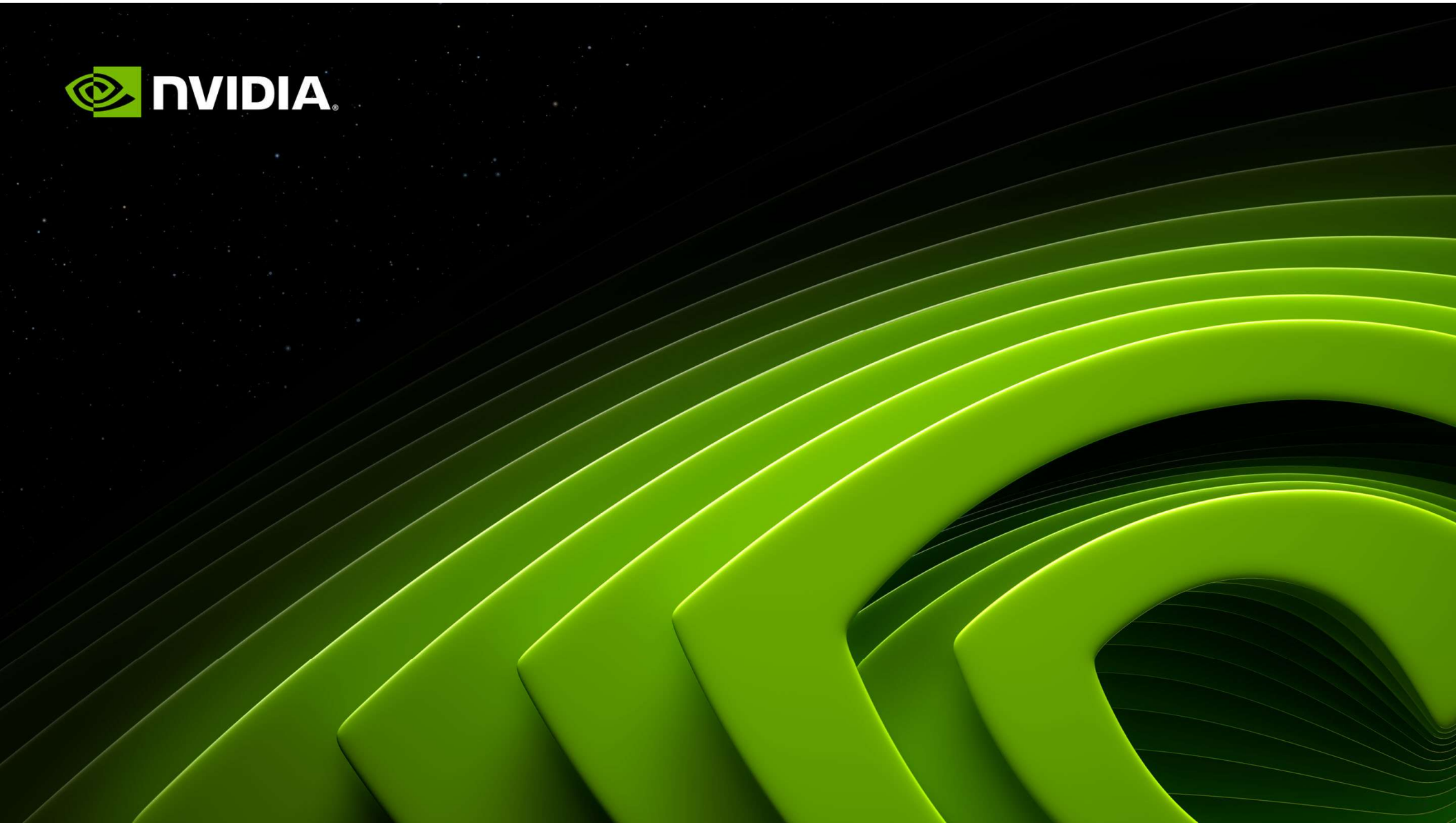
Call to Action

Get started with Gen AI



Learn More!

- Sources & Syllabus
- Transformers, Explained: Understand the Model Behind GPT-3, BERT, and T5 (daleonai.com)
- The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time. (jalammar.github.io)
- neural networks - What exactly are keys, queries, and values in attention mechanisms? - Cross Validated (stackexchange.com)
- Transformers Explained Visually (Part 2): How it works, step-by-step | by Ketan Doshi | Towards Data Science
- Neural Machine Translation by Jointly Learning to Align and Translate
- Attention Is All You Need
- Nearest Neighbor Indexes for Similarity Search | Pinecone


NVIDIA