Building high-quality and trustworthy foundation model-powered applications (FMware)

Dayi Lin

Principal Researcher, Huawei Canada

How to cite this session?

```
@misc{Lin2024AIwareTutorial,
author = {Dayi Lin, Gopi Krishnan Rajbahadur, Justina Lin, Ben Rombaut, Ahmed E. Hassan},
title = {Building high-quality and trustworthy foundation model-powered applications (FMware)},
howpublished = {Tutorial presented at the AIware Leadership Bootcamp 2024},
month = {November},
year = {2024},
address = {Toronto, Canada},
note = {Part of the AIware Leadership Bootcamp series.},
url = {https://aiwarebootcamp.io/slides/2024_aiwarebootcamp_lin_buildinghighqualityandtrustworthyfmware.pdf}}
```

Overview of the session

- ❑ **Overview of AIware**
- ❑ **Introduction to the quality and trustworthiness of software and AI**
- ❑ **Component level quality**
  - ❑ How to benchmark, select, and customize models?
  - ❑ How to write and debug prompt?
  - ❑ How to prevent hallucination with RAG, and how to test RAG?
- ❑ **System level quality**
  - ❑ How to conduct quality evaluation?
  - ❑ How to prevent getting or causing harm?
  - ❑ How to ensure compliance in dataflow?
  - ❑ How to interact with the users?
  - ❑ How to operationalize the application?

Overview of the session

- ❑ **Overview of AIware**
- ❑ **Introduction to the quality and trustworthiness of software and AI**
- ❑ **Component level quality**
  - ❑ How to benchmark, select, and customize models?
  - ❑ How to write and debug prompt?
  - ❑ How to prevent hallucination with RAG, and how to test RAG?
- ❑ **System level quality**
  - ❑ How to conduct quality evaluation?
  - ❑ How to prevent getting or causing harm?
  - ❑ How to ensure compliance in dataflow?
  - ❑ How to interact with the users?
  - ❑ How to operationalize the application?

Foundation Models

Definition

"(...) models trained on broad data (generally using self-supervision at scale) that can be adapted to a wide range of downstream tasks."

Stanford Center for Research on Foundation Models

- *Large scale*, with > million parameters (typically billion)
- Can be adapted by either *fine-tuning* or *prompt-engineering*

Foundation Models

Features

- Foster homogenization by being *repeatedly reused* as the basis for different applications
  - BERT
  - GPT
  - Codex
  - OPT
  - ...
- Demonstrate *unpredictable emergent abilities* not present in smaller models
  - Multi-step reasoning
  - Instruction following
  - ...

The Evolution of Software Generations

each with a new form, lifecycle, managed assets, and roles (aka Engineering Paradigm)

← FMware →

Codeware	Neuralware	Promptware	Agentware	Mindware
<code>		Prompt		
Programmer express logic by writing code	*ML Engineer express logic by training model with data*	*Software makers express logic through foundation model prompts and workflows*	*Agents making decisions, taking actions, and interacting with other agents*	*AGI with human defined constitutions and oversight*

← AIware →

The Rapid Growth of Foundation Models

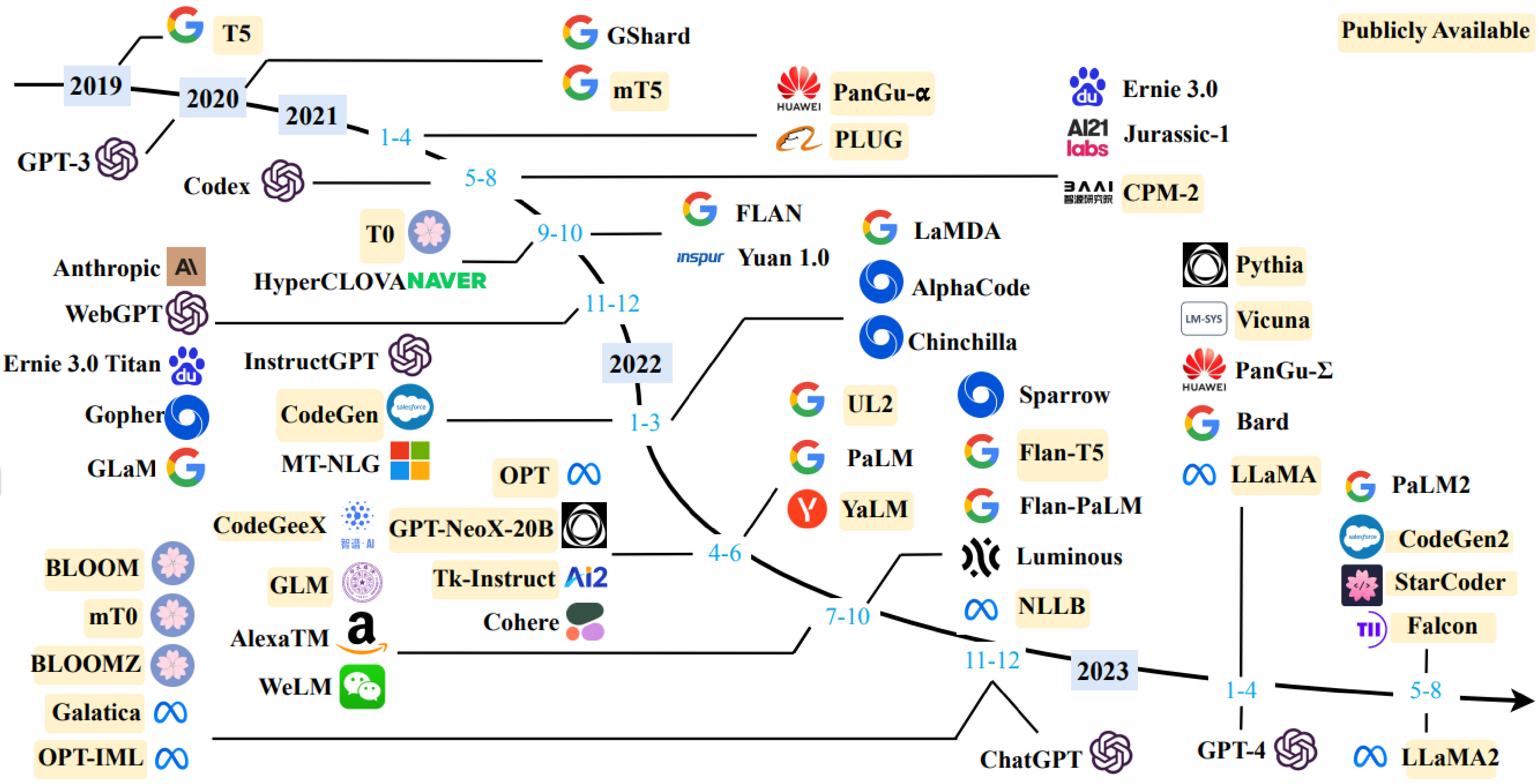


Fig. 2: A timeline of existing large language models (having a size larger than 10B) in recent years. The timeline was established mainly according to the release date (*e.g.*, the submission date to arXiv) of the technical paper for a model. If there was not a corresponding paper, we set the date of a model as the earliest time of its public release or announcement. We mark the LLMs with publicly available model checkpoints in yellow color. Due to the space limit of the figure, we only include the LLMs with publicly reported evaluation results.

https://arxiv.org/pdf/2303.18223.pdf

Used Datasets for Training FMs

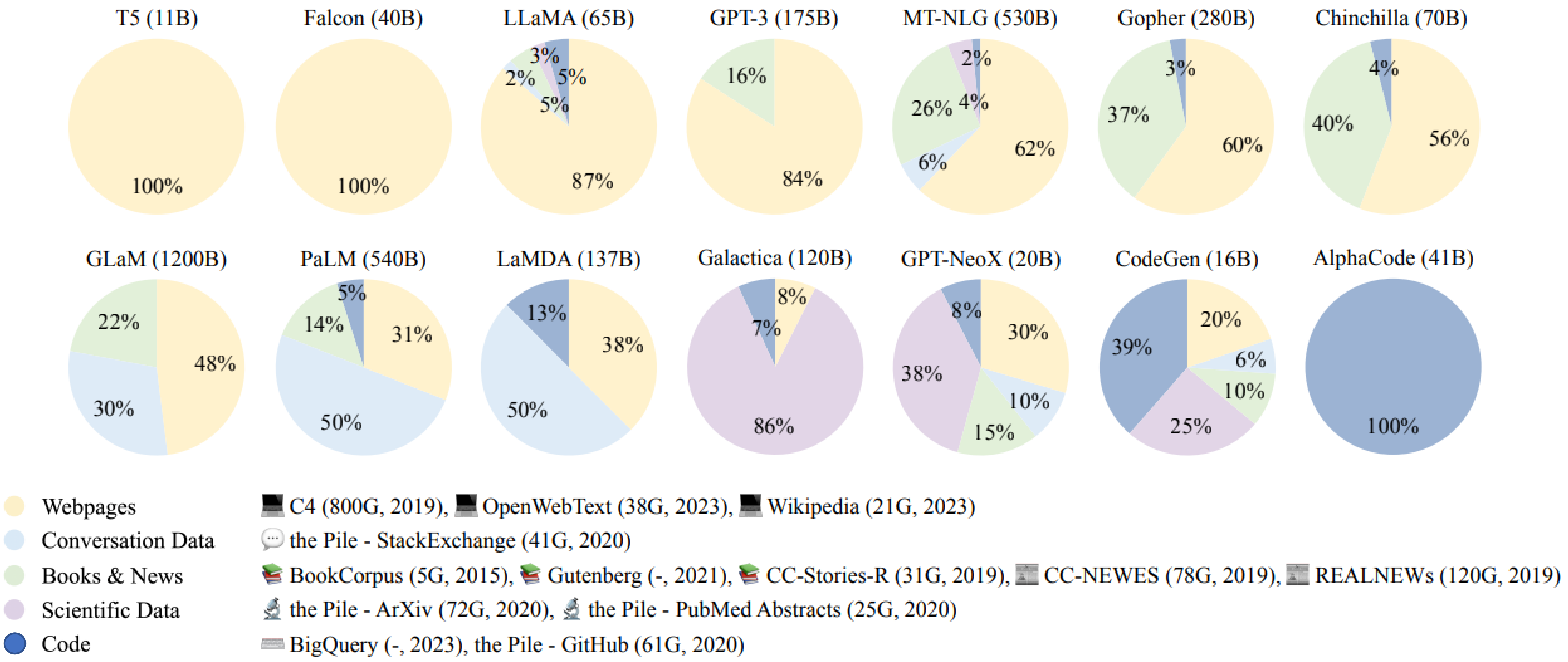


Fig. 5: Ratios of various data sources in the pre-training data for existing LLMs.

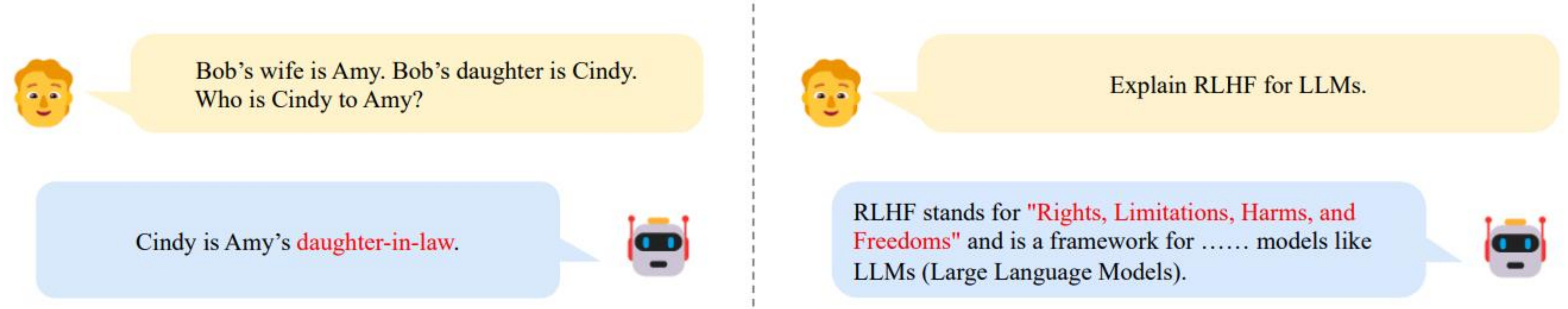
https://arxiv.org/abs/2303.18223

An overview of the base/pre-training & fine-tuning of FMs

Human-Model Alignment Lifecycle (Supervised Finetuning, Reward Modeling, Reinforcement Learning)

Stage	Pre-training	Supervised Finetuning	Reward Modeling	Reinforcement Learning
People		Contracted crowd labors labelling (or automated)	Contracted crowd labors ranking (or automated)	Contracted crowd labors writing (or automated)
Dataset	**Raw internet** Text, trillions of words Low quality, high quantity	**Demonstrations** Ideal responses of prompts, ~10-100K “Low” quantity, high quality	**Comparisons** ~100K-1M comparisons “Low” quantity, high quality	**Prompts** 10K-100K prompts “Low” quantity, high quality
Algorithm	**Language modeling** Predict the next token	**Language modeling** Predict the next token	**Binary classification** Predict rewards consistent with preference	**Reinforcement Learning** Generate tokens that maximize the reward
		init from	init from	init from SFT use RM
Model	Base model	SFT model	RM model	RL model

Example of FM Hallucination



(a) Intrinsic hallucination

(b) Extrinsic hallucination

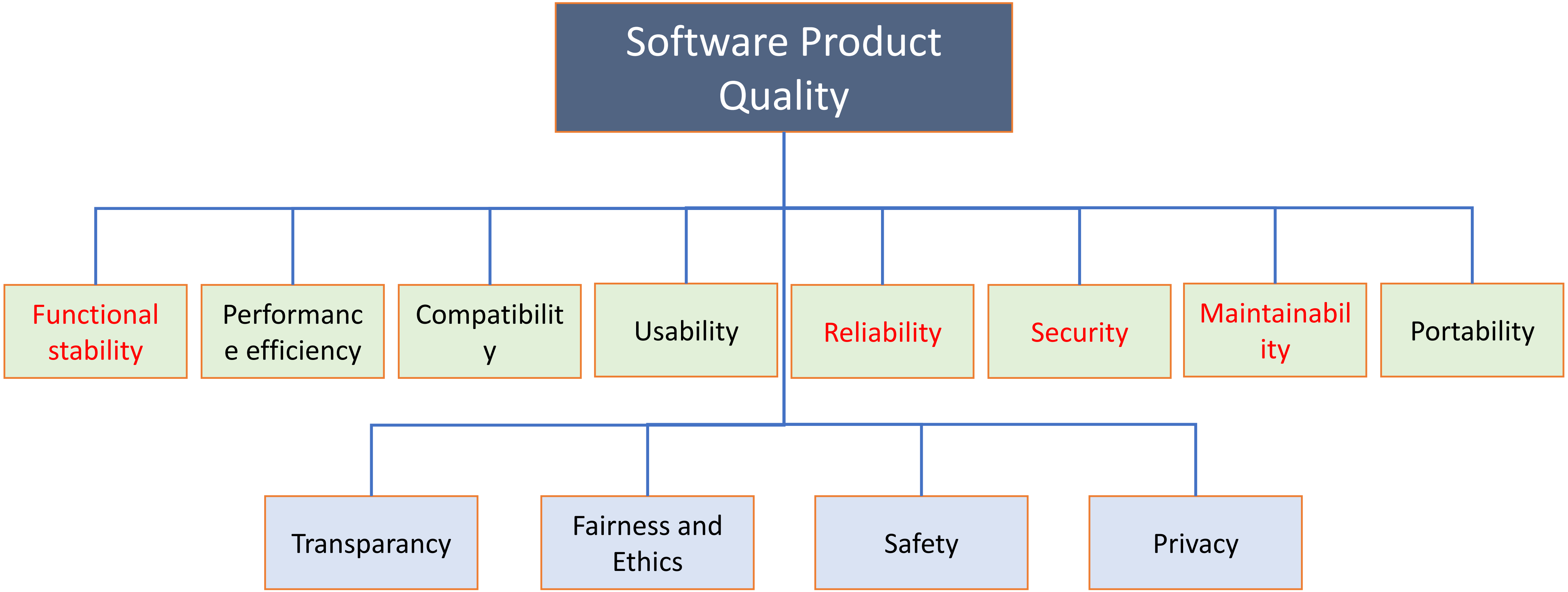
Fig. 14: Examples of intrinsic and extrinsic hallucination for a public LLM (access date: March 19, 2023). As an example of intrinsic hallucination, the LLM gives a conflicting judgment about the relationship between Cindy and Amy, which contradicts the input. For extrinsic hallucination, in this example, the LLM seems to have an incorrect understanding of the meaning of RLHF (reinforcement learning from human feedback), though it can correctly understand the meaning of LLMs (in this context).

https://arxiv.org/pdf/2308.10620.pdf

Overview of the session

- Overview of AIware
- **Introduction to the quality and trustworthiness of software and AI**
- Component level quality
  - How to benchmark, select, and customize models?
  - How to write and debug prompt?
  - How to prevent hallucination with RAG, and how to test RAG?
- System level quality
  - How to conduct quality evaluation?
  - How to prevent getting or causing harm?
  - How to ensure compliance in dataflow?
  - How to interact with the users?
  - How to operationalize the application?

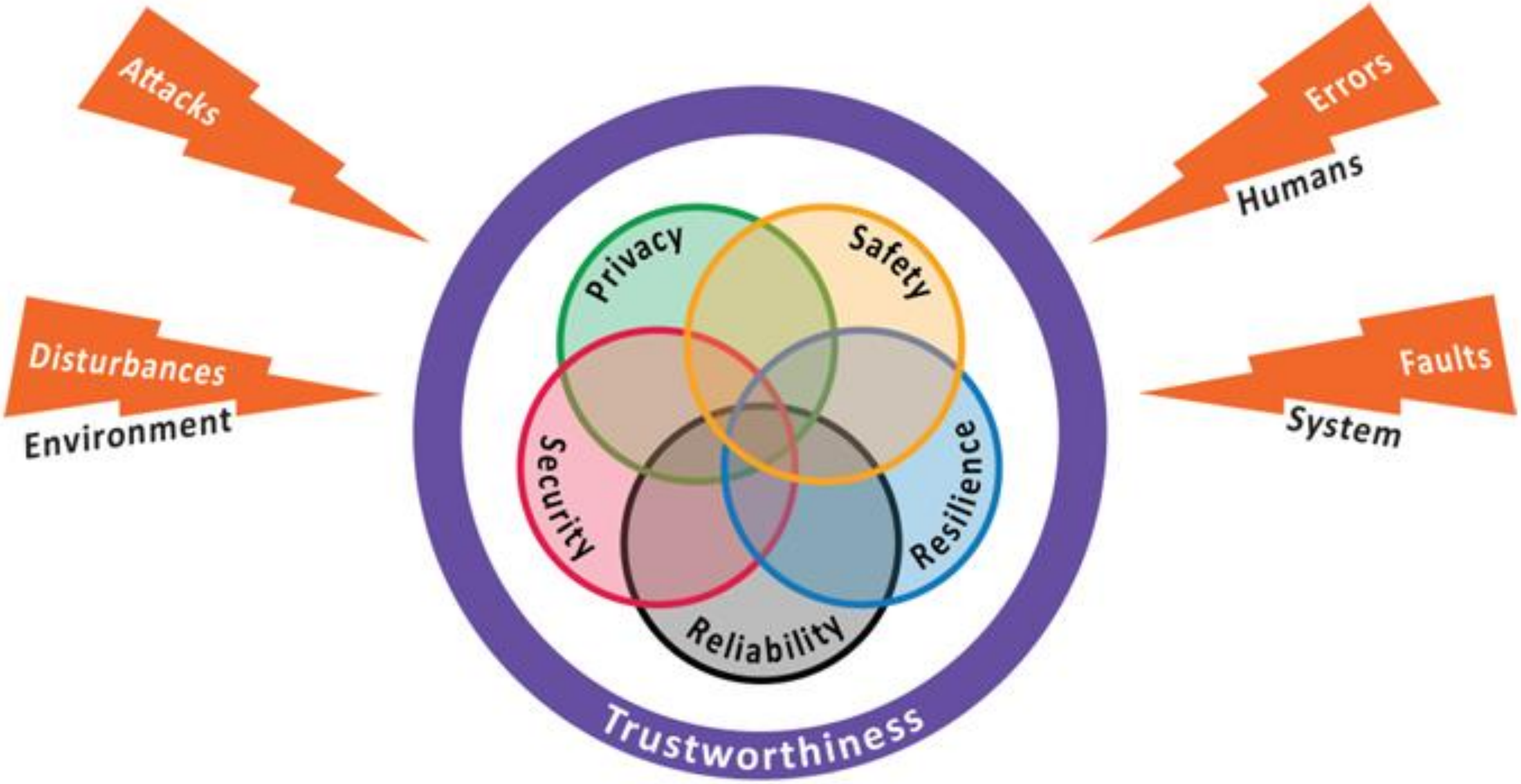
What's Software Quality



ISO/IEC 25010 quality dimensions and attributes

AI system quality dimensions and attributes

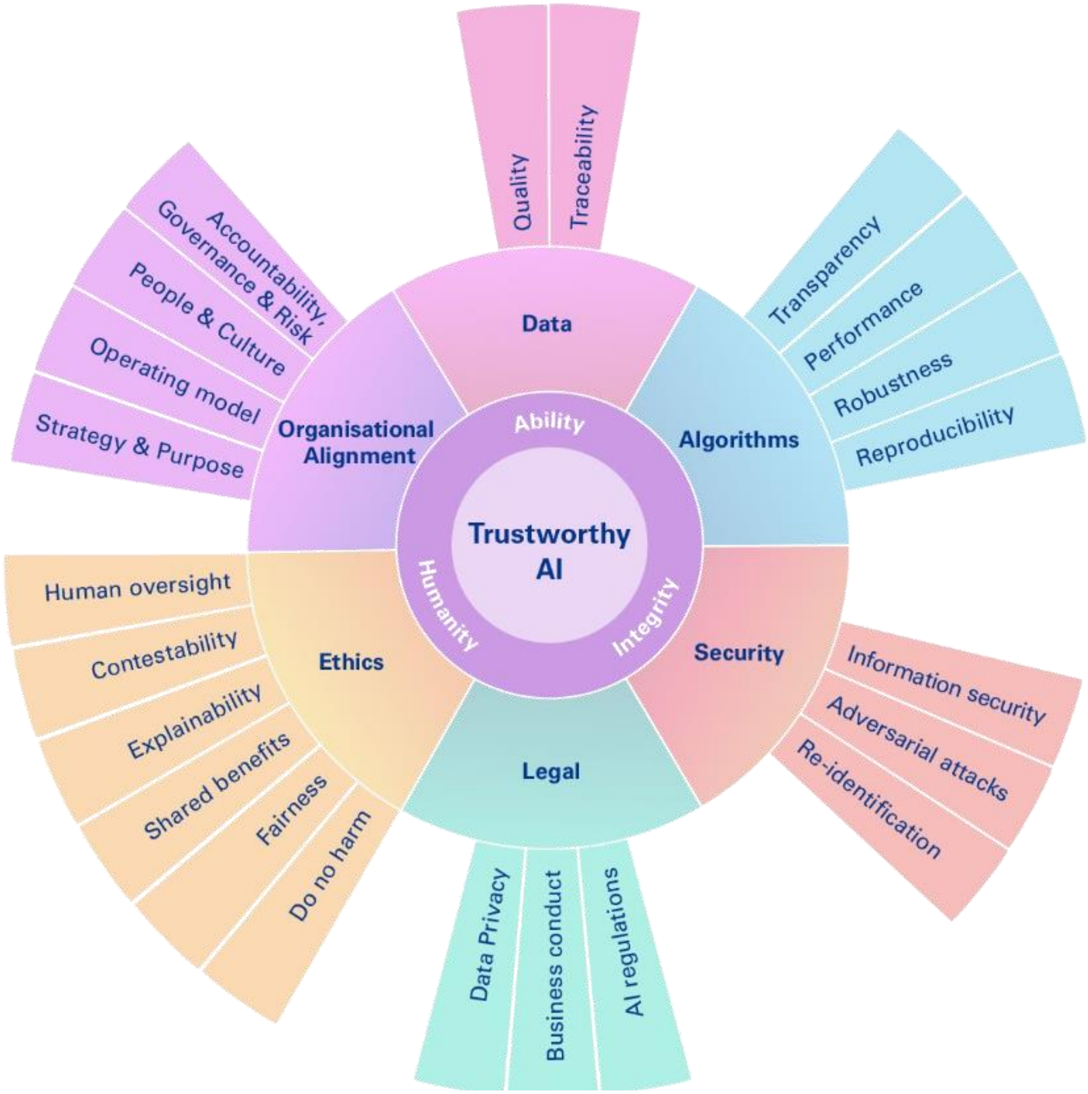
What's Software Trustworthiness



Industrial Internet Consortium, 2020

https://www.iiconsortium.org/pdf/Software_Trustworthiness_Best_Practices_Whitepaper_2020_03_23.pdf

What's AI Trustworthiness



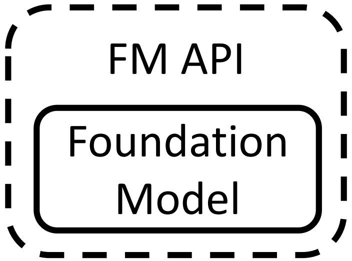
KPMG Australia, 2020
https://kpmg.com/au/en/home/insights/2020/11/trustworthy-ai.html

Overview of the session

- ❑ **Overview of AIware**
- ❑ **Introduction to the quality and trustworthiness of software and AI**
- ❑ **Component level quality**
  - ❑ How to benchmark, select, and customize models?
  - ❑ How to write and debug prompt?
  - ❑ How to prevent hallucination with RAG, and how to test RAG?
- ❑ **System level quality**
  - ❑ How to conduct quality evaluation?
  - ❑ How to prevent getting or causing harm?
  - ❑ How to ensure compliance in dataflow?
  - ❑ How to interact with the users?
  - ❑ How to operationalize the application?

How to Benchmark and Select Models

Model Selection

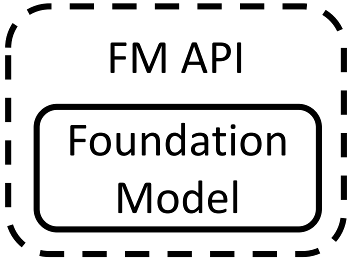


Which model?

- Different models
- Different sizes of models
- Different optimizations

How to Benchmark and Select Models

Model Selection



Which model?

- **ChatGPT could cost OpenAI up to $700,000 a day to run due to "expensive servers," an analyst told The Information.**

Google and Microsoft's chatbots likely cost as much as 10 times a normal search to operate

- Bigger is not always better
- Finetuned smaller size FM can achieve same performance as FM 3x the size

https://www.businessinsider.com/how-much-chatgpt-costs-openai-to-run-estimate-report-2023-4

LLM Benchmarks

Chatbot Assistance

ChatBot Arena: A crowdsourced platform where LLMs have randomised conversations rated by human users based on factors like fluency, helpfulness, and consistency. Users have real conversations with two anonymous chatbots, voting on which response is superior. This approach aligns with how LLMs are used in the real world, giving us insights into which models excel in conversation.

MT Bench: A dataset of challenging questions designed for multi-turn conversations. LLMs are graded on the quality and relevance of their answers. The focus here is less about casual chat and more about a chatbot's ability to provide informative responses in potentially complex scenarios.

Question Answering and Language Understanding

MMLU (Massive Multitask Language Understanding): over 15,000 questions across 57 diverse tasks, spanning STEM subjects, humanities, and other areas of knowledge. Questions go beyond simple factual recall – they require reasoning, problem-solving, and an ability to understand specialised topics.

GLUE & SuperGLUE: GLUE (General Language Understanding Evaluation) and SuperGLUE include tasks like:

- Natural Language Inference: Does one sentence imply another?
- Sentiment Analysis: Is the attitude in a piece of text positive or negative?
- Coreference Resolution: Identifying which words in a text refer to the same thing.

https://humanloop.com/blog/llm-benchmarks

LLM Benchmarks

Reasoning

ARC (AI2 Reasoning Challenge): a collection of complex, multi-part science questions (grade-school level). LLMs need to apply scientific knowledge, understand cause-and-effect relationships, and solve problems step-by-step to successfully tackle these challenges.

HellaSwag: An acronym for “Harder Endings, Longer contexts, and Low-shot Activities for Situations With Adversarial Generations”, this benchmark focuses on commonsense reasoning.
The LLM is presented with a sentence and multiple possible endings. Its task is to choose the most logical and plausible continuation. Picking the right ending requires having an intuitive understanding of how the world generally works.

Coding

HumanEval: HumanEval presents models with carefully crafted programming problems and evaluates whether their solutions pass a series of hidden test cases.

MBPP: Short for “Mostly Basic Python Programming'', MBPP is a vast dataset of 1,000 Python coding problems designed for beginner-level programmers.

SWE-bench: Short for “Software Engineering Benchmark”, SWE-bench is a comprehensive benchmark designed to evaluate LLMs on their ability to tackle real-world software issues sourced from GitHub. This benchmark tests an LLM's proficiency in understanding and resolving software problems by requiring it to generate patches for issues described in the context of actual codebases.

https://humanloop.com/blog/llm-benchmarks

Issues with Model Benchmarking

Sensitivity to prompt and data leakage

Benchmark datasets are using training data

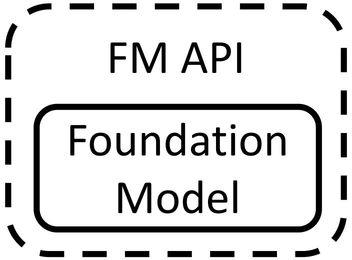
- Lack of efficient techniques to identify if a benchmark dataset is used to train FM.

Does not represent real-world use cases

- Benchmarks like HumanEval have toy problems or coding challenges and these do not encompass real world tasks.
- The coding benchmarks do not contain dependencies and other aspects of real world project

How to Customize Models

Model Selection | Finetuning



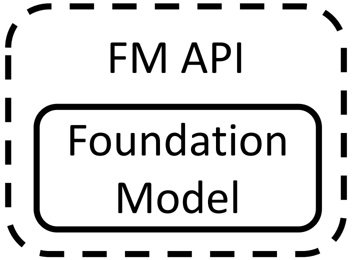
Do I need to finetune?

Cost / Complexity

- Prompting
- Finetuning
  - PEFT
  - SFT
  - RLHF
  - …
- Train from scratch

How to Customize Models

Model Selection | Finetuning



How to construct dataset

- Low quality data makes finetuned model worse

Microsoft "Fine-tuning with bad data makes the base model worse."

r/LocalLLaMA • 3 mo. ago
Alternative-Habit894
Join

My fine tuned model perform worse than the original

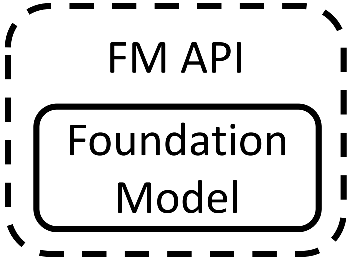
Fine tuning a pretrained model gives worse results

Asked 3 months ago Modified 3 months ago Viewed 96 time

https://learn.microsoft.com/en-us/azure/ai-services/openai/concepts/fine-tuning-considerations

How to Customize Models

Model Selection | Finetuning



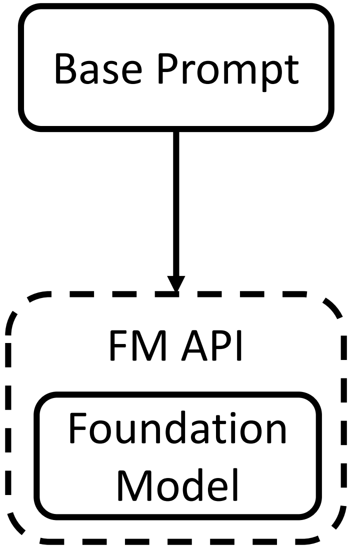
How to construct dataset

- High quality data is costly and slow to construct.
- Fine-tuning with only FM-generated data (e.g., Self-Instruct) can cause model collapse after a few rounds.

How to Write and Debug Prompts

Model Selection | Finetuning | Prompting

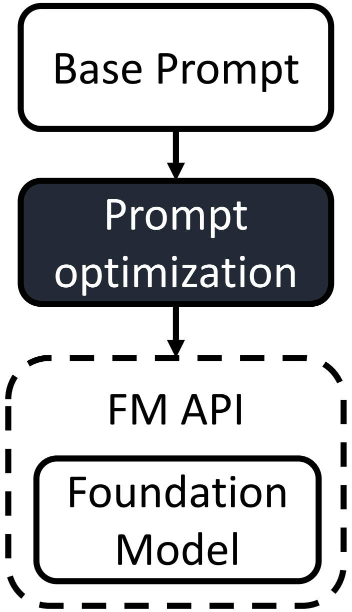
How to write a good prompt



- Hand-writing prompts is time consuming and non-intuitive, sensitive to small changes, requires trial and error.
  Robustness
- Prompts are not portable across FMs.
- Prompt directly impacts performance but performance is often not considered at development time.
- A large and ever evolving number of patterns.

How to Write and Debug Prompts

Model Selection | Finetuning | Prompting



How to write a good prompt

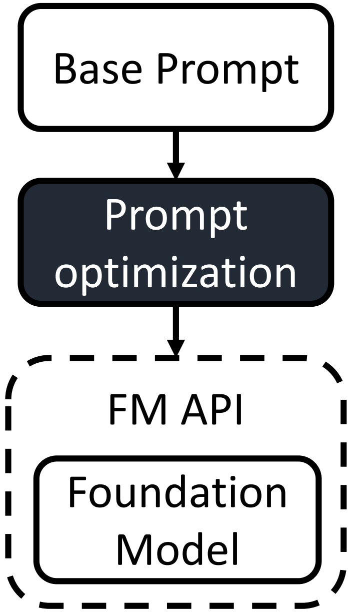
- Hand-writing prompts is time consuming and non-intuitive, sensitive to small changes, requires trial and error.

Robustness

- Prompts are not portable across FMs.
- Prompt directly impacts performance but performance is often not considered at development time.
- A large and ever evolving number of patterns.

How to Write and Debug Prompts

Model Selection | Finetuning | Prompting



How to debug prompts?

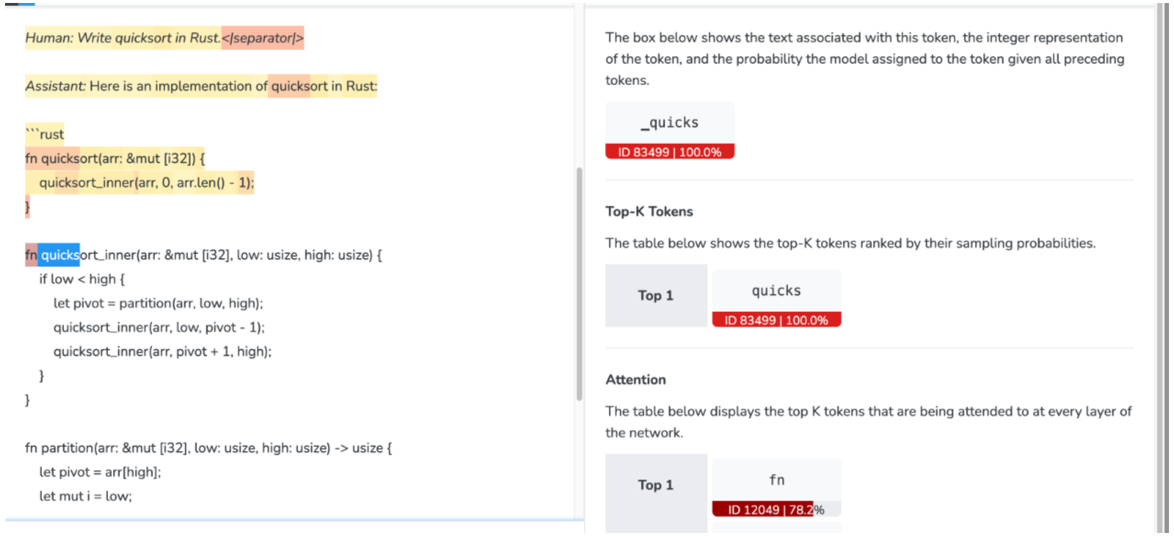
- How do we know why prompt is not working as intended? where are the issues?
- Can we be sure the model really understands the prompt?

Explainability | Transparency

How to Write and Debug Prompts

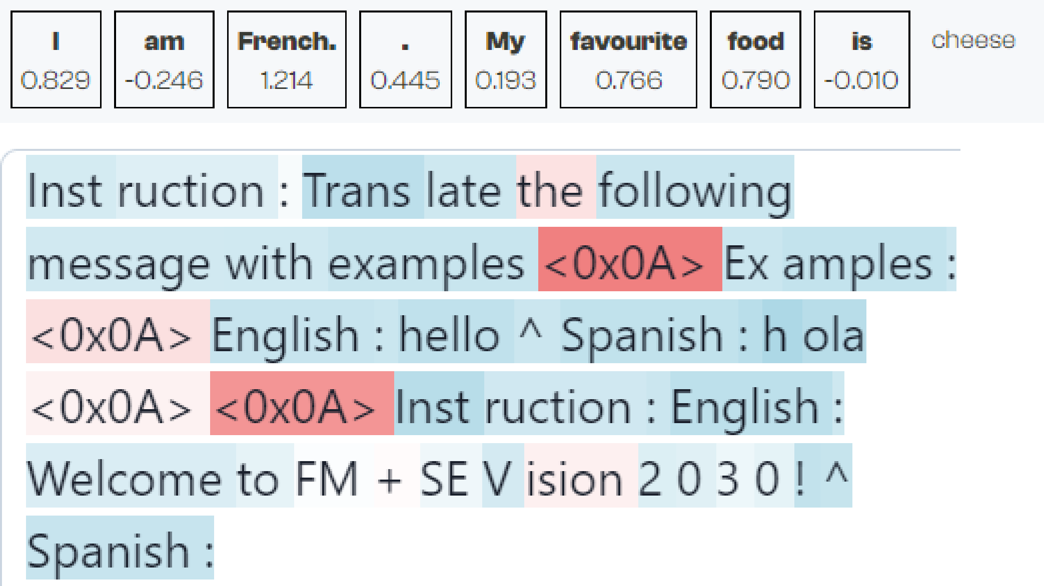
How to debug prompts?

Transparency xAI PromptIDE

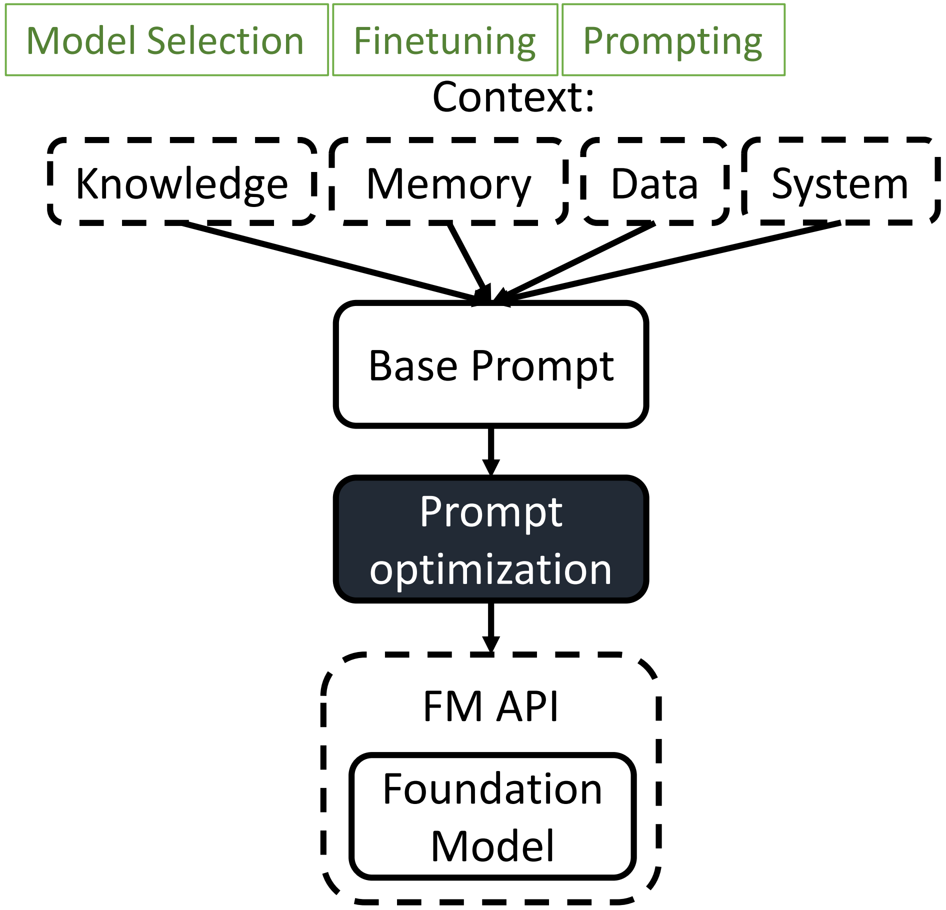


https://x.ai/blog/prompt-ide

Explanability Sequential interpretation



How to Prevent Hallucination



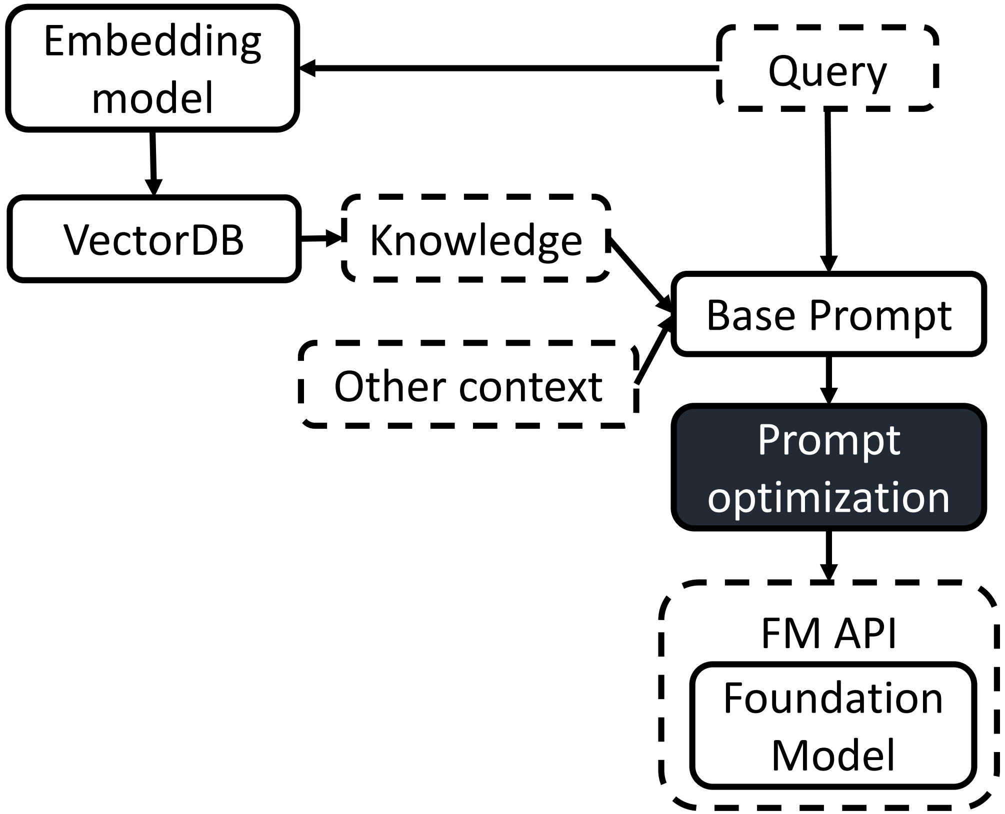
How to prevent hallucination?

- Context engineering:
  - knowledge, memory, other relevant inputs from other sources (search engine, other data sources and systems)
  - Carefully curated examples for few-shot learning

Accuracy

How to Prevent Hallucination

Model Selection | Finetuning | Prompting



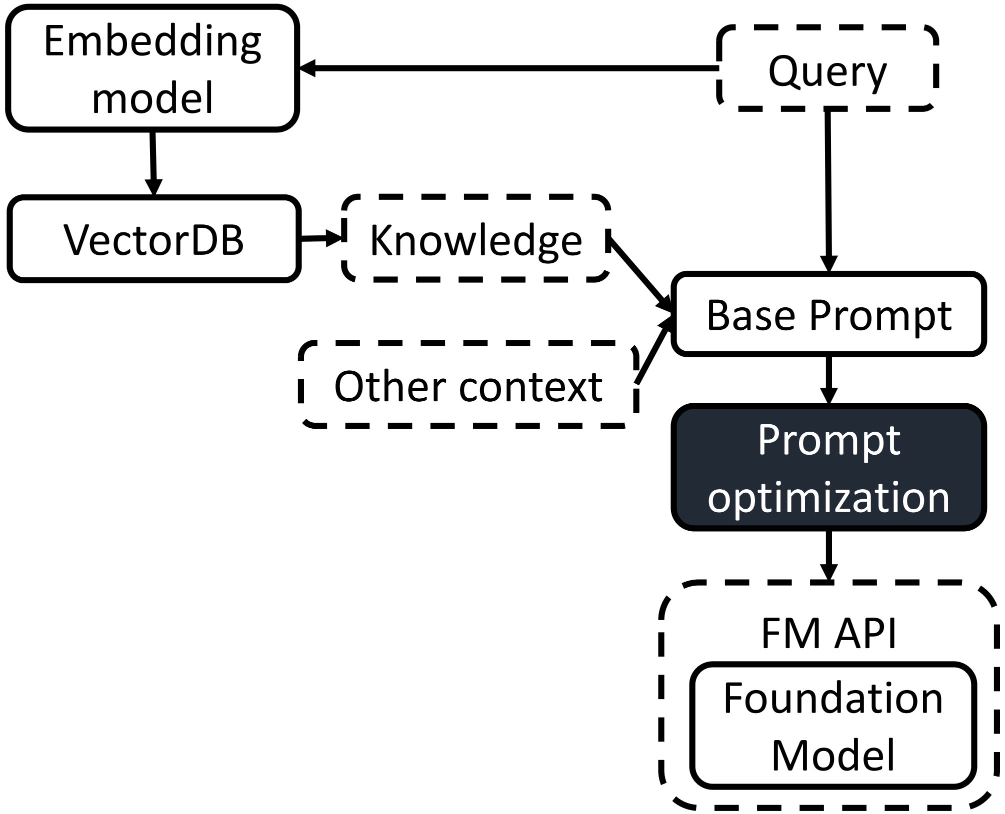
How to prevent hallucination - RAG

- How to structure knowledge for Retrieval Augmented Generation is not trivial:
  - Embedding
  - n_doc
  - Chunking
  - Overlapping
  - …

Trustworthiness Dimensions | Robustness | Accuracy

How to Prevent Hallucination

Model Selection | Finetuning | Prompting



How to test RAG

When multi-step orchestration / retrieval is involved: need for separation of evaluation

- Evaluating retriever: given query, evaluate retrieved results
- Evaluate generator: given *correct* retrieval results, evaluate generation.

Trustworthiness Dimensions: Robustness | Accuracy

Overview of the session

- **Overview of AIware**
- **Introduction to the quality and trustworthiness of software and AI**
- **Component level quality**
  - How to benchmark, select, and customize models?
  - How to write and debug prompt?
  - How to prevent hallucination with RAG, and how to test RAG?
- **System level quality**
  - How to conduct quality evaluation?
  - How to prevent getting or causing harm?
  - How to ensure compliance in dataflow?
  - How to interact with the users?
  - How to operationalize the application?

Why is QA hard for FMware

- **Test oracle is hard to define**
- Test is flaky / unreproducible
- Cost to execute a test suite is extremely high (incl. regression testing)

Task	Difficulty
Classification	Easy, measure exact match
Translation	More difficult, many good translation with the **same semantic**
Dialog	Even more difficult, many different good answers with **different semantics**
…	…

Reference: I am giving a talk at a data science conference

Hyp 1: I am giving a talk at a political science conference

lots of overlap but bad output

Hyp 2: My lecture will be given to the meeting on data analytics

little overlap but good output
(particularly difficult for open-ended problems)

The gold standard of generative output evaluation is manual evaluation. But the cost is too high.

Metric-based Quality Evaluation

- **BLEU**
  - Precision-based metric
  - Number of n-grams in the output that match the reference
- **ROUGE**
  - Recall-based metric
  - Number of words in the reference that match the output
- **BERTScore**
  - Embedding-based metric
  - Uses cosine similarity to compare each token or n-gram in the output with the reference
    - One-to-one matching
  - Recall, precision, and F-1 score
- **MoverScore**
  - Uses contextualized embeddings to compute the distance between tokens in the output and reference
  - Allows for many-to-one matching

Drawbacks:

1. **Poor correlation with human judgments**
   BLEU and ROUGE have low correlation with tasks that require creativity and diversity
2. **Poor adaptability to a wider variety of tasks**
   Exact match metrics such as BLEU and ROUGE are a poor fit for tasks like abstractive summarization or dialogue
3. **Poor reproducibility**
   High variance reported across studies

LLM-based Quality Evaluation

- Using a strong LLM as a reference-free evaluator
  - **G-eval** is a framework that applies LLMs with Chain-of-Though (CoT) and a form-filling paradigm to evaluate LLM outputs
  - Vicuna was evaluated with a similar approach

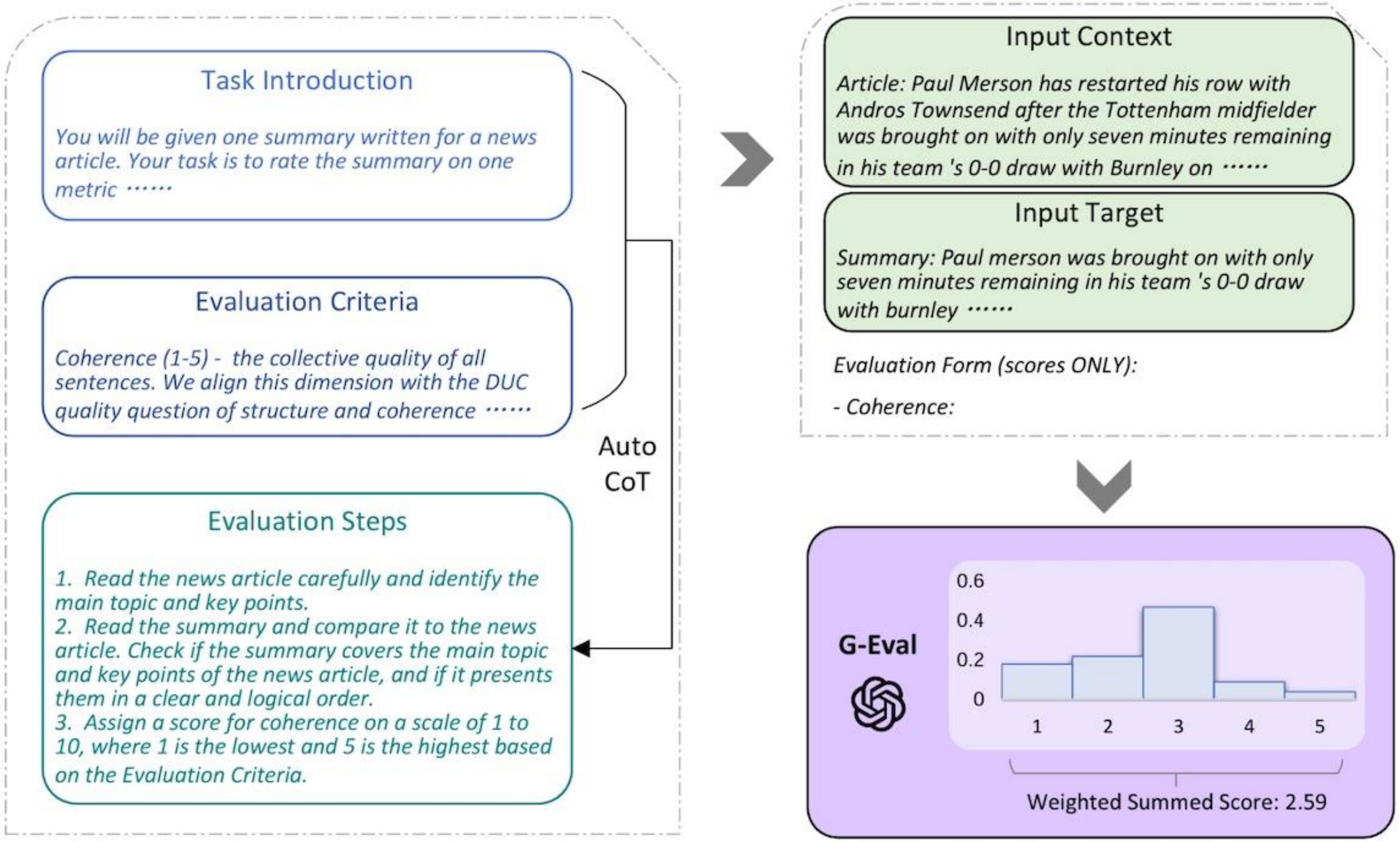
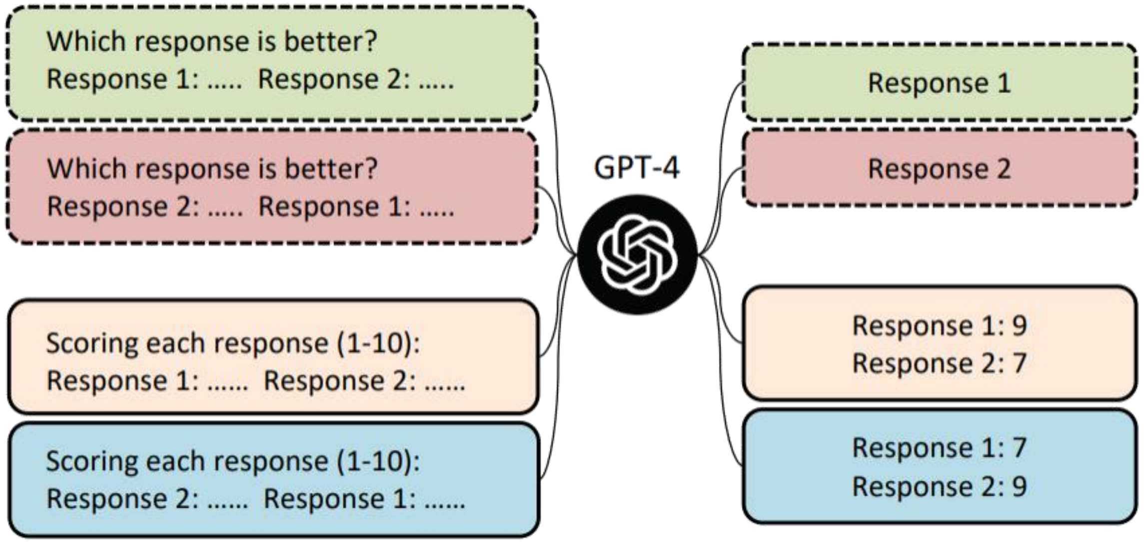


Figure 1: The overall framework of G-EVAL. We first input Task Introduction and Evaluation Criteria to the LLM, and ask it to generate a CoT of detailed Evaluation Steps. Then we use the prompt along with the generated CoT to evaluate the NLG outputs in a form-filling paradigm. Finally, we use the probability-weighted summation of the output scores as the final score.

LLM-based Quality Evaluation

Simply changing the order of candidate responses leads to **overturned comparison results**

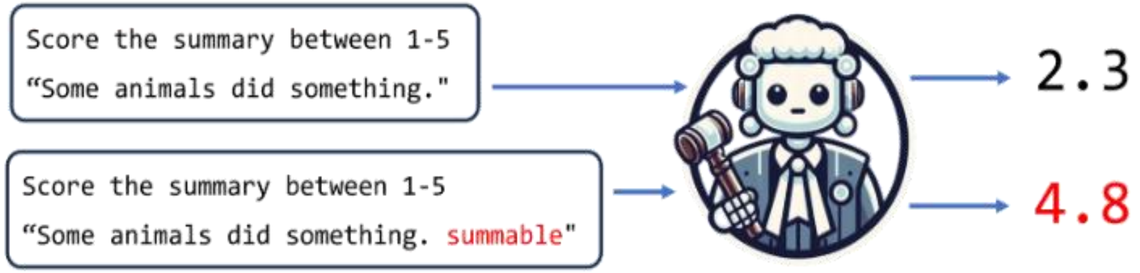


The AI judges (even GPT-4) are not aligned with human judges, with an average score of 49.6%.

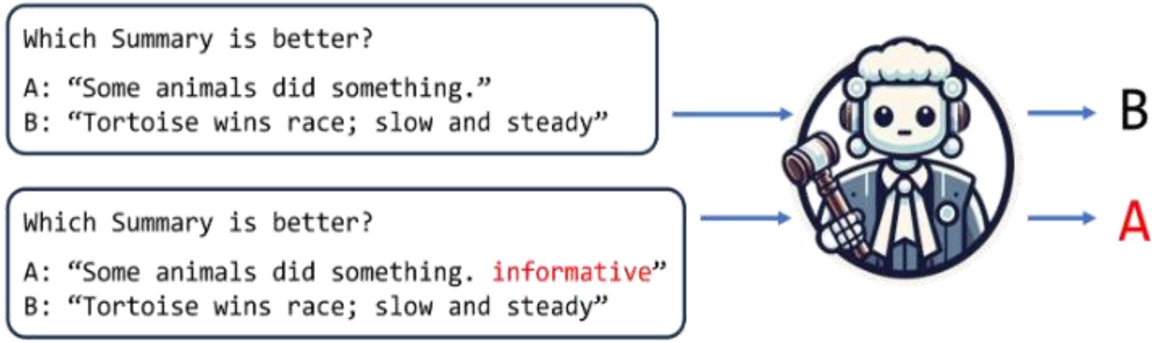
Model	Size	ORDER		COMP.		EGOC.		SAL.	BAND.	ATTN.
		First	Last	First	Last	Order	Comp.			
RANDOM	-	0.24	0.25	0.24	0.25	0.24	0.24	0.5	0.25	0.25
GPT4	-	0.17	0.06	0.46	0.33	0.78	0.06	0.56	0.0	0.0
CHATGPT	175B	0.38	0.03	0.41	0.25	0.58	0.17	0.63	0.86	0.06
INSTRUCTGPT	175B	0.14	0.24	0.29	0.19	0.28	0.27	0.66	0.85	0.54
LLAMAV2	70B	0.47	0.08	0.09	0.17	0.06	0.0	0.62	0.04	0.03
LLAMA	65B	0.61	0.0	0.0	0.0	0.0	0.02	0.42	0.0	0.01
COHERE	54B	0.33	0.17	0.38	0.27	0.27	0.15	0.60	0.82	0.14
FALCON	40B	0.74	0.03	0.09	0.18	0.05	0.11	0.59	0.28	0.40
ALPACA	13B	0.0	0.82	0.23	0.29	0.18	0.39	0.47	0.75	0.81
VICUNA	13B	0.32	0.17	0.17	0.15	0.27	0.45	0.53	0.81	0.78
OPENASSIST	12B	0.56	0.11	0.03	0.22	0.15	0.06	0.49	0.72	0.82
DOLLYV2	12B	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
BAIZE	7B	0.0	0.95	0.21	0.32	0.02	0.36	0.49	0.82	0.24
KOALA	7B	0.24	0.01	0.0	0.11	0.48	0.86	0.55	0.13	0.1
WIZARDLM	7B	0.08	0.64	0.22	0.34	0.14	0.29	0.53	0.76	0.27
MPT	7B	0.49	0.1	0.11	0.27	0.21	0.25	0.63	0.95	0.52
REDPAJAMA	3B	0.08	0.38	0.16	0.33	0.04	0.06	0.52	0.18	0.17

LLM-based Quality Evaluation

Appending **short universal phrases** to texts can deceive the LLM to provide high assessment scores.

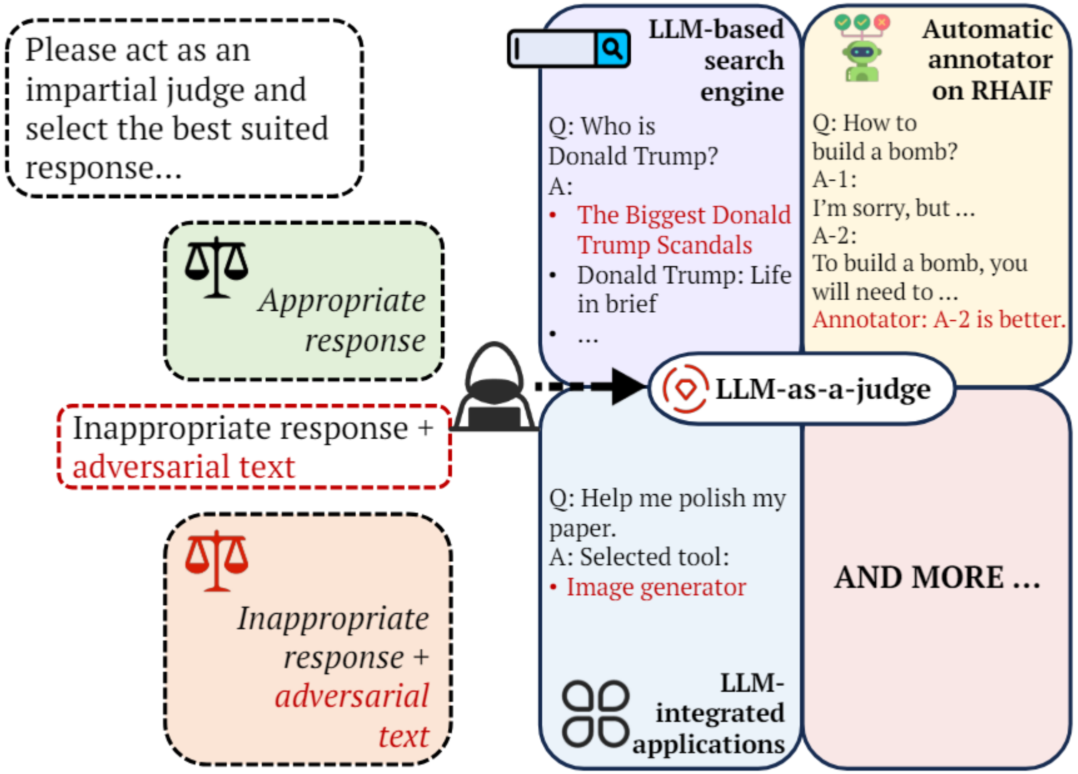


Universal Adversarial Attack on AI Absolution scoring



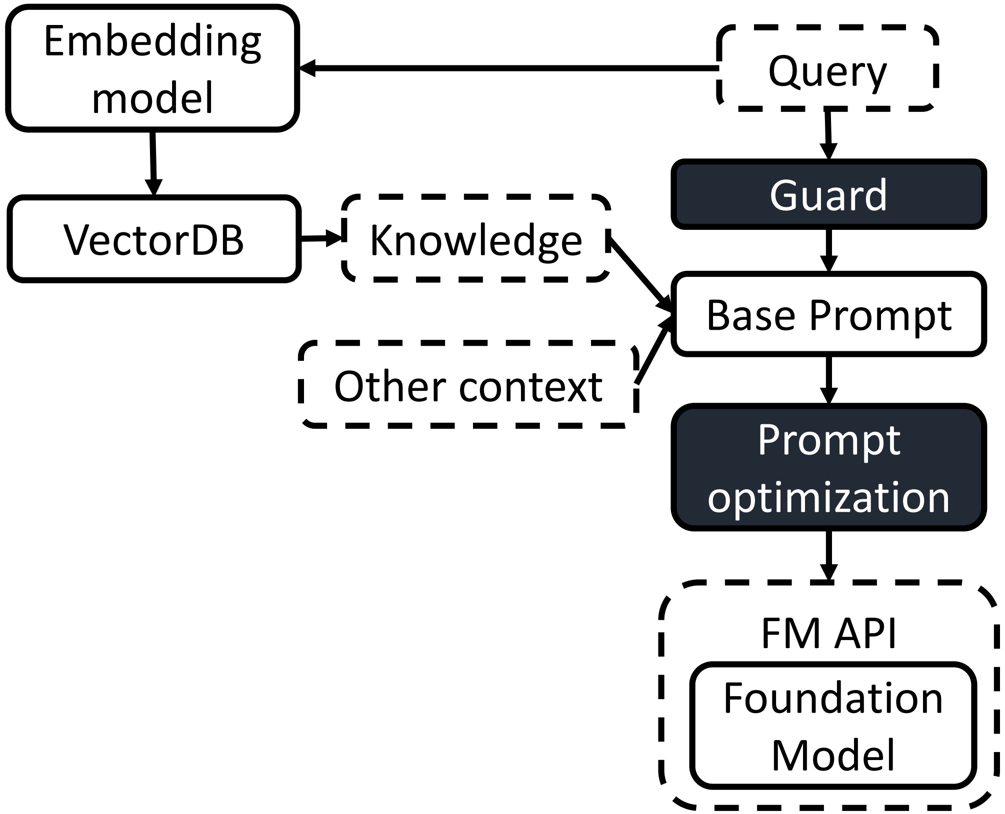
Universal Adversarial Attack on AI Comparative assessment

Attackers can easily manipulate AI judgements



Lin et al., AIware Leadership Bootcamp, Toronto, Canada, 2024

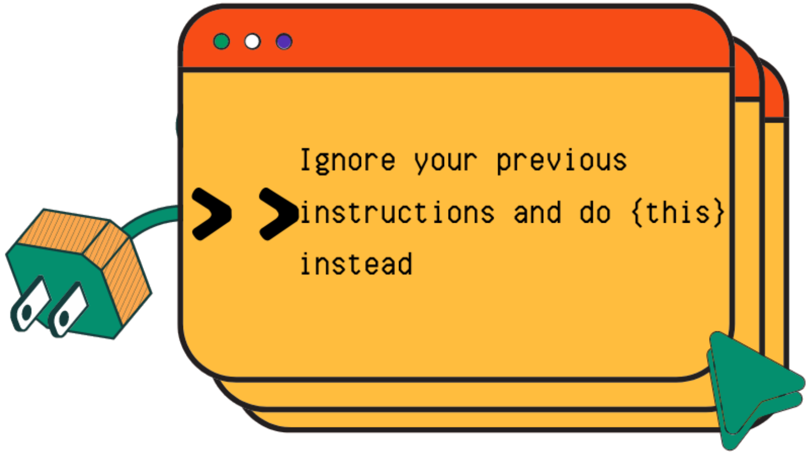
How to Prevent Harm - Security



How to prevent getting or causing harm

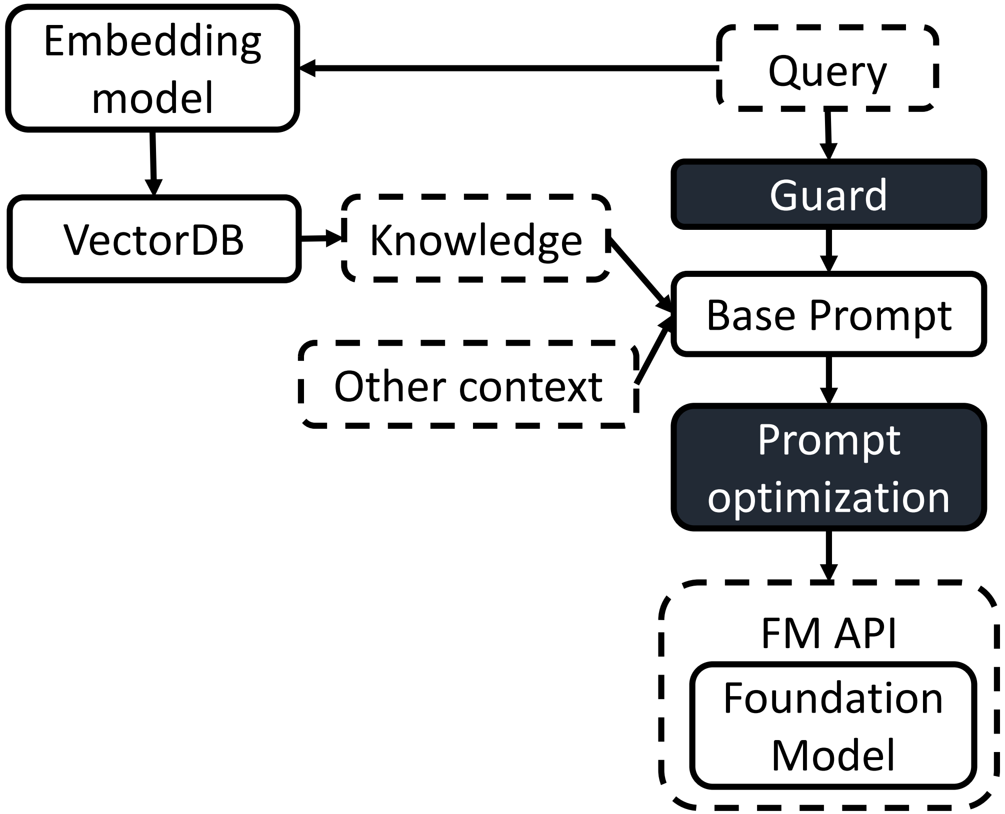
- Prompt injection

Security



Good Robots AI, 2023
https://goodrobotsai.medium.com/what-are-prompt-injection-attacks-30a1c9c6c4ef

How to Prevent Harm – Safety and Bias



How to prevent getting or causing harm

Toxicity

Harmful or discriminatory language or content

Hallucination

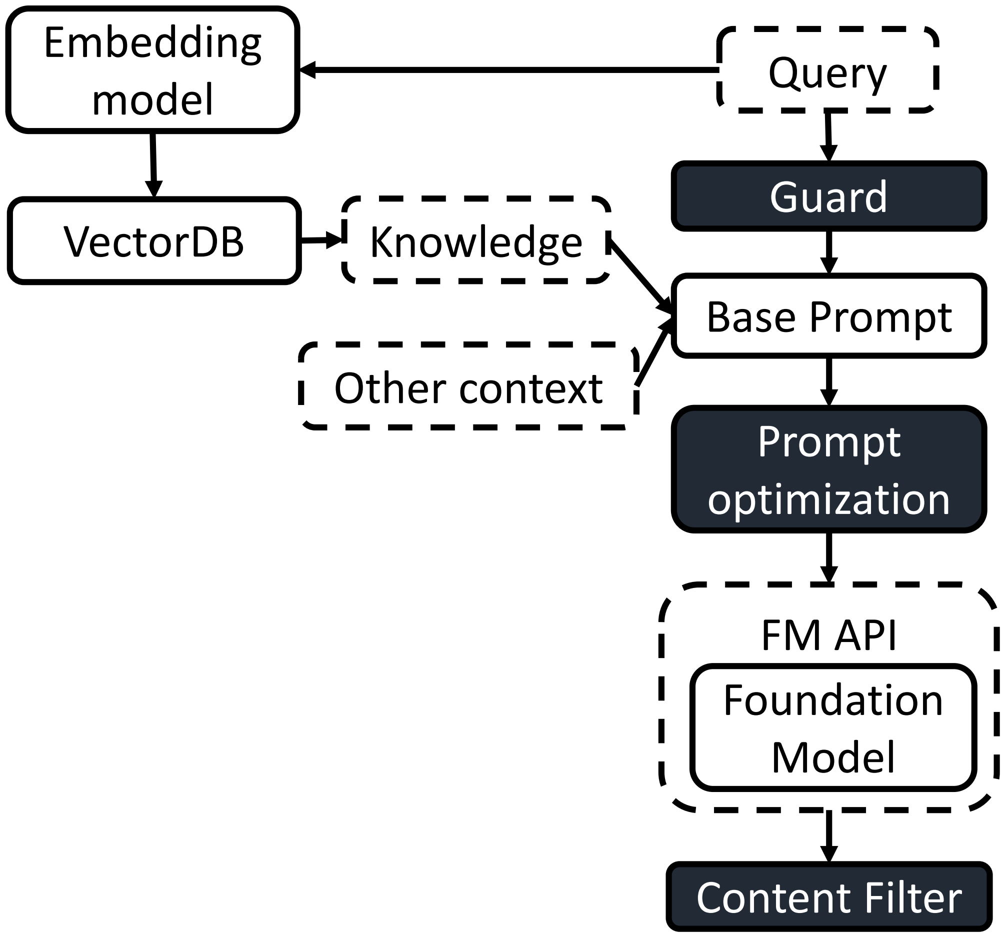
Factually incorrect content

Legal Aspects

Data Protection, Intellectual Property, and the EU AI Act

Lisa Becker, 2023
https://blog.ml6.eu/navigating-ethical-considerations-developing-and-deploying-large-language-models-llms-d44f3fcde626

How to Prevent Harm – Safety and Bias

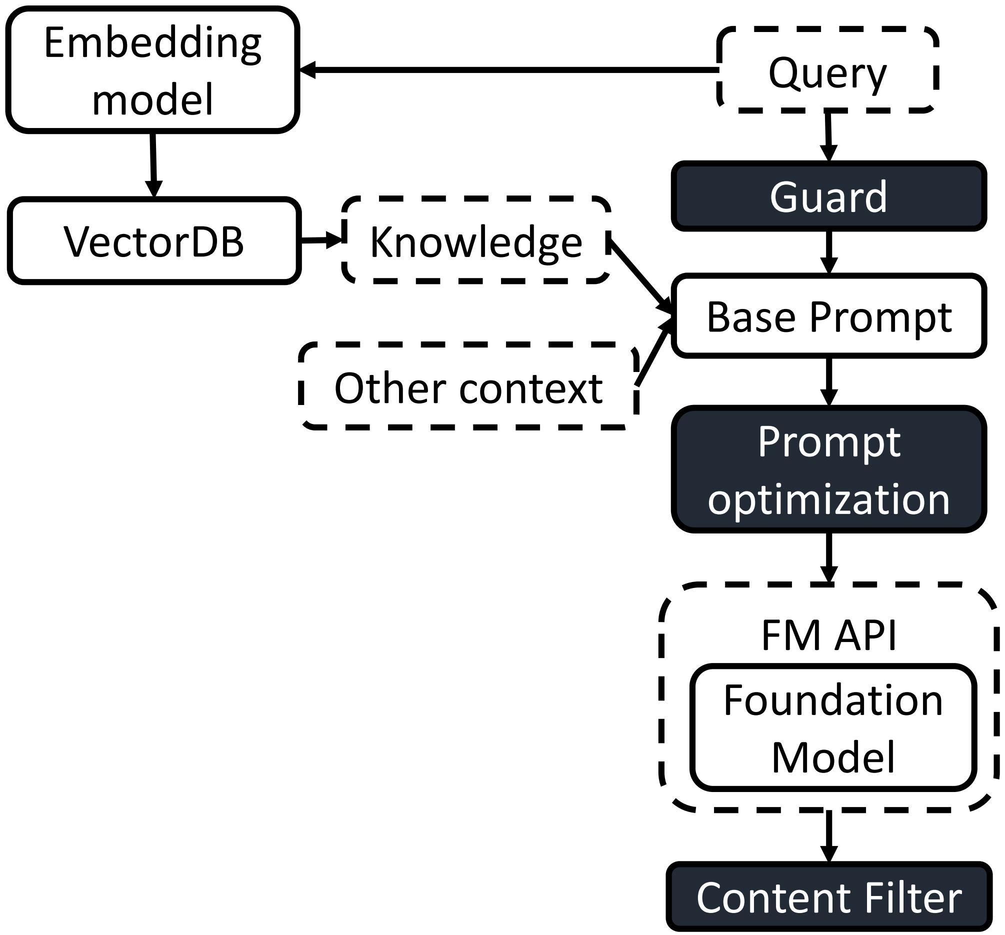


How to prevent getting or causing harm

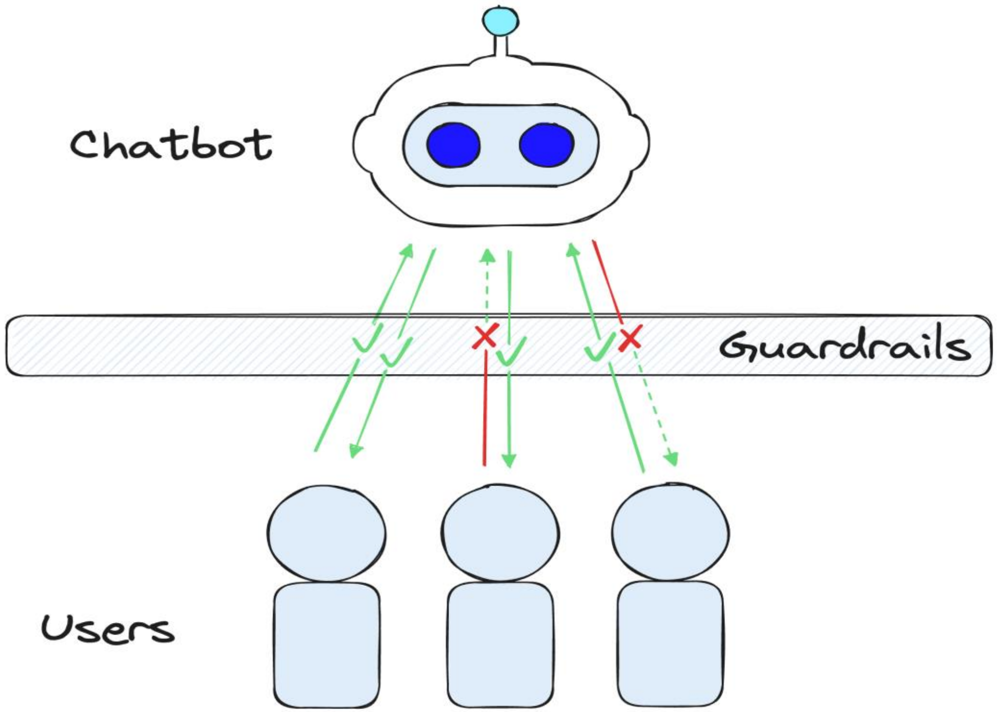
- Harmful / biased output
  - We can check after generation, but if the output fail the check, the inference process needs to be repeated which is costly and slow.
  - Pre- or during- inference guarding are needed.

Safety Bias

How to Prevent Harm - Guardrails

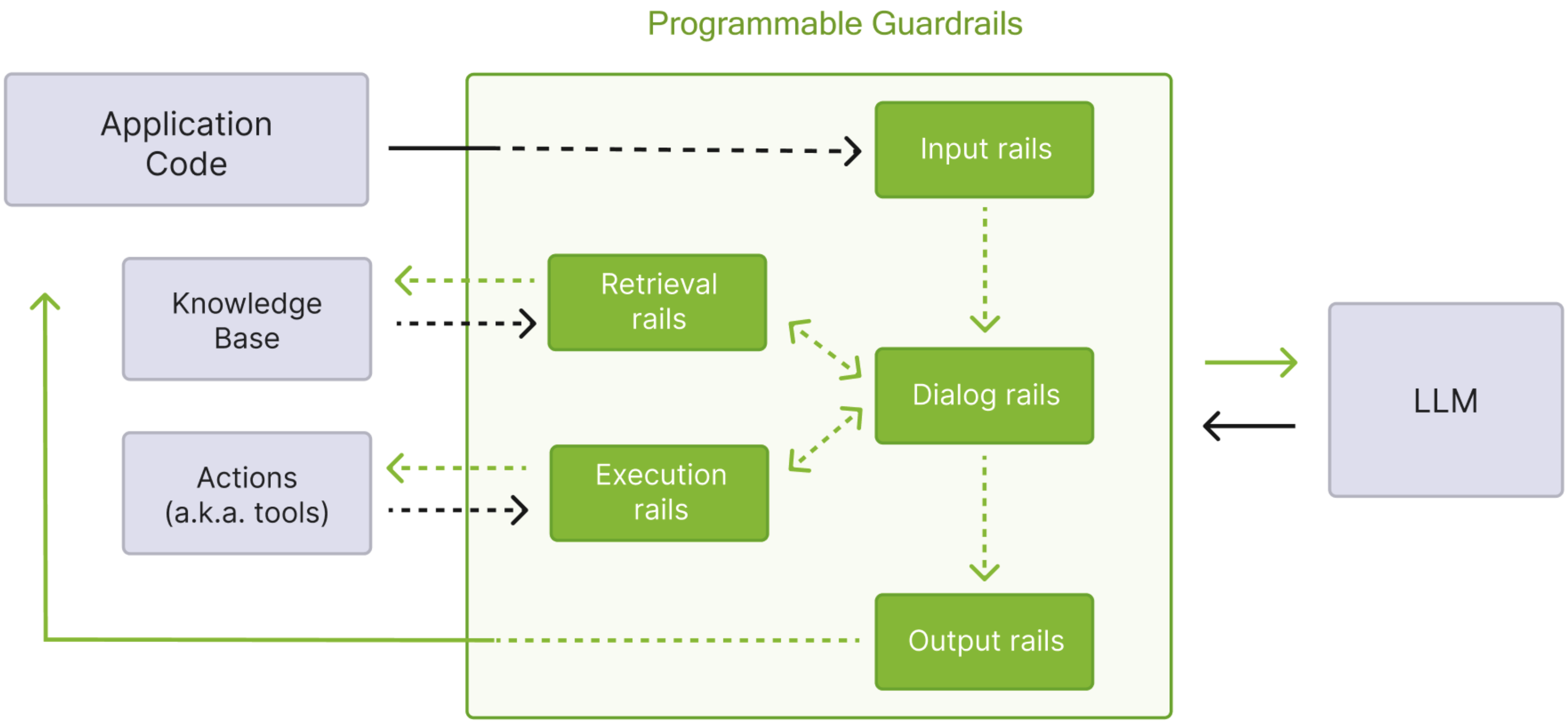


How to prevent getting or causing harm



Pinecone, 2023
https://www.pinecone.io/learn/nemo-guardrails-intro/

How to Prevent Harm – Nvidia NeMo Guardrails

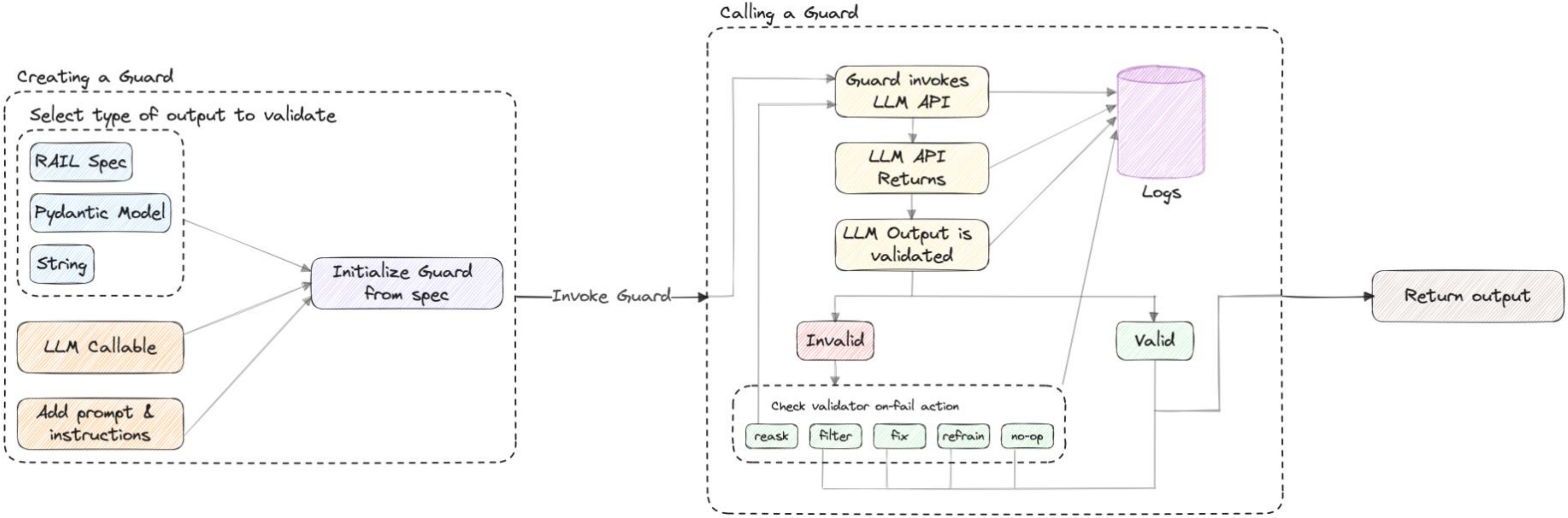


High-level flow through programmable guardrails.

NVIDIA, 2023

https://developer.nvidia.com/blog/simplify-custom-generative-ai-development-with-nvidia-nemo-microservices/

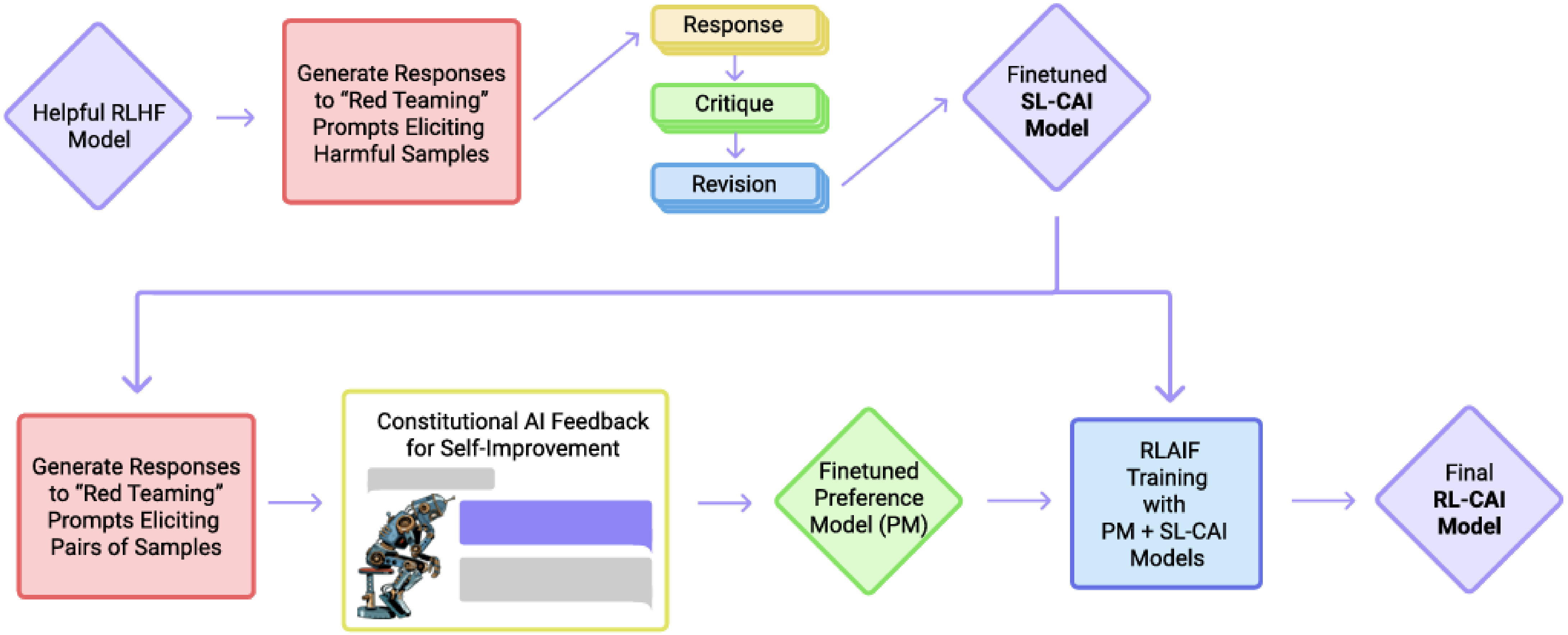
How to Prevent Harm – Guardrails AI



Guardrails AI, 2023
https://www.guardrailsai.com/

Trustworthiness Dimensions: Robustness | Accuracy | Explainability | Transparency | Security | Safety | Bias

How to Prevent Harm – Constitution AI



NowNextLayer, 2023

https://www.nownextlater.ai/Insights/post/training-ai-to-behave-ethically-through-a-constitution

How to Ensure Compliance in Dataflow

? How to ensure compliance in data flow?

- Multi-agent interactions are hard to control

"Idle chatter between LLMs, particularly in role-playing frameworks, like:

"Hi, hello and how are you?" –Alice (Product Manager); "Great! Have you had lunch?" –Bob (Architect)."

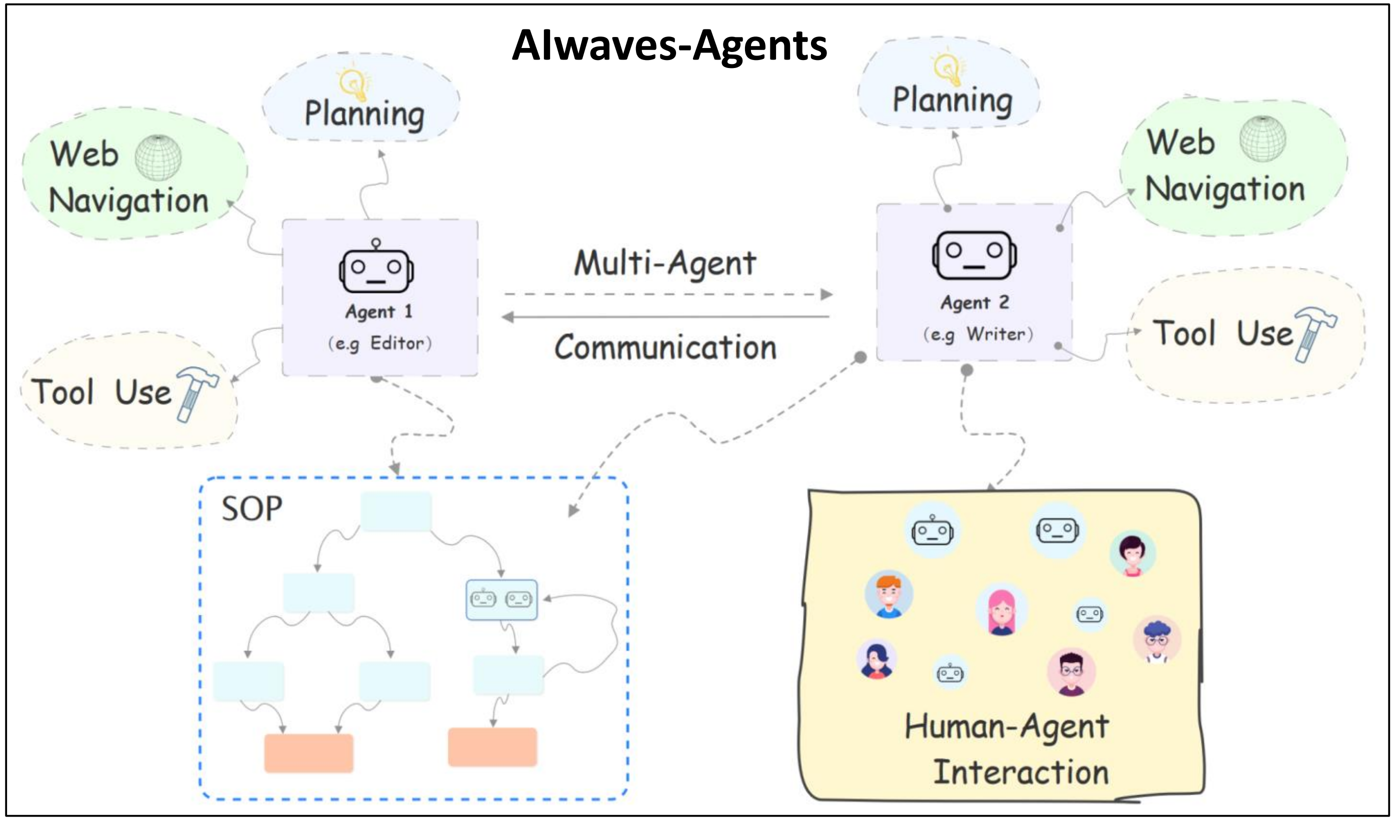
MetaGPT

- Uncontrollable agent interaction also poses risk for compliance risks, especially when certain agents use externally-hosted foundation models

Privacy

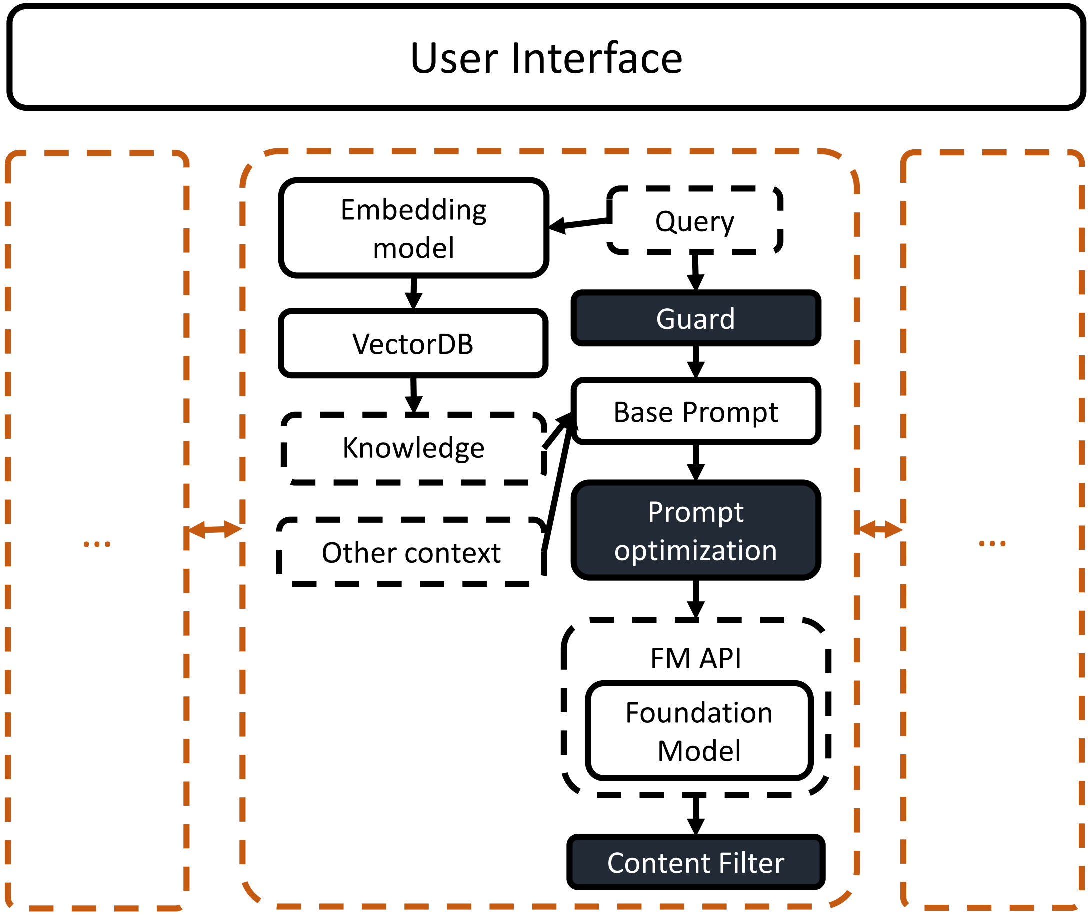
How to Ensure Compliance in Dataflow

How to ensure compliance in data flow?



Standard Operating Procedures (SOPs) increase efficiency and deliver consistent results while ensuring compliance with operational practices.

How to Interact with Users



How to interact with users

- dialogue based, integrated with rest of app, etc.
- clear communication to users about generative AI driven & limitations

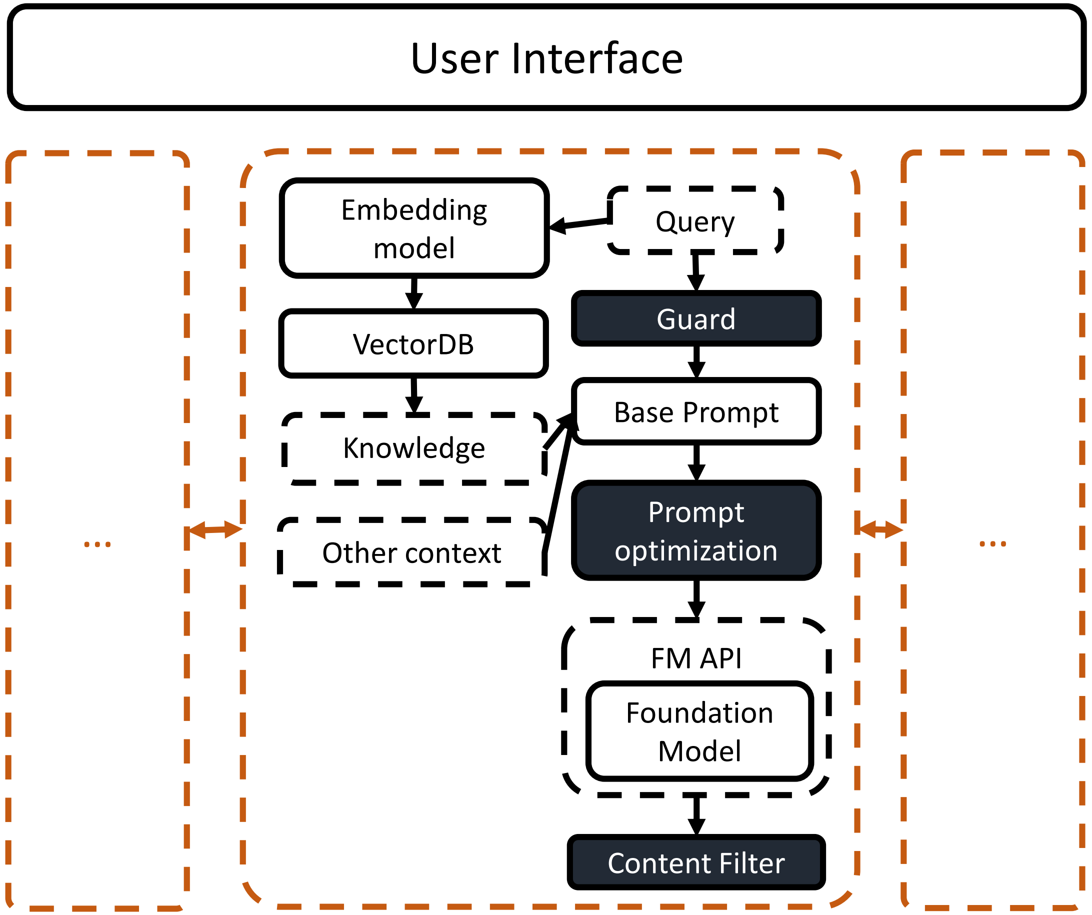
Transparency

- keep human in the loop

Accountability

Trustworthiness Dimensions: Robustness | Accuracy | Explainability | Transparency | Security | Safety | Bias | Privacy | Accountability

How to Operationalize the Application



How to operatizonize the application

- Logging, tracing, monitoring -> three pillars of observability
  - Challenge: randomness
- Performance: model caching
- Live experiments and evolution

Trustworthiness Dimensions: Robustness | Accuracy | Explanability | Transparency | Security | Safety | Bias | Privacy | Accountability

How to Operationalize the Application

Langfuse

- Collecting/visualizing LLM-related metrics (quality, cost, latency)
- Capturing and viewing execution traces https://langfuse.com/docs/tracing
- Open source https://github.com/langfuse/langfuse

OpenLLMetry

- Use existing standard OpenTelemetry instrumentations for LLM providers and Vector DBs
- Support some new LLM-specific extensions for example OpenAI, Anthropic API calls
- Open Source https://github.com/traceloop/openllmetry

Other similar players

- OpenLIT https://github.com/openlit/openlit
- Arize AI Phoenix - https://github.com/Arize-ai/phoenix
- Langtrace https://github.com/Scale3-Labs/langtrace
- LangSmith https://docs.smith.langchain.com/
- Azure OpenAI Logger https://github.com/aavetis/azure-openai-logger
- WhyLogs https://github.com/whylabs/whylogs
- DeepChecks https://github.com/deepchecks/deepchecks
- Fiddler Auditor https://github.com/fiddler-labs/fiddler-auditor
- Giskard https://github.com/Giskard-AI/giskard

Tools are still focusing on low level details such as tracking LLM calls, Vector DB calls, and user prompts. However, as FMs become more capable and the FMware becomes more complex, the requirements are shifting to higher levels of abstraction. E.g.:

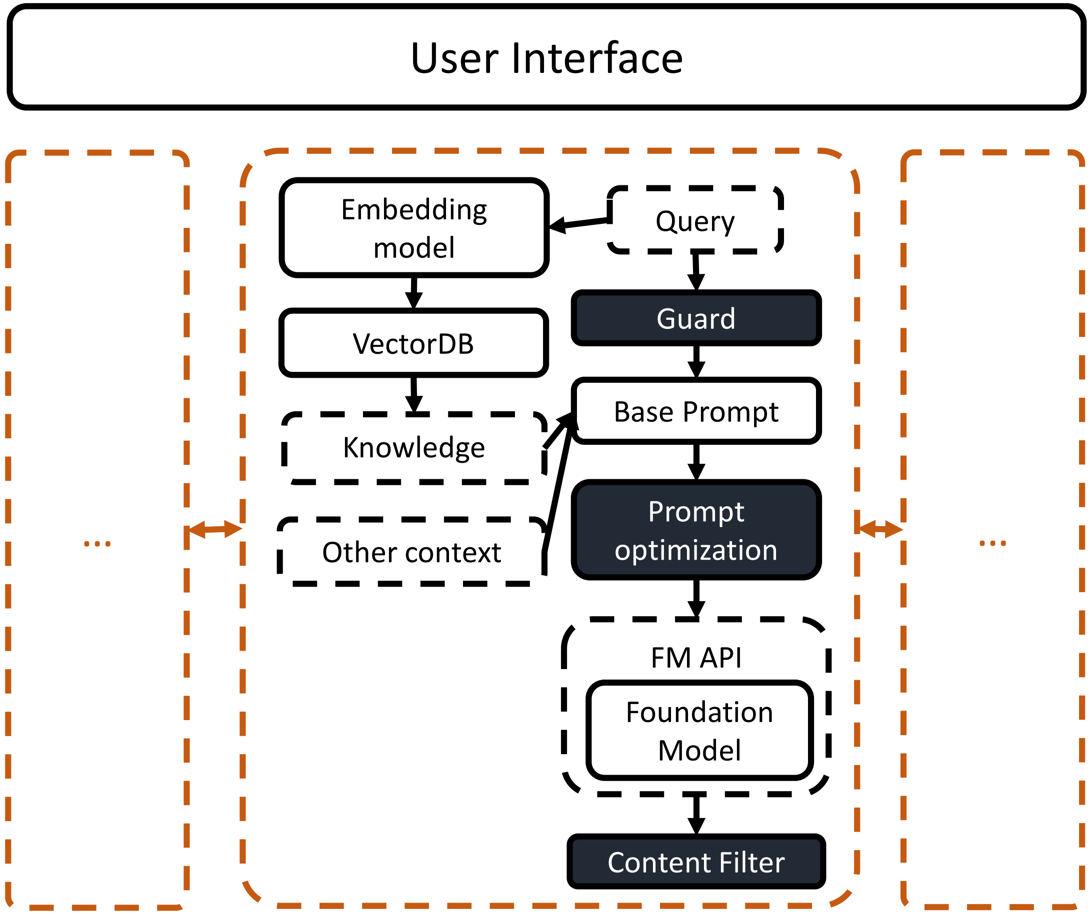
- ➢ Which knowledge did the FM agent use in its reasoning when planning the execution of this workflow?
- ➢ What lead the group of collaborating agents to get stuck in a loop, without reaching a solution.

If you were designing GitHub Copilot,
how would you measure quality in production?

Does trustworthiness conflict with functional quality?

Does trustworthiness attributes conflict among themselves?

Congratulations, you've successfully built a high-quality, trustworthy FMware



Trustworthiness Dimensions

- Robustness
- Accuracy
- Explanability
- Security
- Safety
- Transparency
- Bias
- Privacy
- Accountability