



Large Language Models in High Energy Physics (succinct survey) and directions of future developments

Reporter: Andrey Ye. Shevel

Co-authors: Alexey Naikov, Anatoly Oreshkin, Alexey Shvetsov

Outline

- Large Language Model & Large Reasoning Model (LLM/LRM)
- The cost of training
- Retrieval Augmented Generation (RAG) and domain specifics
- Example of LLM/LRM in computing system
- Possible future developments

LLM/LRM

- A language is a system of signs having meaning by convention. A productive language can formulate an unlimited variety of sentences.
- LLMs can perform a wide range of natural language processing (NLP) tasks, such as text generation, summarization, translation, and question answering, thanks to their training on massive and diverse datasets often comprising billions or trillions of words.
- LLM reasoning refers to the ability of LLMs to process, **understand**, and make logical connections between various pieces of information to *generate coherent and contextually relevant outputs*. They mentioned as Large Reasoning Model (LRM). Further I'll tell LLM for both LLM & LRM.
 - “In the context of large language models (LLMs), **understanding** refers to the model's ability to produce the most relevant outcome of words for a given prompt”
- Key characteristics of LLMs include:
 - **Scale**: They have hundreds of millions to hundreds of billions of parameters, enabling them to capture complex patterns in language.
 - **Capabilities**: like mentioned above.
 - **Training**: like mentioned above.
 - **Foundation Models**: Many LLMs serve as "foundation models", which can be fine-tuned for specific applications or domains.
 - **Examples**: OpenAI's GPT, Google's BERT and Gemini, Meta's Llama, Deepseek, Alibaba qwen, Anthropic Claude series.
 - LLM uses many methods: Statistics, Rule based, Natural Language Processing (NLP), Linguistics, etc

Estimated shipments of Nvidia GPU by customers

Nvidia H100 GPU Shipments by Customer

Estimated 2023 H100 shipments by end customer.

Omdia estimates Nvidia sold ~500k A100 and H100 GPUs in Q3, and lead time for H100-based servers is up to 52 weeks.



Source: Omdia Research



https://regmedia.co.uk/2024/12/20/nvidia_omdia_estimates_2024.png

https://www.reddit.com/r/singularity/comments/1890o9y/nvidia_gpu_shipments_by_customer/

Popular LLMs

LLMs in 2025

LLM	Developer	Multimodal?	Reasoning?	Access
GPT-4o	OpenAI	Yes	No	Chatbot and API
o3 and o1	OpenAI	No	Yes	Chatbot and API
Gemini	Google	Yes	No	Chatbot and API
Gemma	Google	No	No	Open
Llama	Meta	No	No	Chatbot and open
R1	DeepSeek	No	Yes	Chatbot, API, and open
V3	DeepSeek	No	No	Chatbot, API, and open
Claude	Anthropic	Yes	Yes	Chatbot and API
Command	Cohere	No	No	API
Nova	Amazon	Yes	No	API
Large 2	Mistral AI	Yes (Pixtral)	No	API
Qwen	Alibaba Cloud	Yes (Qwen2.5-VL)	No	Chatbot, API, and open
Phi	Microsoft	No	No	Open
Grok	xAI	No	Yes	Chatbot and open

From «HEPiX Technology Watch Working Group Report» <https://indico.cern.ch/event/1477299/contributions/6373271/>

LLMs in High Energy Physics (HEP)

- Literature Search & Summarization
 - Speed up review of vast HEP literature (arXiv, CERN, INSPIRE-HEP)
 - Auto-summarize papers, highlight key equations/concepts
- Code Generation
 - Generate/optimize code for Monte Carlo simulations, detector analysis
 - Assist in ROOT, GEANT4, or Python scripting
- Data Analysis Assistant
 - Help to build ML models for particle classification or anomaly detection
 - Improve workflow and reproducibility
- Monitoring and controlling complex components
 - Large detectors
 - Accelerators
- Symbolic Reasoning & Theoretical Aid
 - Suggest hypotheses or connections between concepts
 - Translate physics problems into mathematical formalism

HEP Application Examples

- Large language models for human-machine collaborative particle accelerator tuning through natural language
[<https://www.science.org/doi/10.1126/sciadv.adr4173>]
- Xiwu Model: flexible system switching between foundation models (LLaMA, Vicuna, ChatGLM, Grok-1) [<https://arxiv.org/abs/2404.08001>]
- Physics-specific large-scale AI models inspired by LLMs [<https://arxiv.org/abs/2501.05382>]
- L-GATr: Lorentz Geometric Algebra Transformer for HEP data analysis [<https://arxiv.org/pdf/2411.00446>,
<https://arxiv.org/html/2405.14806v1>]

LLMs in Formal Theorem Proving

- Potential for automated proofs in mathematics and physics
 - Formal Theorem Proving by Rewarding LLMs to Decompose Proofs Hierarchically [<https://arxiv.org/pdf/2411.01829>]
 - STP: Self-play LLM Theorem Provers with Iterative Conjecturing and Proving [<https://arxiv.org/pdf/2502.00212v4>]
- Languages for theorem provers Lean, Coq, Isabelle

LLMs risks

- Many LLMs does generate inferences (answers) like **black-box** i.e. difficult to know the reason why the LLM gives specific answer
- To mitigate the problem there are several approaches:
 - in particular, one might use e.g. the tool DLBacktrace for LLM architecture-agnostic insights into decision pathways
[<https://doi.org/10.48550/arXiv.2411.12643>]
 - another one is to apply special architecture and domain specific data where LLM is used.
- ***Important:** Any LLM answer must be verified independently especially in possible critical consequences.*

LLM and Domain-Specific Data

- Controlled datasets
 - The First Release of ATLAS Open Data for Research
[<https://cds.cern.ch/record/2921349/files/ATL-SOFT-PROC-2025-007.pdf>]
 - AccGPT: A CERN Knowledge Retrieval Chatbot
[<https://indico.cern.ch/event/1338689/contributions/6010661/>]
 - Towards Unlocking Insights from Logbooks Using AI
[<https://arxiv.org/html/2406.12881v1>]
- Methods to organize accurate data
 - FAIR principle [<https://www.go-fair.org/fair-principles/>] - Findability, Accessibility, Interoperability, Reusability
 - HPC-FAIR: A Framework Managing Data and AI Models for Analyzing and Optimizing Scientific Applications [<https://www.osti.gov/biblio/2504172>]

The LLMs example in computing infrastructure

- Initial idea was to develop *artificial assistant (AA)*, where developer/administrator enters *a* question on natural language and gets the answer on the same language.
- There were developed *chatbot prototype for interaction with local description* of server network segment.

Architecture

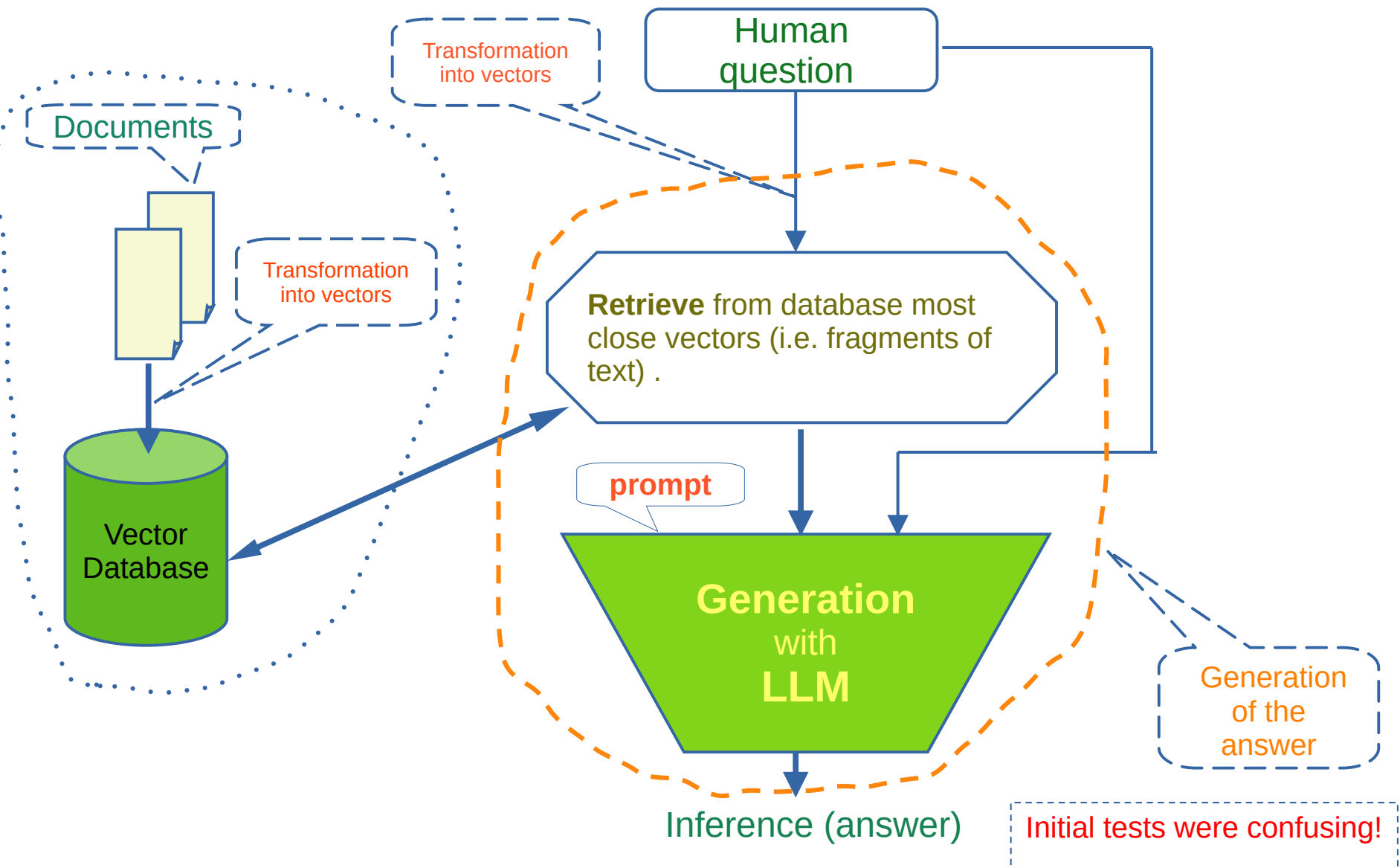
- The architecture Retrieval Augmented Generation (RAG) [<https://doi.org/10.48550/arXiv.2410.12837>, <https://doi.org/10.48550/arXiv.2501.09136>] has been chosen (invented in ~2021 [<https://doi.org/10.48550/arXiv.2005.11401>]).
- Advantages:
 - Pre-trained LLM — “knowledge transfer”.
 - All retrieval and generation operations are performed on local server, i.e. no local data be sent outside the LAN.
 - For local needs it is possible to use local GPU resources.

Test setup

- There were exploited three versions:
 - The Linux server Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz, 128GB, four simple GPUs “GeForce GTX 1080 Ti”, python3.12, Linux CLI.
 - MAC book pro m3, 36 GB.
 - Multipurpose computing center at NRC “Kurchatov Institute” <http://computing.nrcki.ru/> under SLURM with available for testing one GPU V100-PCIE-32GB.
- The initial python program for RAG was obtained from <https://gitlab.com/rahasak-labs/iollama> (redeveloped later on).
- Local LLMs instances like llama3.2, deepseek-r1, aya:35b, qwen2.5:32b, etc were used on platform ollama.
- The transformation of the text description into vector database (or vector array) (“embedding”) was used with several tools like LaBSE, paraphrase-multilingual-MiniLM-L12-v2, sbert_large_nlu_ru, etc. obtained from <https://huggingface.co/>.
- Web Frontend was configured with the package streamlit obtained from <https://github.com/streamlit>
- Initial version of the description for *server network segment* consisted of around hundred pages.

Basic (naive) Retrieval Augmented Generation (RAG)

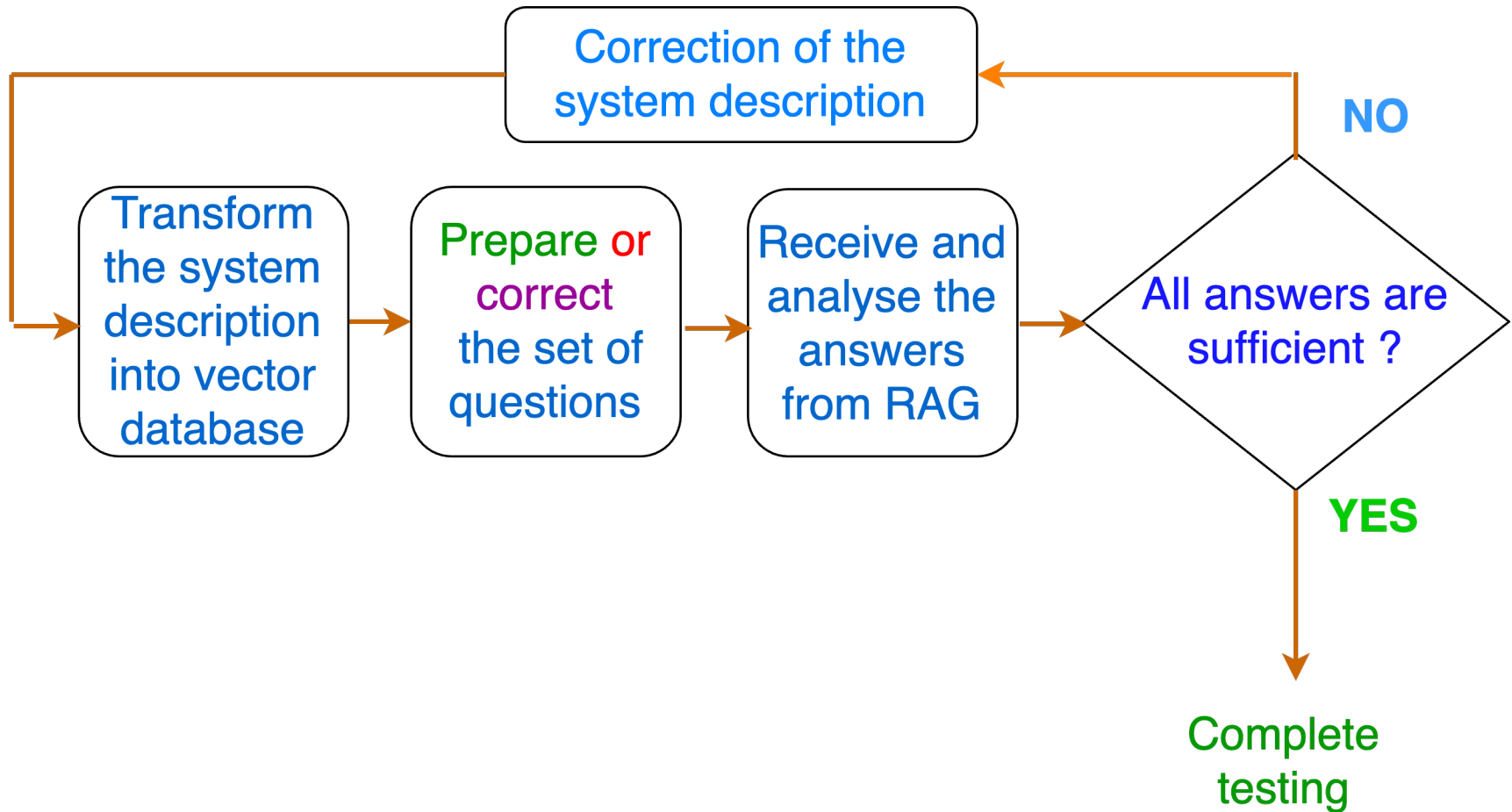
Transformation into vectors = embedding



Initial testing

- Around 10 questions were prepared and entered with the Web frontend.
- The output looked confusing:
 - Some answers were completely wrong.
 - Others were fantasie (hallutination) — *looks correct but not true.*
 - Anothers were not completely true.
- *To certificate the LLM answers* a review of the server network segment description has been done.
- Above review shown us that several parts of the description were not clear and complete as it must be.
 - **Our conclusion:** the description has to be improved and the same questions must be entered and to estimate the answers again.

The cycle to improve the description



At the end of the cycle

- Following was done:
 - Improved language, added several paragraphs.
 - The developers of the description discussed many times “what is what and from where it comes” (**very important**).
 - Here the RAG installation was used as the *partner (or special peer reviewer)* for developers to improve the document.
- The result: improved version of the server network segment description (the number of pages increased by about 50%, the content became more structured and complete, the terminology became more consistent).

Practical considerations

- A simple option is to order a list on one of the existing LLM, for example deepseek-r1 or in your RAG.
- You can use a prompt like “*Please prepare 70 test questions to check the quality of the server network description.*”
- Generated list of test questions might be entered into RAG one by one.
- The quality of answers *must be estimated* by mark from 0 to 10.
- The cycle might be considered as finished if there is no estimation mark is less than 7 from 10.
 - **Important remark:** the estimations are performed by the developers.
- The required time to produce the answer on a question was ~10 minutes in current test prototype (the number of pages ~150).
 - **Important remark:** in case of increasing document volume e.g. by 10^{**3} times a 10^{**3} times more powerful GPU cluster may be required.

A description improvement time depends on:

- Proximity of **LLM** to the description topics.
- The degree of conformity of the text-to-vector database transformation (**embedding**) correspondence to the description language/jargon.
- The volume of the description and available GPU set.
- Accuracy of instructions (**prompt**) to LLM.
- Choice of appropriate values of LLM parameters (temperature, top_p, num_ctx, etc)
- Completeness and clarity of the initial version of the description.
- **Most important:** *qualification of the description developers team.*

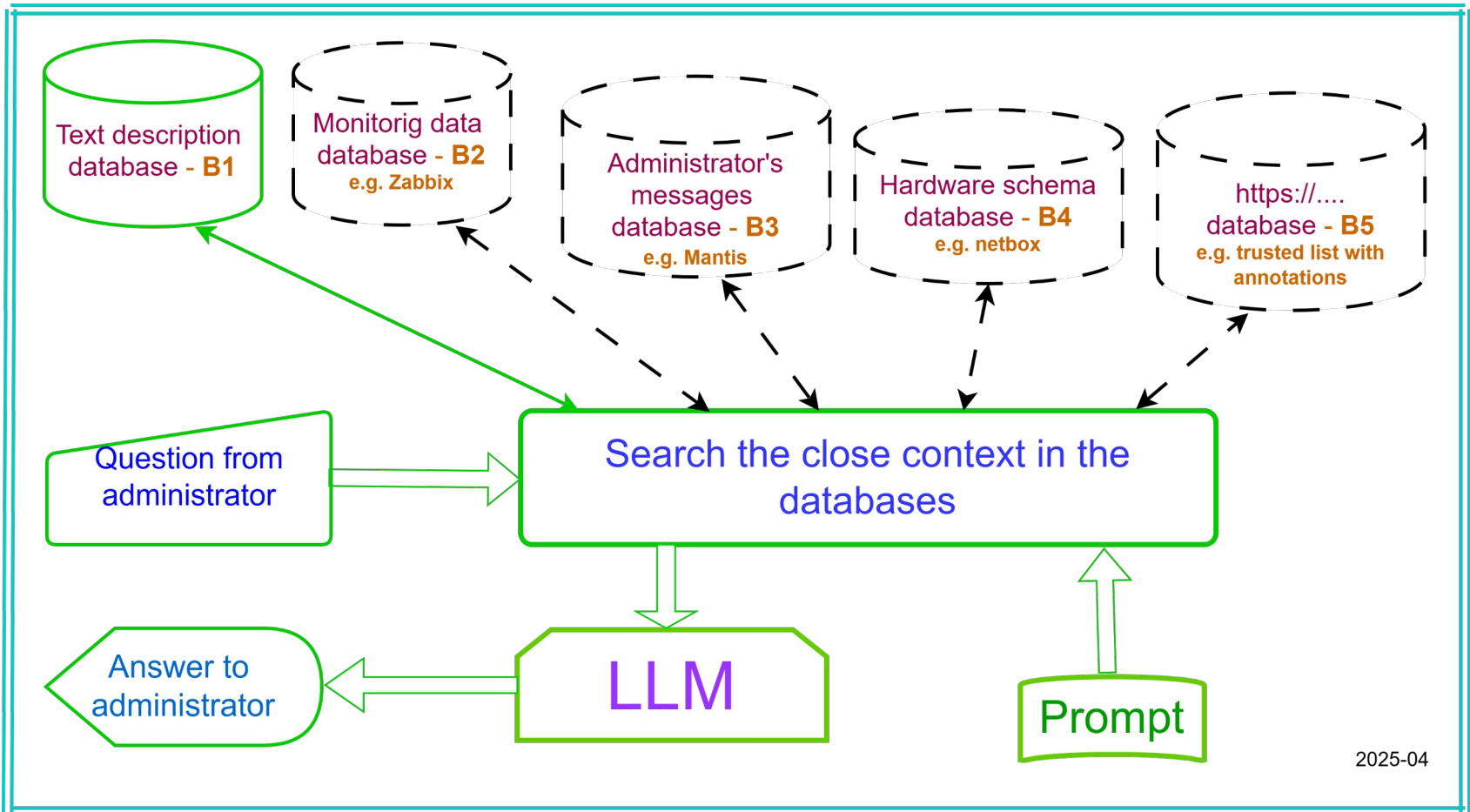
Architecture advantages

- Using the RAG architecture in development or administration simplifies/reduces:
 - Time for developing a description.
 - Involvement of new participants in development/administration.
 - Maintenance of the developed system.
 - Further modernization during operation.
 - Creation of a digital twin of a system being developed (or has been developed).
 - Discussions within the developers team (or administrators) to clarify the RAG architecture create greater mutual understanding within the team.

LLM in computing infrastructures future

- In the foreseeable future, the described technology will become a standard step in creating documentation for systems being developed.
 - The documents of the system description must be kept up to date.
- The development of a RAG architecture is an inevitable step towards creating a full-scale **digital twin** of a large system under development to help administrators or developers.

Digital twin: (possible team of RAGs)



LLM/LRM future developments

- Specialized RAGs for detectors, accelerators, computing infrastructure, simulations, etc
- Dedicated models, rather than relying solely on commercial LLMs.
- The pace of innovation of LLMs into HEP
 - Chep-2024 panel discussion conclusion:
 - The impact of ... LLMs on HEP computing would be evolutionary rather than revolutionary. Their integration is anticipated to occur progressively, building on existing systems and methodologies.

Future needs

- To empower the researchers they would need:
 - computing clusters equipped for LLM/LRM;
 - specific RAG configurations for concrete topics and areas of applications which might be considered as a team of RAGs (agents);
 - WLCG/Grid like infrastructure to access mentioned above team of RAGs together to get inferences for complex problems.

Questions?

Spare slides might
be interesting as well

Example of the dialog

Вы ввели: 2025-06-27 21:07:42 — Are there any inconsistencies in the infrastructure description that could mislead the administrator?

RAG_model: deepseek-r1:32b

RAG_embed_model: multilingual-e5

RAG_template: csd_ANGL_cryoem_template_2025-06-27_1215.txt

RAG_work_dir: /srv/users/shevel_aye/RAG-2024-07-30/iollama

RAG_docs: csd

The list of files in the directory 'csd'

— services-notes-2025-05-06.pdf

Example of the dialog – cont-1

ответы экспериментальной модели следует проверять!

2025-06-27 21:11:00 — Ответ модели: { "answer": "<think> Okay, I need to address the inconsistencies in the infrastructure description that could mislead an administrator. Let me go through each point carefully.

First, looking at the disk usage thresholds, there's a discrepancy between 95% and 98%. This inconsistency might cause confusion when setting up alerts or understanding system behavior. It would be best to standardize this across all scripts to avoid misunderstandings.

Next, the email addresses are inconsistent. The same person is listed with two different email domains. I should verify which one is correct and update all references accordingly to prevent miscommunication.

Lastly, there's a lack of monitoring for critical services like LDAP and VPN. Without proper checks in place, potential outages might go unnoticed, leading to service disruptions. Adding cron jobs or scripts to monitor these services would help catch issues early.

Example of the dialog – cont.-2

To summarize, the key steps are:

1. Standardize disk usage thresholds.
2. Correct and unify email addresses.
3. Implement monitoring for critical services with alerts.

This will ensure consistency and reliability in infrastructure management. </think>

Example of the dialog – cont.-3

Yes, there are inconsistencies in the infrastructure description that could potentially mislead an administrator:

1. Disk Usage Threshold Discrepancy:

- On page 149, `TestDiskUsage.bash` is set to trigger at 98% disk usage (`NOTACCESS=${ NOTACCESS + 1 }`).
- However, on page 146, it mentions a threshold of 95% (`if [${NOTACCESS} -gt 1] ; then`).

2. Email Address Inconsistency:

- On page 146, the script `TestServers.bash` is attributed to `Andrey.Shevel@pnpi.spb.ru` .
- But on page 148, it's mentioned as `shevel_aye@pnpi.nrcki.ru` .

3. Missing Monitoring for Critical Services:

- Page 44 discusses issues with LDAP and VPN but does not mention any monitoring or alerting mechanisms for these services.
- There is no cron job or script checking the status of `slapd` , `sssd` , or `openvpn` services.

These inconsistencies could lead to confusion when troubleshooting issues or managing infrastructure. For example, an administrator might be unaware of the correct email address to contact for support or may miss critical alerts about LDAP/VPN outages.

Example of the dialog – cont.-4

To resolve these:

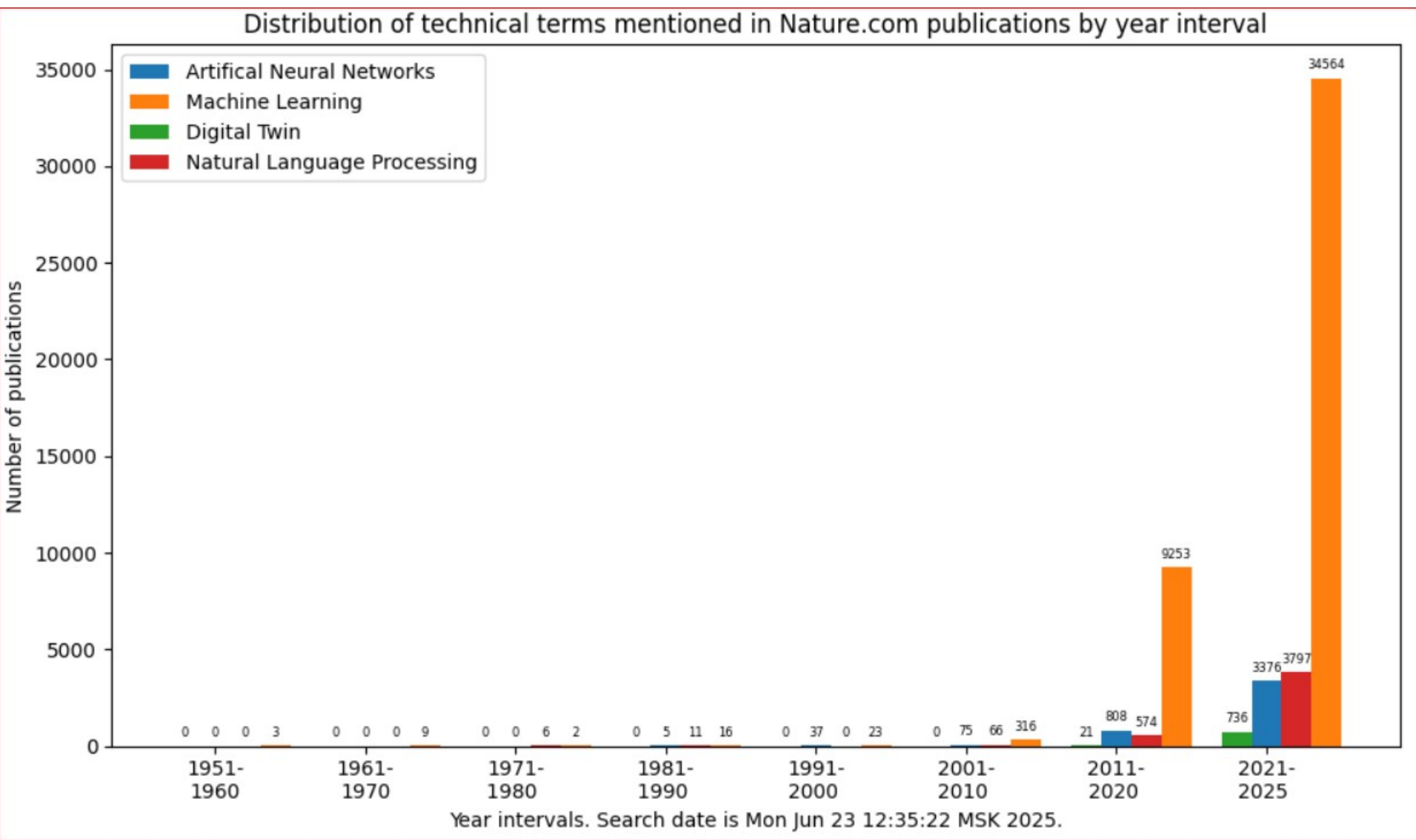
1. Standardize the disk usage threshold across all scripts.
2. Verify and update email addresses to ensure consistency.
3. Add monitoring for critical services like `slapd` , `sssd` , and `openvpn` with appropriate alerting mechanisms." }

Expected large sources of experimental data

- HL LHC ~2030
- Square Kilometre Array (SKA) ~2029
- Cherenkov Telescope Array Observatory (CTAO) ~2028
- Facility for Antiproton and Ion Research (FAIR) ~2028
- NICA (mpd) ~2026

Embedding

- **Embedding** — это преобразование текста документа (описание системы) в векторную базу данных.
- **Embedding** выполняется специальными программными средствами, которые реализованы на основе тренированных искусственных нейронных сетей.



CPU&GPU hardware progress












- <https://indico.cern.ch/event/1477299/contributions/6381816/>

Data Center AI Chip roadmap



Cores	AMD EPYC	CCD (Zen5/Zen5c)	Base/Boost* (up to GHz)	Default TDP (W)	L3 Cache (MB)	Price (1 KU, USD)
192 cores	9965	"Zen5c"	2.25 / 3.7	500W	384	\$14,813
160 cores	9845	"Zen5c"	2.1 / 3.7	390W	320	\$13,564
144 cores	9825	"Zen5c"	2.2 / 3.7	390W	384	\$13,006
128 cores	9755	"Zen5"	2.7 / 4.1	500W	512	\$12,984
	9745	"Zen5c"	2.4 / 3.7	400W	256	\$12,141
96 cores	9655	"Zen5"	2.6 / 4.5	400W	384	\$11,852
	9655P	"Zen5"	2.6 / 4.5	400W	384	\$10,811
	9645	"Zen5c"	2.3 / 3.7	320W	384	\$11,048
72 cores	9565	"Zen5"	3.15 / 4.3	400W	384	\$10,486
64 cores	9575F	"Zen5"	3.3 / 5.0	400W	256	\$11,791
	9555	"Zen5"	3.2 / 4.4	360W	256	\$9,826
	9555P	"Zen5"	3.2 / 4.4	360W	256	\$7,983
	9535	"Zen5"	2.4 / 4.3	300W	256	\$8,992
48 cores	9475F	"Zen5"	3.65 / 4.8	400W	256	\$7,592
	9455	"Zen5"	3.15 / 4.4	300W	192	\$5,412
	9455P	"Zen5"	3.15 / 4.4	300W	192	\$4,819
36 cores	9365	"Zen5"	3.4 / 4.3	300W	256	\$4,341
32 cores	9375F	"Zen5"	3.8 / 4.8	320W	256	\$5,306
	9355	"Zen5"	3.55 / 4.4	280W	256	\$3,694
	9355P	"Zen5"	3.55 / 4.4	280W	256	\$2,998
	9335	"Zen5"	3.0 / 4.4	210W	256	\$3,178
24 cores	9275F	"Zen5"	4.1 / 4.8	320W	256	\$3,439
	9255	"Zen5"	3.25 / 4.3	200W	128	\$2,495
16 cores	9175F	"Zen5"	4.2 / 5.0	320W	512	\$4,256
	9135	"Zen5"	3.65 / 4.3	200W	64	\$1,214
	9115	"Zen5"	2.6 / 4.1	125W	64	\$726
8 cores	9015	"Zen5"	3.6 / 4.1	125W	64	\$527

Humanity's Last Exam

Model	Accuracy (%) ↑	Calibration Error (%) ↓
 <u>Gemini 2.5 Pro</u>	21.6	72.0
 <u>o3</u>	20.3	34.0
 <u>o4-mini</u>	18.1	57.0
 <u>DeepSeek-R1-0528*</u>	14.0	78.0
 <u>o3-mini*</u>	13.4	80.0
 <u>Gemini 2.5 Flash</u>	12.1	80.0
 <u>Qwen3-235B*</u>	11.8	74.0
 <u>Claude 4 Opus</u>	10.7	73.0
 <u>DeepSeek-R1*</u>	8.5	73.0
 <u>Claude 3.7 Sonnet</u>	8.0	80.0
 <u>o1</u>	8.0	83.0

Natural neural networks

- Comparison of natural neural networks sizes
https://en.wikipedia.org/wiki/List_of_animals_by_number_of_neurons
- Massively dense reconstruction of the structural connections and functions of an entire millimeter volume of mouse visual cortex
<https://www.nature.com/collections/bdigiaicbd>