The future with AI (5)

15 feb., 18:30 – 20:00
Cluj-Napoca, Strada Teodor Mihali 62

> Let’s talk about “prompt engineering”

curs-ml.com
Machine Learning Course

TECH 'N TRADE

Agenda

- Introduction
- Community news
- AI News
- Tech review
- (Break)
- Let’s talk about “prompt engineering”
- (Networking)

curs-ml.com
Machine Learning Course

TECH 'N TRADE

Community News

Community news

- GitHub repository with all the codes an presentation

- Discord server (free to join) - https://discord.gg/qd687uSW

GitHub

AI news

AI News

- DeepSeek Coder
- OpenLLMetry-JS
- MiQU-1 70b (Mixtral Medium Model)
- Self Rewarding Language Models
- BioDrone (China)
- Qwen-VL (China)

DeepSeek Coder

- GitHub Repo
- Copilot Alternative
- Demo

OpenLLMetry-JS

- GitHub Repo

+

+

- ✅ Traceloop
- ✅ Dynatrace
- ✅ Datadog
- ✅ New Relic
- ✅ Honeycomb
- ✅ Grafana Tempo
- ✅ HyperDX
- ✅ SigNoz
- ✅ Splunk

MiQU-1 70b (Mistral Medium)

- MiQU-1 70b (Mistral Medium Model)

Rank	Model	Arena Elo	95% CI	Votes	Organizatior
1	GPT-4-0125-preview	1253	+10/-11	3922	OpenAI
2	GPT-4-1106-preview	1252	+5/-6	35385	OpenAI
3	Bard (Gemini Pro)	1224	+9/-9	9081	Google
4	GPT-4-0314	1190	+5/-6	18945	OpenAI
5	GPT-4-0613	1162	+4/-5	29950	OpenAI
6	Mistral Medium	1150	+6/-7	15447	Mistral
7	Claude-1	1149	+6/-6	18189	Anthropic
8	Claude-2.0	1132	+6/-7	12131	Anthropic

Self-Rewarding Language Models

- RLHF without the RL (Facebook)
  - Classification with HF (DPO)
  - Classification with LLM Feedback (SRLM)

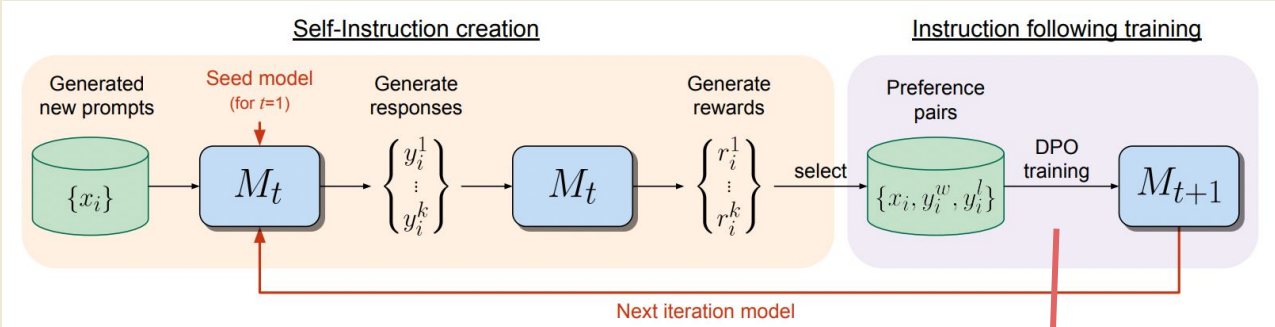


Figure 1: **Self-Rewarding Language Models.** Our self-alignment method consists of two steps: (i) *Self-Instruction creation*: newly created prompts are used to generate candidate responses from model M_t, which also predicts its own rewards via LLM-as-a-Judge prompting. (ii) Instruction following training: preference pairs are selected from the generated data, which are used for training via DPO, resulting in model M_{t+1}. This whole procedure can then be iterated resulting in both improved instruction following and reward modeling ability.

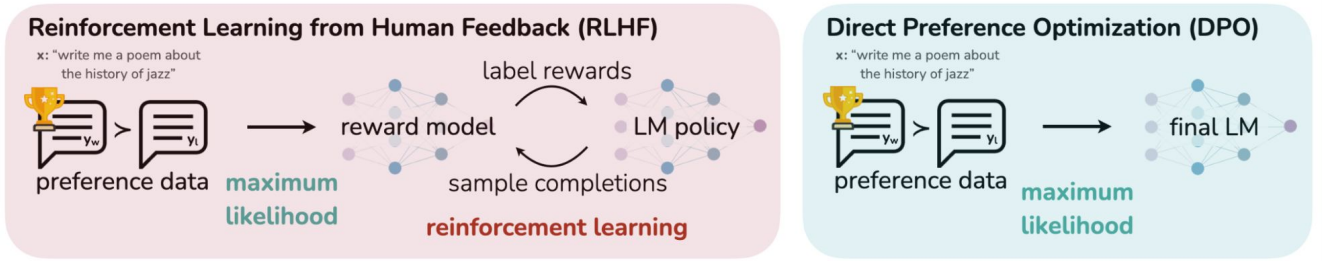


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

BioDrone (China)

- GitHub Repo
- Drone Dataset (142 GB)

Qwen-VL (China)

- GitHub Repo
- Visual Language Model (GPT4-V)
- Demo

Model	DocVQA	ChartQA	AI2D	TextVQA	MMMU	MathVista	MM-Bench-CN
Other Best Open-source LVLM	81.6% (CogAgent)	68.4% (CogAgent)	73.7% (Fuyu-Medium)	76.1% (CogAgent)	45.9% (Yi-VL-34B)	36.7% (SPHINX-V2)	72.4% (InternLM-XComposer-VL)
Gemini Pro	88.1%	74.1%	73.9%	74.6%	47.9%	45.2%	74.3%
Gemini Ultra	90.9%	80.8% [1]	79.5% [1]	82.3% [1]	59.4% [1]	53.0% [1]	-
GPT-4V	88.4%	78.5%	78.2%	78.0%	56.8%	49.9%	73.9%
Qwen-VL-Plus	91.4%	78.1%	75.9%	78.9%	45.2%	43.3%	68.0%
Qwen-VL-Max	93.1% [1]	79.8% [2]	79.3% [2]	79.5% [2]	51.4% [3]	51.0% [2]	75.1% [1]

Tech news

Ray-Ban Meta

- Spy camera glasses
  - Photos
  - Videos
- Live-streaming (on Facebook)
- Speakers / Microphone
- Llama 2 (AI) Assistant
- (not sold in EU…)

(Break)

Let’s talk about “prompt engineering”

Everyone does LLM these days..

Quick recap of an LLM

Next-Token Prediction in Math

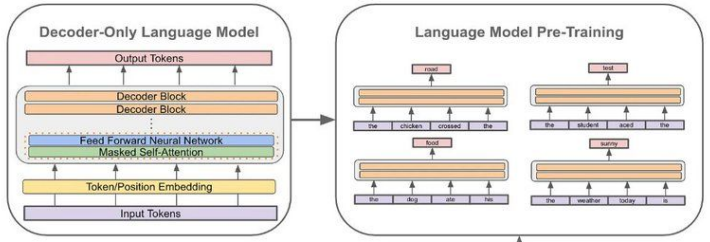
$$\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$$

$$\mathcal{L}(\mathcal{U}) = \sum_{i=1}^{N} \log\left(\mathbb{P}(u_i | u_{i-k}, \ldots, u_{i-1}, \Theta)\right)$$

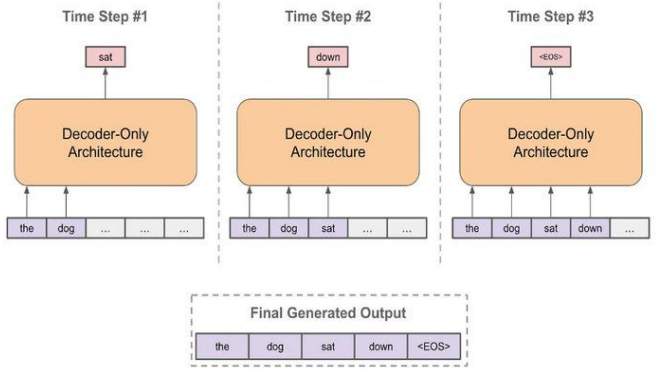
Language model loss over the full text corpus

Conditional probability of i-th token given k preceding tokens and model parameters θ

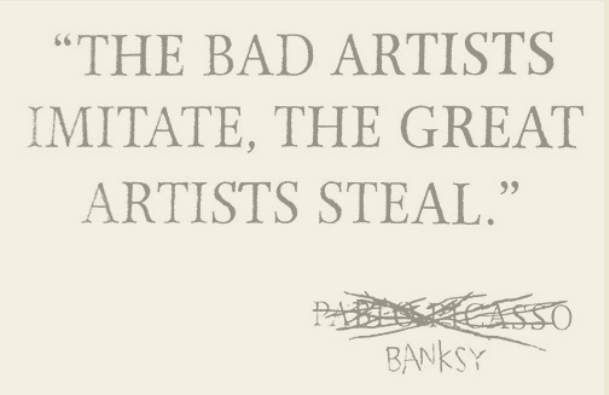
Pre-Training with Next-Token Prediction



Autoregressive Decoding



What is the “Prompt”



- It’s the input text.
- But not all input texts are equal.

“All ~~models~~ prompts are wrong, but some are useful.”

- ~~George Box~~ Cristian Lungu

(correct) Prompt format

```
<s> [INST] QUERY_1 [/INST] ANSWER_1</s> [INST] QUERY_2 [/INST]
ANSWER_2</s>...
```

- Specific to each model

Use-case: Classification (ChatGPT)

```
{job_posting}
---------------

Is this job (A) a job fit for a recent graduate, or
(B) a job requiring more professional experience?"""
```

Use-case: Classification

```
system = "You are Frederick, an AI expert in career advice. You
are tasked with sorting through jobs by analysing their content
and deciding whether they would be a good fit for a recent
graduate or not."

user 1 = """A job is fit for a graduate if it's a
junior-level position that does not require extensive prior
professional experience.
When analysing the experience required, take into account that
requiring internships is still fit for a graduate. I will
give you a job posting and you will analyse it, step-by-step,
to know whether or not it describes a position fit for a
graduate. Got it?"""

assistant 1 = "Yes, I understand. I am Frederick, and I will
analyse your job posting."

user 2 = """Great! Let's begin then :)
For the given job:
{job_posting}
---------------
Is this job (A) a job fit for a recent graduate, or
(B) a job requiring more professional experience.
Answer: Let's think step by step to reach the right
conclusion"""
```

Use-case: Classification

```
system = "You are Frederick, an AI expert in career advice. You
are tasked with sorting through jobs by analysing their content
and deciding whether they would be a good fit for a recent
graduate or not."

user 1 = """A job is fit for a graduate if it's a
junior-level position that does not require extensive prior
professional experience.
When analysing the experience required, take into account that
requiring internships is still fit for a graduate.
I will give you a job posting and you will analyse it,
step-by-step, to know whether or not it describes a position fit
for a graduate. Got it?"""

assistant 1 = "Yes, I understand. I am Frederick, and I will
analyse your job posting."

user 2 = """Great! Let's begin then :)
For the given job:
{job_posting}
--------------
Is this job (A) a job fit for a recent graduate, or
(B) a job requiring more professional experience.
Answer: Let's think step by step to reach the right
conclusion"""
```

Use-case: Classification

– The baseline prompt without any decorations

Persona definition – Define the role / character this AI will play

Task definition – Define the task this AI will solve

Split prompt – Use both the *system* and *user* sections. *System* contains the imperative "who you are".

Zero-shot CoT (Chain of thought) – Prime the assistant as it is

Name – Give the LLM a name

Reiterate – Repeat certain main, important instructions.

Positive Feedback – Reasure that the above instructions are correct and to be followed.

Edge cases – ...

Be right! – Ask the model to find the correct solution, not *any* solution

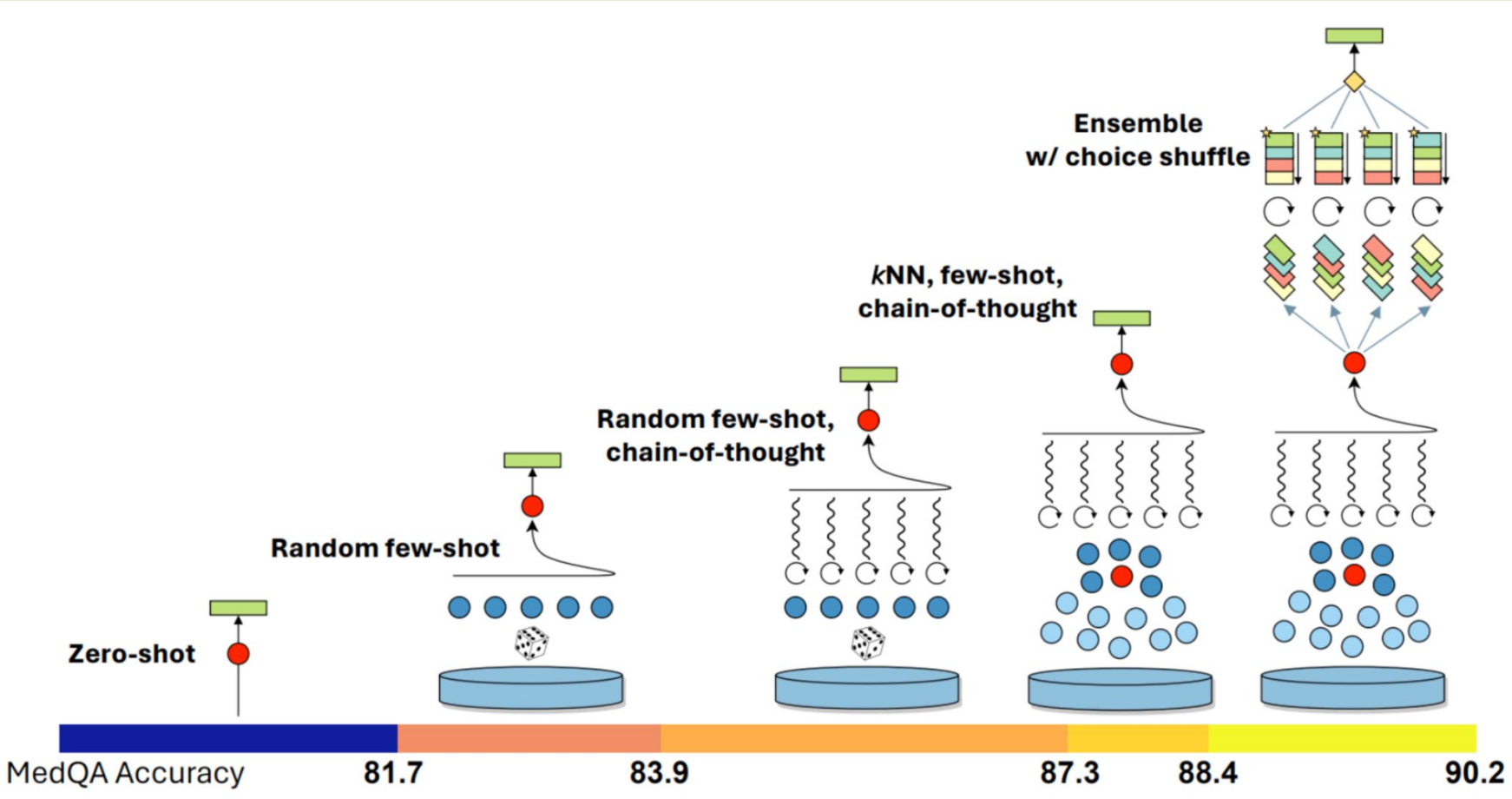
Mocked Exchange – Break down on the prompt into a mocked conversation that the agent should be bound

Use-case: Classification

```
system = """You are Frederick, an AI expert in career advice.
          You are tasked with sorting through jobs by
          analysing their content and deciding whether
          they would be a good fit for a recent graduate or not.""",
user 1 = """A job is fit for a graduate if it's a junior-level
          position that does not require extensive prior
          professional experience.
          When analysing the experience required, take
          into account that requiring internships is still
          fit for a graduate.
          I will give you a job posting and you will analyse
          it, step-by-step, to know whether or not it
          describes a position fit for a graduate.
          Got it?"""
ssistant 1 = """Yes, I understand. I am Frederick, and I will analyse
          your job posting.""",
user 2 = """Great! Let's begin then :)

          For the given job:
          {job_posting}
          ----------

          Is this job (A) a job fit for a recent graduate,
          or (B) a job requiring more professional experience.
          Answer: Let's think step by step
          to reach the right conclusion"""
```

Persona definition
Task definition
Split prompt
Zero-shot CoT (Chain of thought)
Name
Reiterate
Positive Feedback
Edge cases
Be right!
Mocked Exchange

Use-case: MedQA

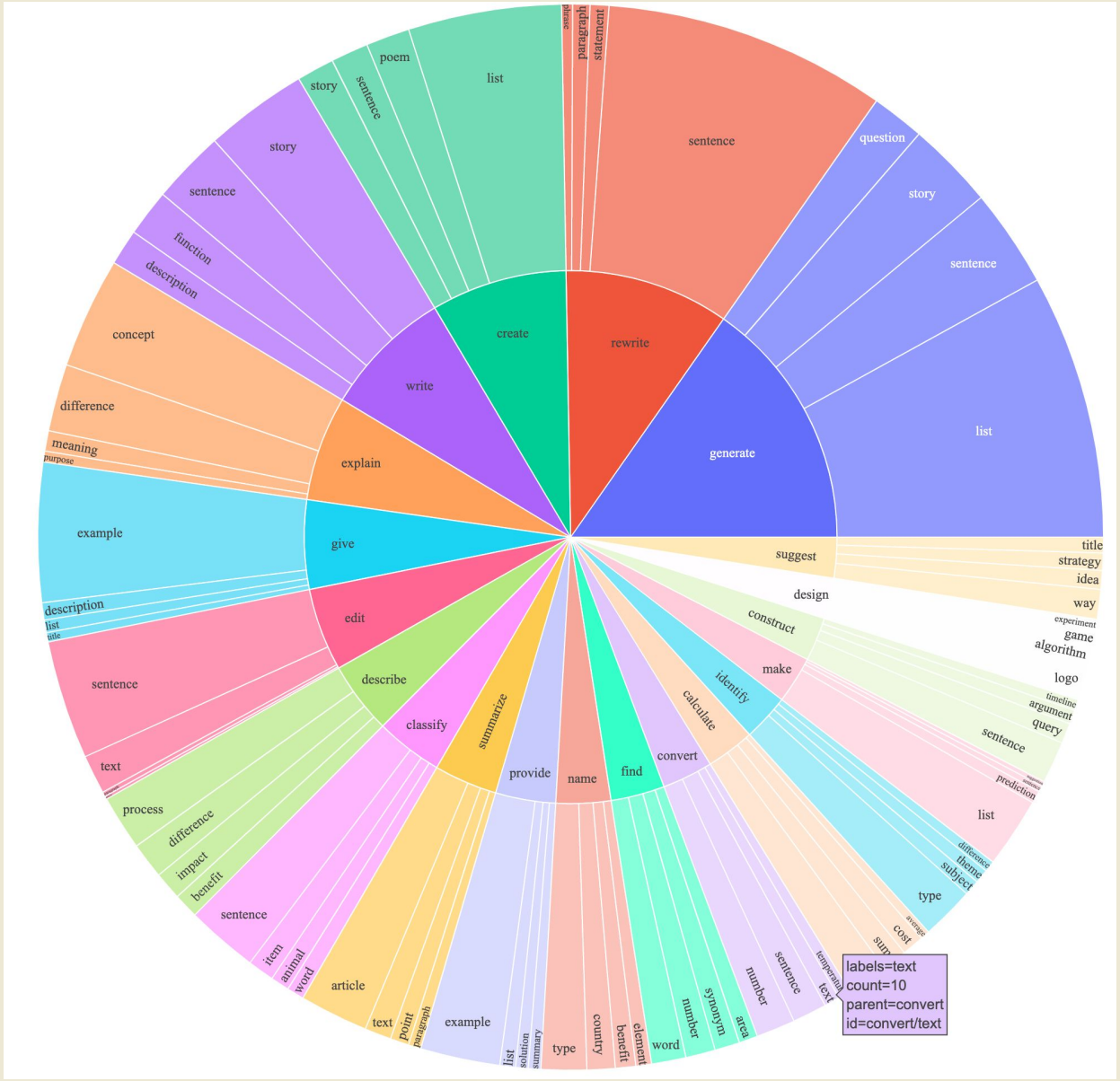


Why do we need to this?

Because of instruction tuning

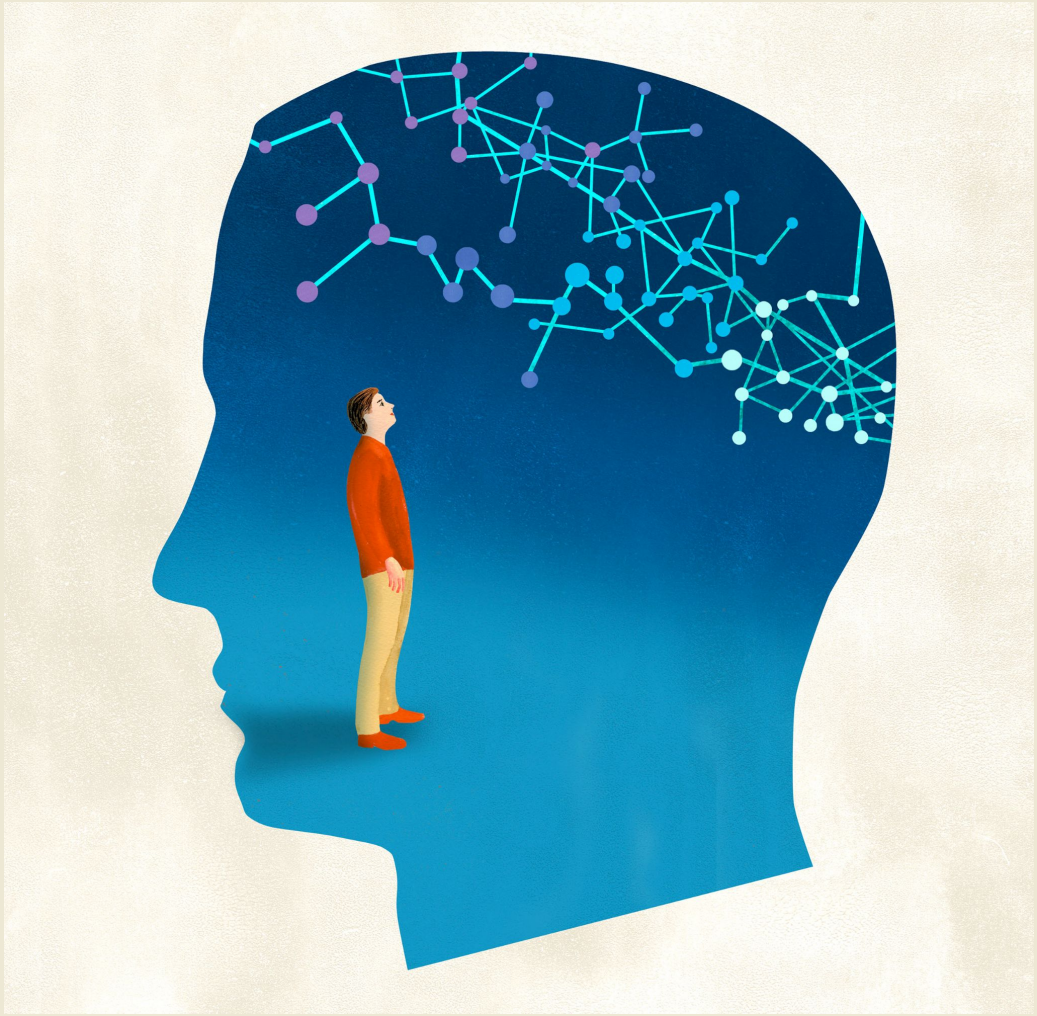
```
### Instruction:
{instruction}

### Input:
{input}

### Response:
```

Because of instruction tuning (Alpaca)



curs-ml.com
Machine Learning Course

Reverse engineering the language patterns of the instructors



Prompt Engineering
is NOT Engineering

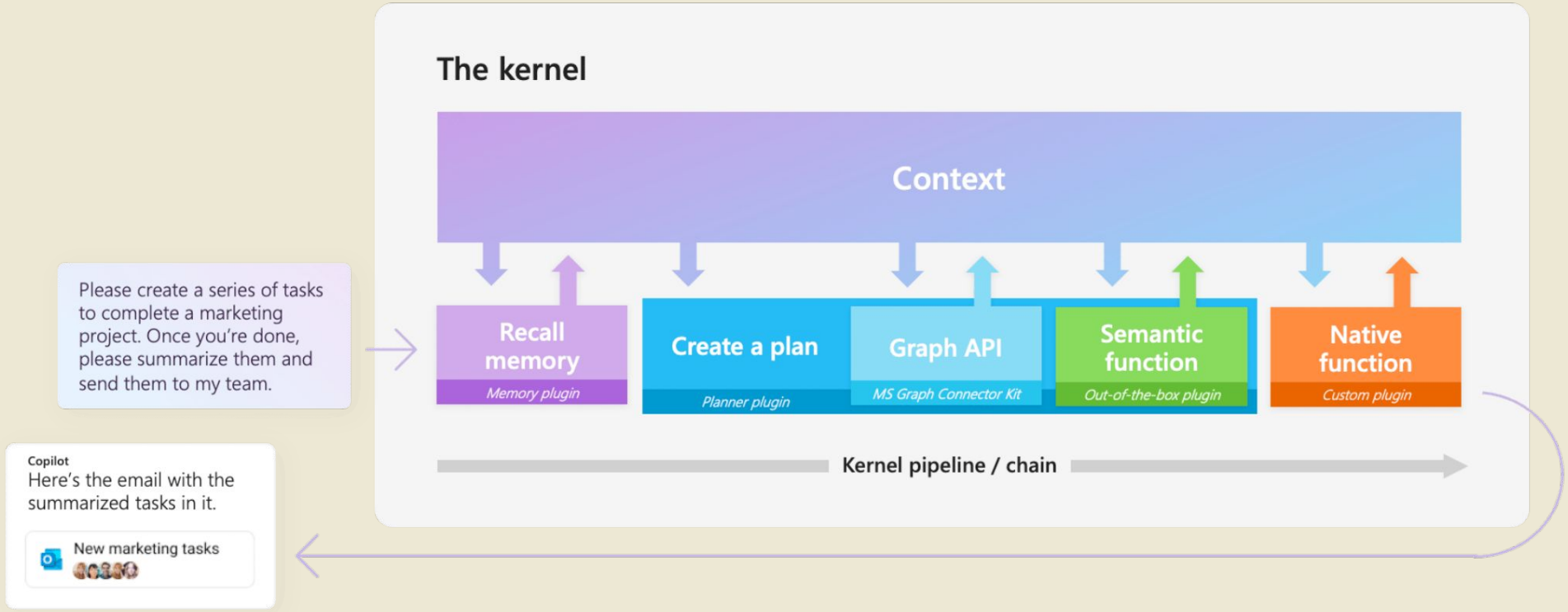
Use-Case: LLM Guardrails - Instructor

```python
import instructor
from openai import OpenAI
from pydantic import BaseModel

# Enables `response_model`
client = instructor.patch(OpenAI())


class UserDetail(BaseModel):
    name: str
    age: int


user = client.chat.completions.create(
    model="gpt-3.5-turbo",
    response_model=UserDetail,
    messages=[
        {"role": "user", "content": "Extract Jason is 25 years old"},
    ],
)

assert isinstance(user, UserDetail)
assert user.name == "Jason"
assert user.age == 25
```

Use-Case: LLM Guardrails - Instructor

```python
if mode == Mode.TOOLS:
    kwargs["messages"].append(
        {
            "role": "tool",
            "tool_call_id": response.choices[0]
            .message.tool_calls[0]
            .id,
            "name": response.choices[0]
            .message.tool_calls[0]
            .function.name,
            "content": f"Recall the function correctly, fix the errors and exceptions found\n{e}",
        }
    )
else:
    kwargs["messages"].append(
        {
            "role": "user",
            "content": f"Recall the function correctly, fix the errors and exceptions found\n{e}",
        }
    )
if mode == Mode.MD_JSON:
    kwargs["messages"].append(
        {
            "role": "assistant",
            "content": "```json",
        },
    )
```

Use-Case: [Semantic Kernel]



Use-Case: Semantic Kernel

```
{{#*inline "RetryLogic"}}
{{~#if lastError}}
{{#message role="system"}}## Previous attempt
This previous plan failed to achieve the goal:
```handlebars
{{lastPlan}}
```

The error was:
```
{{lastError}}
```

Try again to achieve the goal while fixing the error.
{{/message}}
{{/if}}
{{/inline}}
```

Other “esoteric” techniques

- Sparse Language Representation
  - This is a technique for "priming" the LLM in the context prompt (system prompt) with specific words that would allow the attention mechanism for promoting specific **paths** from the llm memory.

  - This is basically a way to condense into the context window (in a token efficient way) information about what the original system prompt is.
- Dynamic Few Shots
- …

Conclusions

- Every model has a different “best prompt”
- Prompts are brittle
- All “best practices” are just hacks that try to mimic the training dataset
- This is not engineering

Thank you!

curs-ml.com
Machine Learning Course

TECH 'N TRADE