Novus: We have

- **a writer application that can generate seo optimized content**
- **Domain specific application: Call center, RAG, sales 360**
- **Agent orchestration application-> new main product**

What is an Agent:

- A large language model (LLM) that accepts user prompts (instructions) and can optionally integrate with external tools (e.g., weather API, YouTube downloader, flight booking service) and a database for message storage. For further details, please check the following resource: Smith Langchain Hub.

RAG:

What is RAG?

- Retrieval Augmented Generation (RAG) is a sophisticated architecture that blends search capabilities with Large Language Models (LLMs) to enhance the relevance and accuracy of generated responses
- RAG is a method to perform condition generations. This conditioning is a go-to method to mitigate LLM hallucinations. p(token|pervious_token, Corpus) -> pushes generations to be be more “factual”
- Naive RAG acts like a research assistant for a large language model. It first gathers information (data sourcing) and organizes it efficiently (data indexing). This information might be translated into a more suitable format for computer processing (data embedding). Then, when you ask a question, it searches through this organized data (retrieval) to find the most relevant parts and feeds them along with your question to the LLM for answer generation (generation).
- https://medium.com/@krtarunsingh/advanced-rag-techniques-unlocking-the-next-level-040c205b95bc -> MUST READ AND SUMMARIZE. These are advanced RAG techniques. But Semi, you must be able to explain them when needed…

How do you improve on existing RAG techniques?

- Applying more agentic approaches. I.E. query transformation can be thought of as an agent. Also using the right context enrichment techniques might be beneficial.

Do you train your own embedding models?

- Yes, We can fine-tune or train our models whenever sufficient data is available.

What are the performance gains you achieved? -> Benchmarks? @remind ML team pls

Agent

How do you optimize llm generation in terms of quality?

- Self-assessment: Retrieve the benchmarks from hugging face openllm leaderboard. Different benchmarks target different llm capabilities. For example, GSM8K tests llms mathematical reasoning capabilities. HumanEval tests llms coding abilities.
- Algorithm: Given query *"x"* use the best model for the task.(taha write it better)

How do you optimize llm generation in terms of balancing requests -> vllm ●
We have a library that optimizes different open source models for this task.

- TGI, Ollama, llama cpp
- Inference libraries

How do you deal with the problem of alignment? -> meta agent, ff screen… (taha very technical)
How do you deal with the problem of unstructured outputs that break llm agents -> finetuning (SFT) + In context learning method to provide structured outputs.

Product

How much compute do we need to host our agents?
At minimum but depending on use cases a GPU with vram of 24gbs. For example if you have 1 million calls per month. You would need approximately 8 * A100/H100?? How many gbs are these cards **@semi homework**

GPU memory 80GB source

Open source or closed source?
We support both paradigms.

Where do you store data?

We store data on premises, in the cloud (e.g., AWS), and using other storage solutions.

LLM

Do you have your own models? Do you "just" finetune existing models?

- Our self-assessment, and benchmarking pipelines allow us to dynamically finetune the best model for the task. Thus, achieving optimal task execution.
- We continue pretrainig foundational models on our curated datasets. It's the data that matters, the main algorithm for training such models hasn't changed since 2017.
- We also fully finetuned models. We don't only perform LoRA.