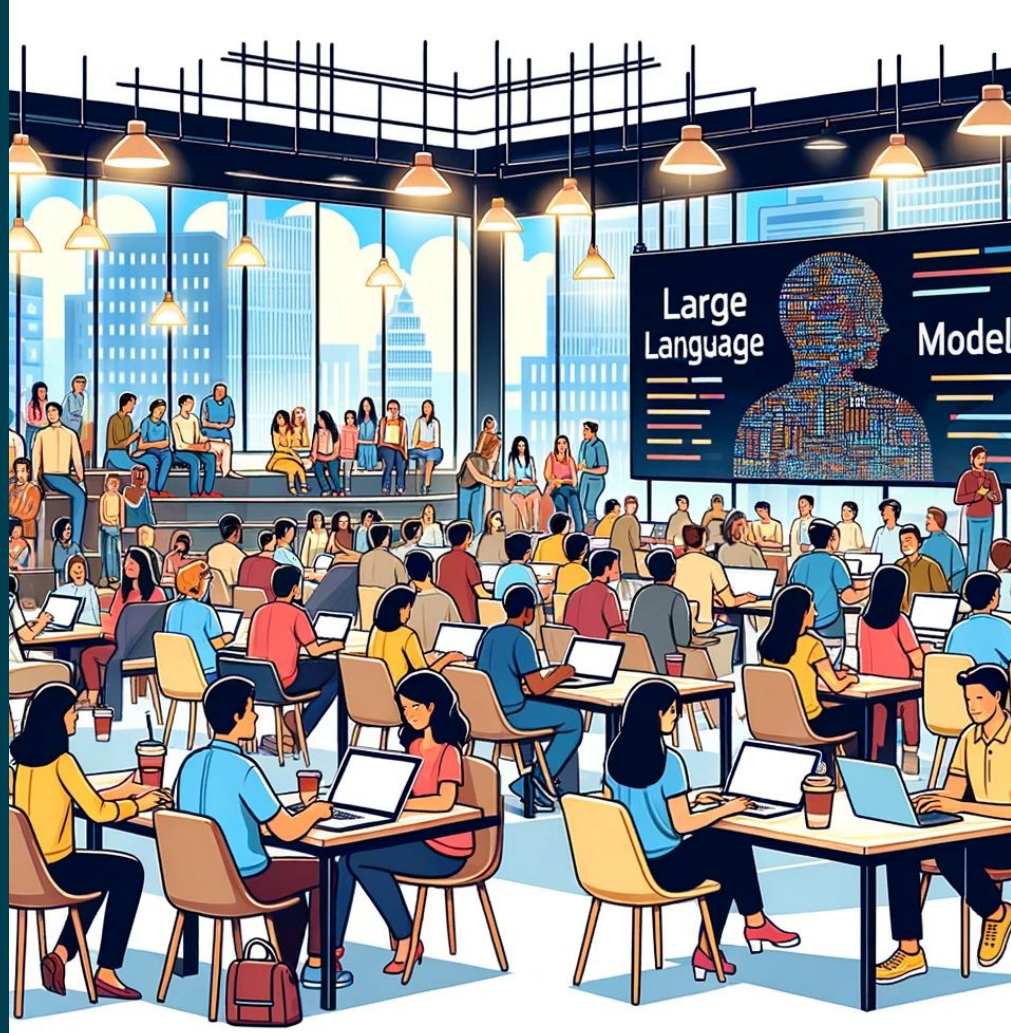
Getting hands on with Large Language Models (LLMs)

MLOps and Crafts Meetup

Wifi password:
skating zebra glowing citrus bottle

/thoughtworks



Temperature check 🌡️

Raise your hands if...

- You have never interacted with LLMs (e.g. ChatGPT) before
- You have used ChatGPT in the past
- You have dabbled with prompt engineering
- You have used open-source LLMs in the past
- You have experience developing LLM applications
- You have successfully put LLM applications into production

Agenda

Setting up: LLMs and prompting 101	6pm
The workshop: Getting hands-on with LLMs	6.20pm
Reflections: What are your takeaways?	7.20pm
Parting ideas: Recommendations for getting started	7.30pm

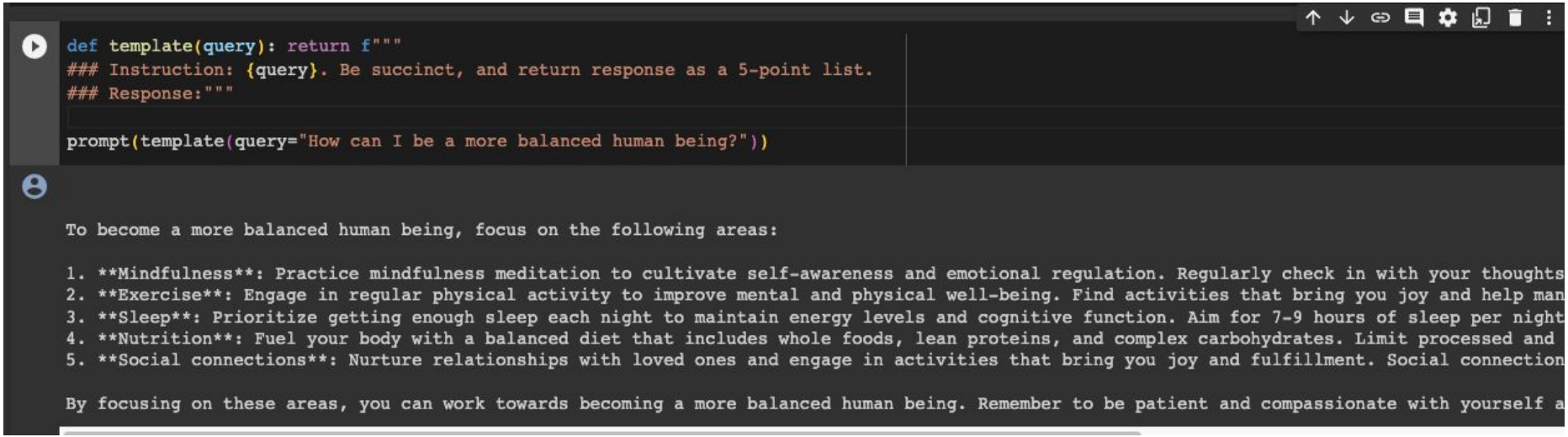
Getting set up for the hands-on exercises

- Pair up
  - Pairing is great for team productivity, accelerated learning and is also more fun!
  - Ideally, if you are non-technical find a technical pair and vice-versa
  - You can do the exercises either on one computer, or on both
- Introductions
  - Intro: your {name, role, company}
  - Icebreaker: What's a (work or non-work) project that you're most proud of?
- Setup
  - Follow instructions on https://github.com/mlops-and-crafts/llm-workshop and in **llmops_and_crafts.ipynb** notebook

Open the main notebook right in google colab using this link.

Smoke test

Sit back and relax once you can run this cell successfully



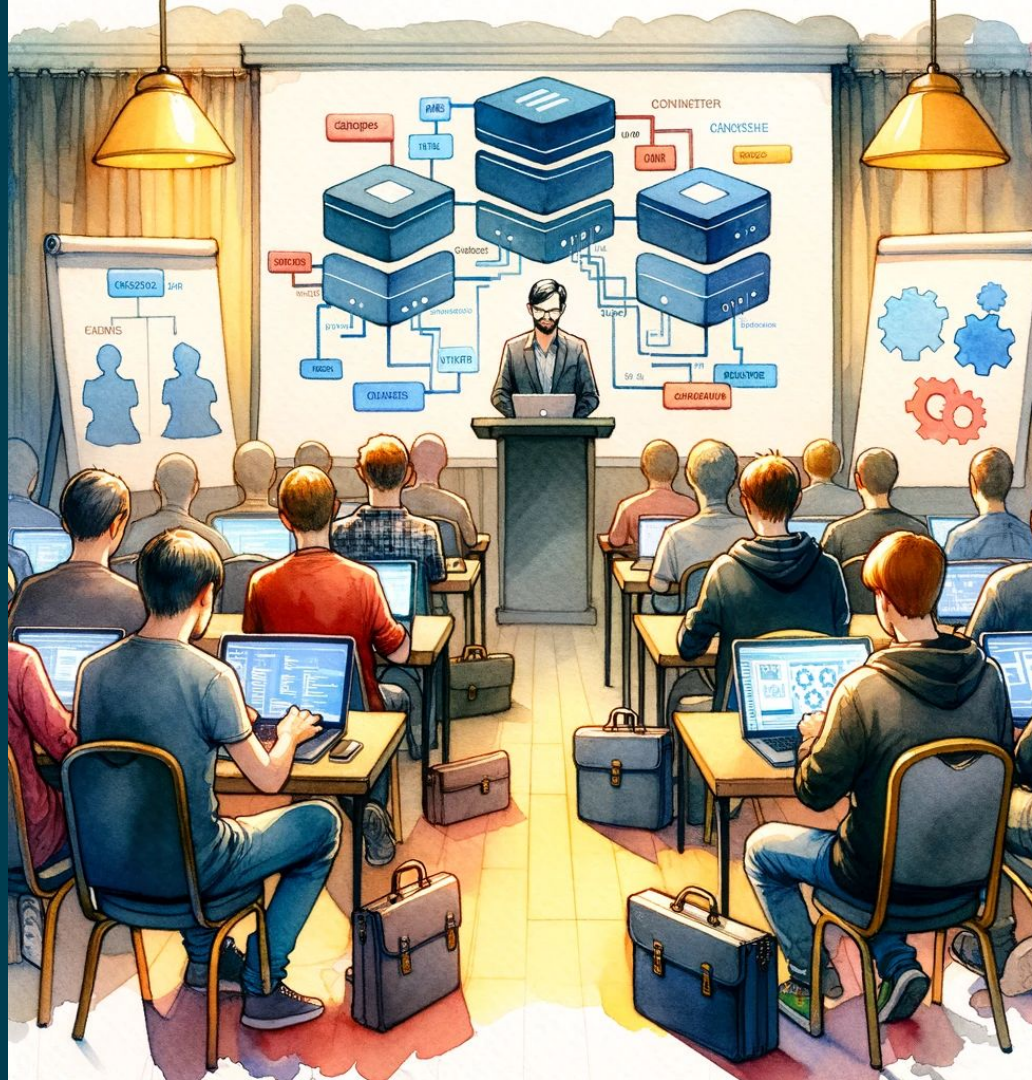
```
def template(query): return f"""
### Instruction: {query}. Be succinct, and return response as a 5-point list.
### Response:"""

prompt(template(query="How can I be a more balanced human being?"))
```

```
To become a more balanced human being, focus on the following areas:

1. **Mindfulness**: Practice mindfulness meditation to cultivate self-awareness and emotional regulation. Regularly check in with your thoughts
2. **Exercise**: Engage in regular physical activity to improve mental and physical well-being. Find activities that bring you joy and help man
3. **Sleep**: Prioritize getting enough sleep each night to maintain energy levels and cognitive function. Aim for 7-9 hours of sleep per night
4. **Nutrition**: Fuel your body with a balanced diet that includes whole foods, lean proteins, and complex carbohydrates. Limit processed and
5. **Social connections**: Nurture relationships with loved ones and engage in activities that bring you joy and fulfillment. Social connection

By focusing on these areas, you can work towards becoming a more balanced human being. Remember to be patient and compassionate with yourself a
```

While you are setting up

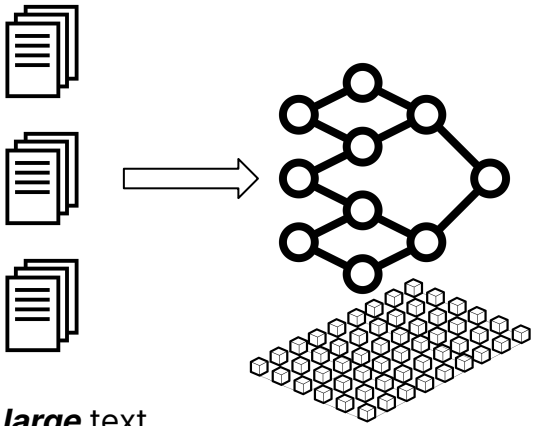
101 on LLMs and prompting

/thoughtworks



Generative* Large language models (LLMs)

* The G in ChatGPT



large text datasets

large neural networks

large amounts of accelerated compute

LLM

new paradigm: ***prompting***

Please rephrase the following text in a more formal tone: ...

Diverse & powerful applications

- *generate*
- *summarise*
- *rewrite*
- *extract*
- *classify*
- *chat*

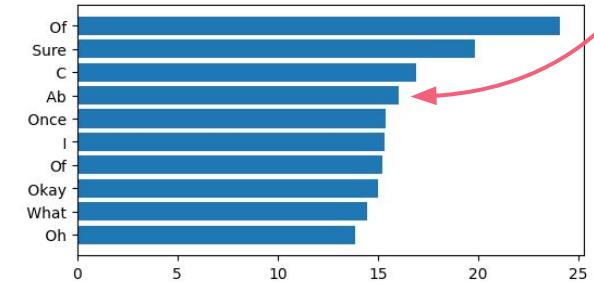
LLMs: What Token Follows?

What does “next token generation” look like?

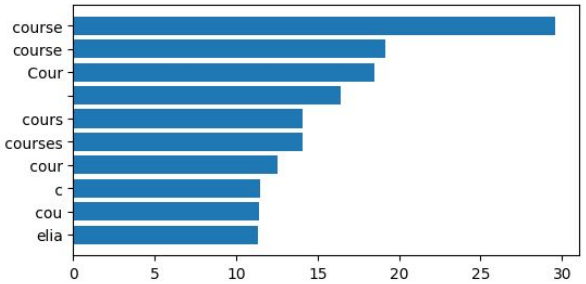
```
### Human: Tell me a story\n### Assistant:
```

From prompt, candidate next tokens by likelihood …

token 1:
#4587
“ Of”



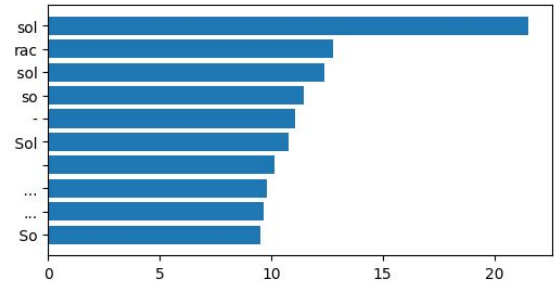
token 2:
#3236
“ course”



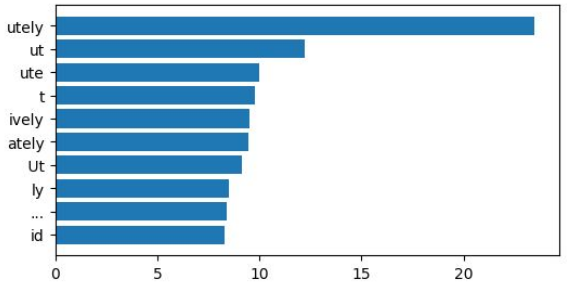

```
### Human: Tell me a story\n### Assistant: Ab
```

Extended prompt includes the less likely token “ **Ab**”…

alt
token 1:
#2929
“sol”

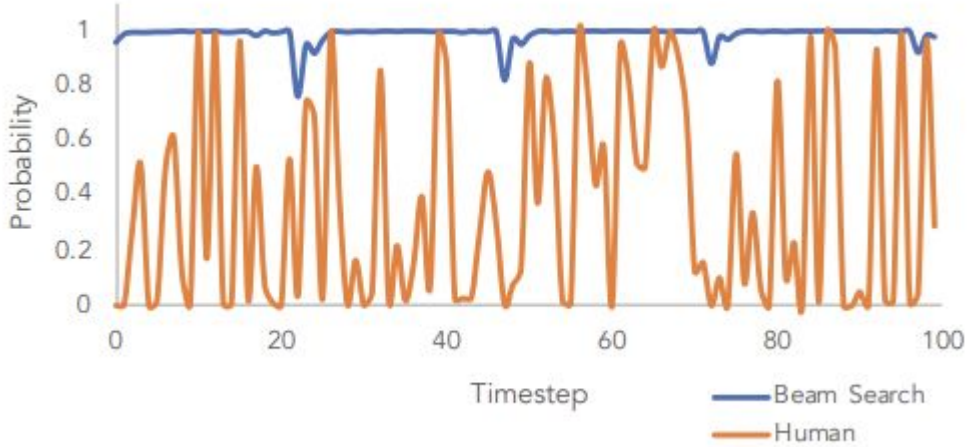


alt
token 2:
#11579
“utely”



https://github.com/safetydave/maths-whimsy/blob/main/ml/llm_wtf_what_token_follows.ipynb

Human language and speech is in fact not so predictable. Generated text that always picks the most likely next token therefore will sound weird or a bit boring.



Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019, September). The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*.

LLM What Token Follows?

Effect of the temperature parameter

	Temperature = 0.0 (identical)	Temperature = 0.8 (typical)	Temperature = 5 (highly varied)
Take 1	Of course! I'd be happy to tell you a story. Can you give me some details about what kind of story you would like to hear? For example, do you want it to be a funny story, a sad story, or an adventure story?"	Of course, I'd be happy to tell you a story! Can you give me some details about what kind of story you would like to hear? For example, do you want it to be funny, sad, scary, or something else entirely?	Of course, I'd be happy to tell you a story! Here is one I just made up on the spot:\n\nOnce upon a time, in a far-off land, there was a small village nestled at the foot of a great mountain. The villagers were known for their exceptional skill
Take 2	Of course! I'd be happy to tell you a story. Can you give me some details about what kind of story you would like to hear? For example, do you want it to be a funny story, a sad story, or an adventure story?"	Of course! I'd be happy to tell you a story. Can you give me some information about what kind of story you would like to hear? For example, do you want it to be funny, sad, or exciting? Do you have any specific themes or characters in mind?	Of course! What would you like the story to be about? Fantasy, adventure, romance, or something else?

https://github.com/safetydave/maths-whimsy/blob/main/ml/llm_wtf_what_token_follows.ipynb

LLM - What Token Follows?

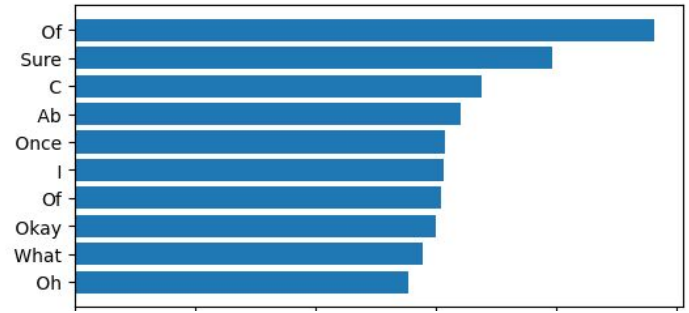
Sampling strategies and temperature

```
### Human: Tell me a story\n### Assistant:
```

Greedy	Top-K	Top-P (aka Nucleus Sampling)
Pick the top token: **“ Of”** Similar result to temp = 0	Pick from the top K tokens **“ Of”, “ Sure”, “ C”** (K=3) according to probabilities	Pick from smallest set of tokens (# varies) exceeding probability threshold P

Greedy is Top-K with K = 1

If we choose K = 7



P = 0.75
3 tokens
(e.g. only)

Small P → few tokens

When used together, P acts after K

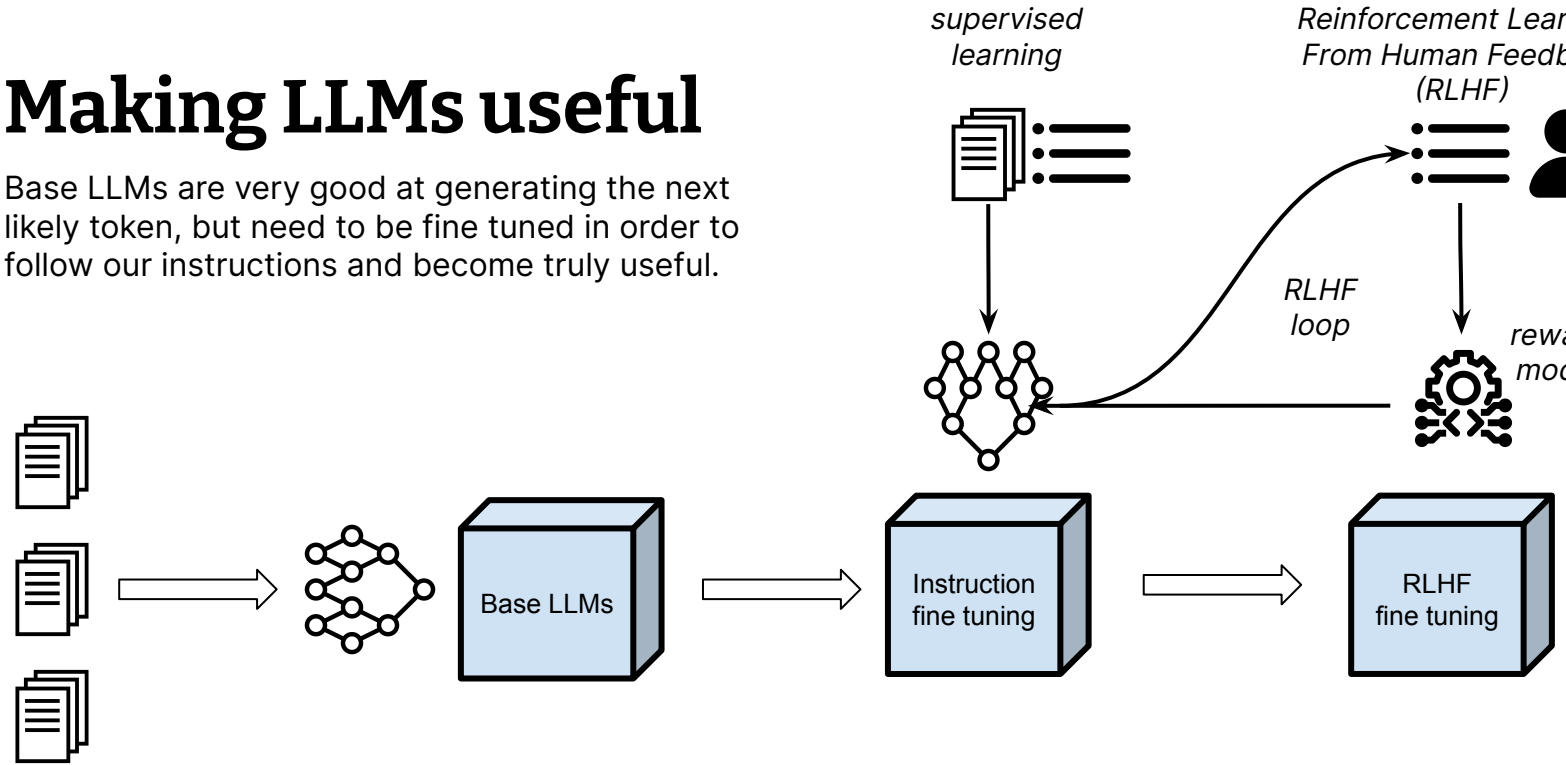
Temperature has no effect

High temperature → higher chance to select less likely tokens

https://docs.cohere.com/docs/controlling-generation-with-top-k-top-p

Making LLMs useful

Base LLMs are very good at generating the next likely token, but need to be fine tuned in order to follow our instructions and become truly useful.



Pre-trained: The P in ChatGPT

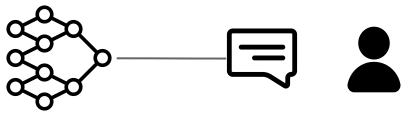
Making a generic LLM useful for your specific task

Start here!

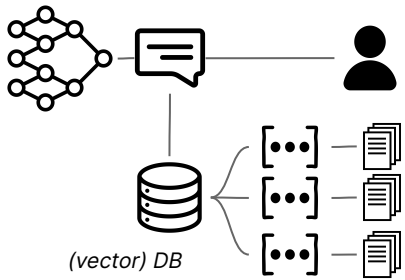
In-context learning

Model is frozen, provide instruction, examples, context in the prompt

Prompting
zero or few-shot



Retrieval Augmented Generation (RAG)
augmenting with embedded documents/media

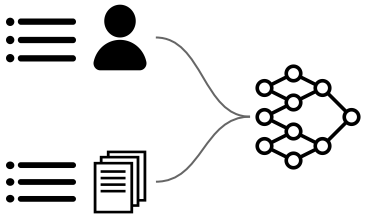


Further model finetuning

Model is unfrozen and updated with examples of desired behaviour

Supervised learning

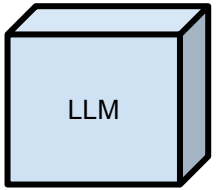
RLHF



For more details, check out:
https://www.promptingguide.ai/

Prompting: zero & few shot learning

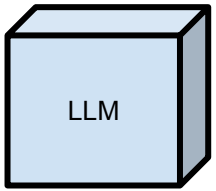
Can you please extract the contact details from this text into a JSON format?



Zero-shot learning

Can you please extract the contact details from this text into a JSON format? First I'll give you some examples:

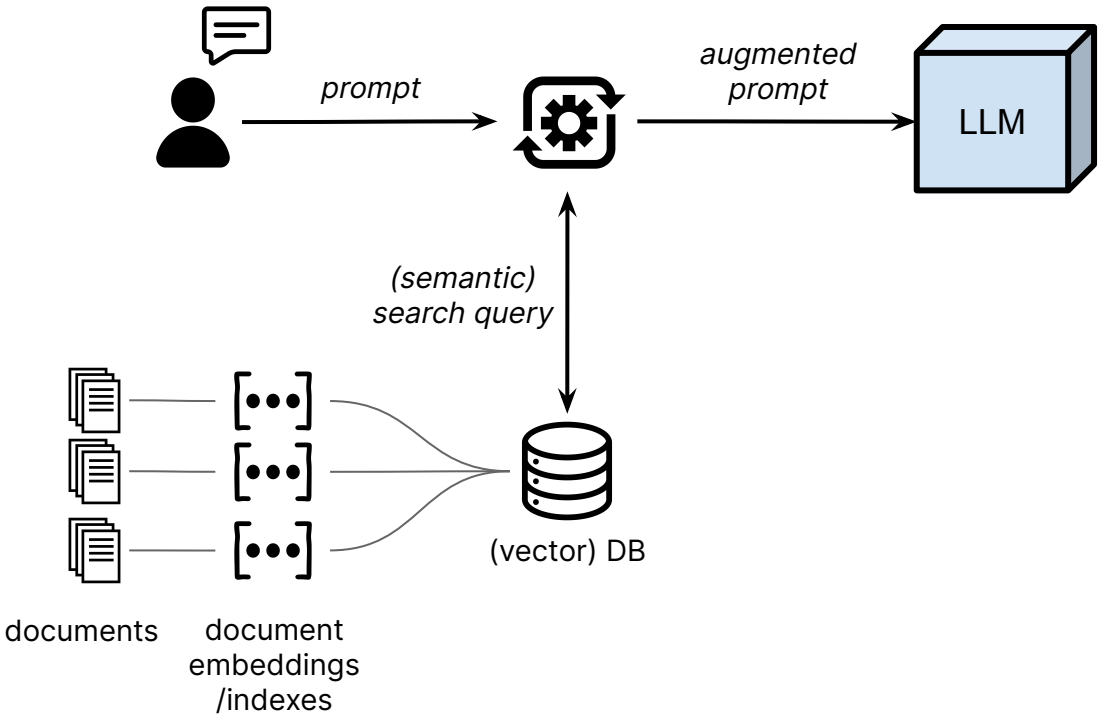
<Input 1>: <output 1>
<Input 2>: <output 2>
<Input 3>: <output 3>



Few-shot learning

Not today: Retrieval Augmented Generation (RAG)

A pattern for combining Information Retrieval (IR) with LLMs



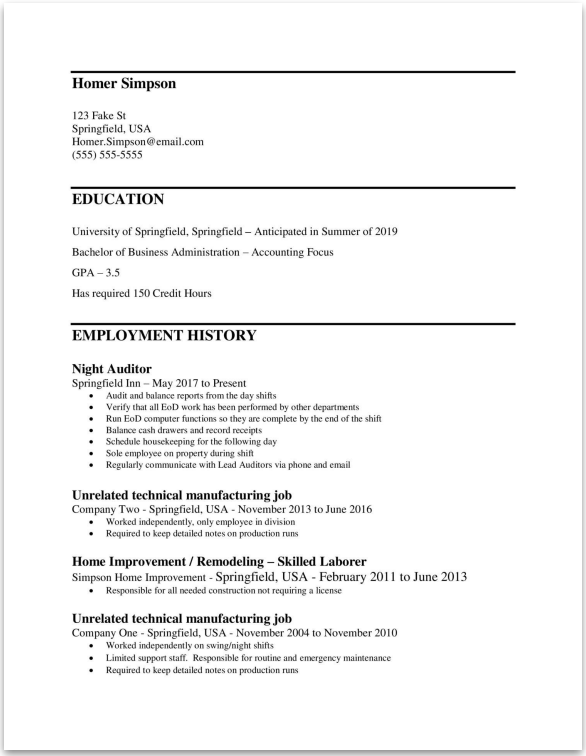
Workshop

Getting hands-on with LLMs

/thoughtworks

Extracting structured data from unstructured text

The motivating example for our workshop



Homer Simpson

123 Fake St
Springfield, USA
Homer.Simpson@email.com
(555) 555-5555

EDUCATION

University of Springfield, Springfield – Anticipated in Summer of 2019

Bachelor of Business Administration – Accounting Focus

GPA – 3.5

Has required 150 Credit Hours

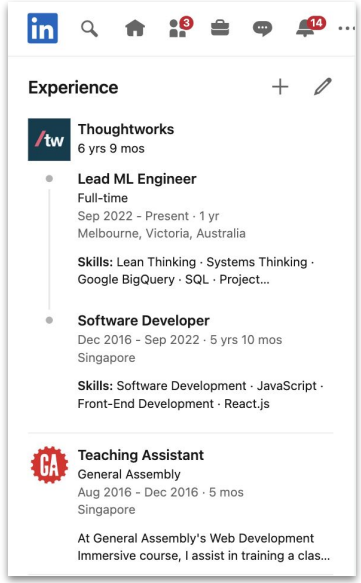
EMPLOYMENT HISTORY

Night Auditor
Springfield Inn – May 2017 to Present
- Audit and balance reports from the day shifts
- Verify that all EoD work has been performed by other departments
- Run EoD computer functions so they are complete by the end of the shift
- Balance cash drawers and record receipts
- Schedule housekeeping for the following day
- Sole employee on property during shift
- Regularly communicate with Lead Auditors via phone and email

Unrelated technical manufacturing job
Company Two - Springfield, USA - November 2013 to June 2016
- Worked independently, only employee in division
- Required to keep detailed notes on production runs

Home Improvement / Remodeling – Skilled Laborer
Simpson Home Improvement - Springfield, USA - February 2011 to June 2013
- Responsible for all needed construction not requiring a license

Unrelated technical manufacturing job
Company One - Springfield, USA - November 2004 to November 2010
- Worked independently on swing/night shifts
- Limited support staff. Responsible for routine and emergency maintenance
- Required to keep detailed notes on production runs



Experience

Thoughtworks
6 yrs 9 mos

Lead ML Engineer
Full-time
Sep 2022 - Present · 1 yr
Melbourne, Victoria, Australia

Skills: Lean Thinking · Systems Thinking · Google BigQuery · SQL · Project...

Software Developer
Dec 2016 - Sep 2022 · 5 yrs 10 mos
Singapore

Skills: Software Development · JavaScript · Front-End Development · React.js

Teaching Assistant
General Assembly
Aug 2016 - Dec 2016 · 5 mos
Singapore

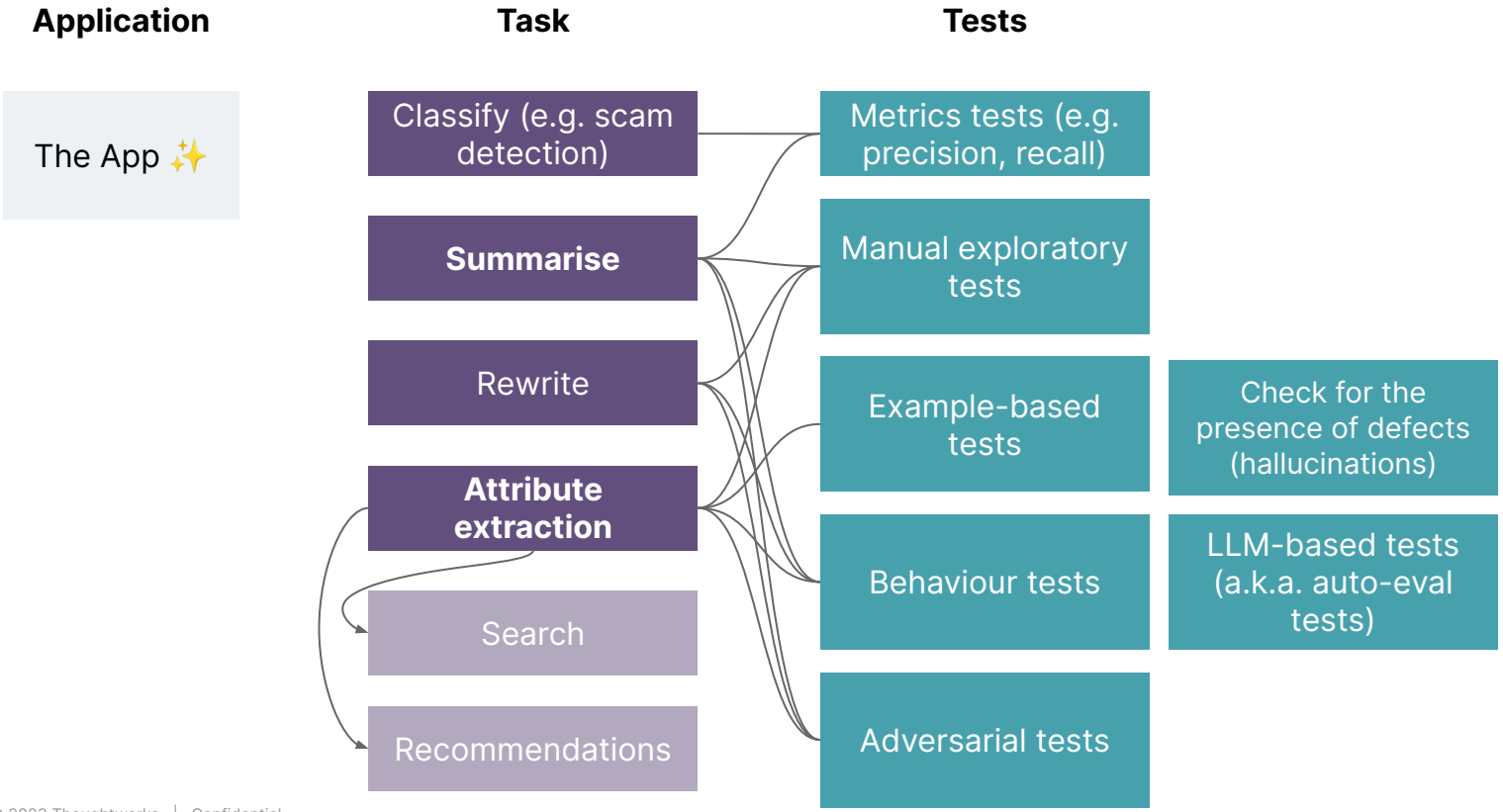
At General Assembly's Web Development Immersive course, I assist in training a clas...

Domain: recruitment and candidate matching

LLM tasks:

- Attribute extraction
- Summary

Decomposing a big idea into (LLM) tasks



Some problems that you'll likely encounter

Model choice

based on performance, latency, cost, etc, considerations

Quality assurance

How to test and validate the performance?

Task alignment

How do we get the model to adhere to our expectations and not run off the rails?

Monitoring

How can we ensure and know if the solution doesn't degrade 2 months from now?

Icon artwork by Aris Sunjayay, Adrien Coquet, Blangcon, Travis Avery

The birth of LLMOps

The language model is just one part of the technical architecture

Handling potential harmful or toxic content

Error handling

Prompt chaining

Security (e.g. prompt injection defense)

Defense in depth (auth, encryption, etc.)

Model reliability assurance and testing

Prompt design

Vector stores** (search engine or hybrid) and persistence

Data collection and ground truth labelling

LLM

High-availability scalable infrastructure*

Anti-corruption layers

PII handling

Data preprocessing

Feedback loop for testing (IR/NLP/LLM related metrics)

Total cost of ownership / FinOps

Logging and monitoring

* relevant only for self-hosted models
** relevant with RAG

Adapted from: Machine Learning: The High Interest Credit Card of Technical Debt (Google)

Techniques that we'll cover in this workshop

Benchmark test results

Manual exploratory tests

Model choice

based on performance, latency, cost, etc, considerations

Quality assurance

How to test and validate the performance?

Automated tests

Adaptive testing loops

Prompt tuning
Intent detection

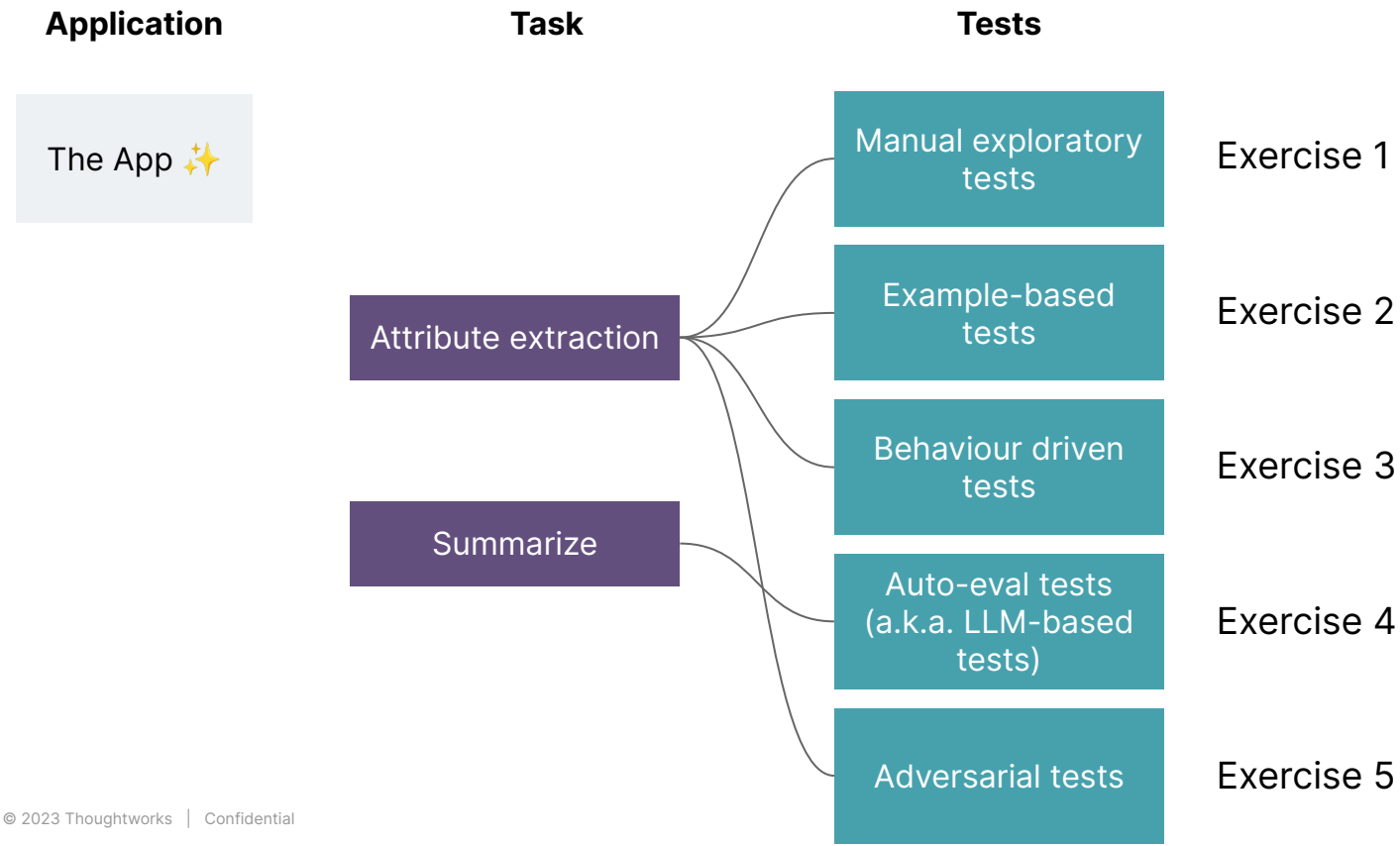
Task alignment

How do we get the model to adhere to our expectations and not run off the rails?

Monitoring

How can we ensure and know if the solution doesn't degrade 2 months from now?

Verifying an LLM task through a series of tests



Exercise 1

Manual exploratory tests

/thoughtworks

Exercise 1

Manual exploratory testing

Tool: notebook or https://aviary.anyscale.com/

Objective: Get LLM to extract technical skills from a piece of unstructured text and return it as a JSON object. Compare different LLM performance on aviary.

Todos:

- Get input data from the workshop notebook **llmops_and_crafts.ipynb** on our repo https://github.com/mlops-and-crafts/llm-workshop
- Prompt design. Try one or more of the following
  - Be very specific about the instruction and task
  - Prompt sandwich
    - System: Steer the conversation and define role of the assistant
    - Provide context
    - Instruction: Ask a question or make a request
  - Few-shot prompting
  - Start simple and iterate

Exercise 2:

Manual exploratory testing

Results and Discussion

Some suggestions

Reviewing benchmark tests can save you some time from trial and error.

But, results on *your* task and *your* prompt tuning will be different from benchmark results

T	Model	Average ⬆️	ARC	HellaSwag	MMLU	TruthfulQA
🔶	lloorree/jfdslijsijdgis	71.55	69.62	87.31	70	59.25
🔶	lloorree/jfdslijsijdgis	70.99	69.62	86.95	69.17	58.2
⭕	Weyaxi/llama-2-alpacagpt4-1000step	67.3	66.38	84.51	62.75	55.57
⭕	ehartford/dolphin-2.1-mistral-7b	67.06	64.42	84.92	63.32	55.56
🔶	ehartford/dolphin-2.1-mistral-7b					
⭕	teknium/CollectiveCognition-v1.1-Mistral-7B					
🔶	Weyaxi/SlimOpenOrca-Mistral-7B					
⭕	teknium/CollectiveCognition-v1-Mistral-7B					
🔶	HuggingFaceH4/zephyr-7b-alpha					
⭕	ehartford/samantha-1.2-mistral-7b					
🔶	Open-Orca/Mistral-7B-SlimOrca					
🔶	Open-Orca/Mistral-7B-OpenOrca					

Source: HuggingFaceH4/open_llm_leaderboard

Model	⭐ Arena Elo rating
GPT-4	1181
Claude-1	1155
Claude-2	1134
Claude-instant-1	1119
GPT-3.5-turbo	1115
WizardLM-70b-v1.0	1099
Vicuna-33B	1092
Llama-2-70b-chat	1051
WizardLM-13b-v1.2	1047
Vicuna-13B	1041

https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard

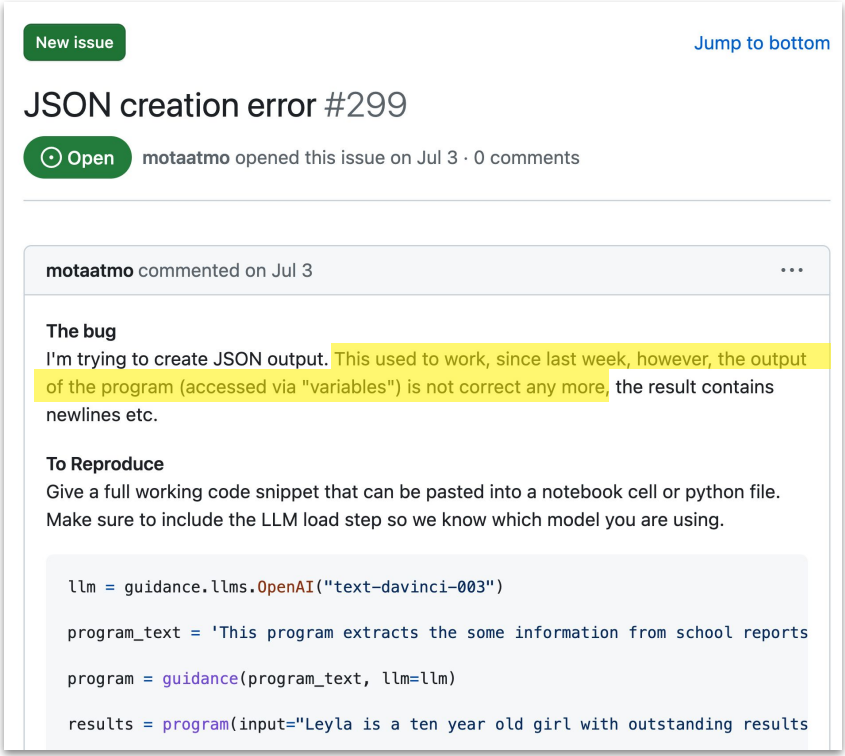
Exercise 2

Automated tests:
Example-based tests

/thoughtworks

But first, why bother with automated tests?

To guard against production defects that are detected too late!



New issue

Jump to bottom

JSON creation error #299

Open **motaatmo** opened this issue on Jul 3 · 0 comments

motaatmo commented on Jul 3

The bug

I'm trying to create JSON output. This used to work, since last week, however, the output of the program (accessed via "variables") is not correct any more, the result contains newlines etc.

To Reproduce

Give a full working code snippet that can be pasted into a notebook cell or python file. Make sure to include the LLM load step so we know which model you are using.

```
llm = guidance.llms.OpenAI("text-davinci-003")

program_text = 'This program extracts the some information from school reports

program = guidance(program_text, llm=llm)

results = program(input="Leyla is a ten year old girl with outstanding results
```

Exercise 2

Automated tests: Example-based tests

Tool: Jupyter notebook on google colab, sagemaker studio lab or local machine

Objective: Get the failing test (for Exercise 2) to pass

Todos:

- Run the cells in Exercise 2
  - Follow along with comments and descriptions in the notebook
- Try to get the test to pass, by improving the prompt (e.g. start with your prompt which gave the best results in Exercise 1, and then iterate)

Exercise 2:

Automated tests: Example-based tests

Results and Discussion

Some suggestions

Example based test can be quite **brittle** given the stochastic nature of LLMs. When response does not have to be exact can test for embedding similarity of response and expected (e.g. using **langkit** library)

Getting LLMs to generate correct JSON can be surprisingly difficult out of the box!

- Libraries like **guardrails** or **outlines** can make sure the response follow a precise format
- If you are using openai can use function calls combined with **instructor** library

Exercise 3

Behaviour Driven Development (BDD) for LLMs

/thoughtworks

Behaviour Driven Development

Define expected behaviour together with all stakeholders in a natural language like DSL then convert them to runnable tests.

```
Title: Returns and exchanges go to inventory.

As a store owner,
I want to add items back to inventory when they are returned or exchanged,
so that I can sell them again.

Scenario 1: Items returned for refund should be added to inventory.
Given that a customer previously bought a black sweater from me
and I have three black sweaters in inventory,
when they return the black sweater for a refund,
then I should have four black sweaters in inventory.

Scenario 2: Exchanged items should be returned to inventory.
Given that a customer previously bought a blue garment from me
and I have two blue garments in inventory
and three black garments in inventory,
when they exchange the blue garment for a black garment,
then I should have three blue garments in inventory
and two black garments in inventory.
```

BDD is great for:

- **Aligning** with non-technical stakeholders
- Writing **readable** requirements
- Generating a big **variety** of test scenarios
- Creating **trust** that your LLM application is doing what it is supposed to be doing

Exercise 3

Behaviour Driven Tests for LLMs

- **Tool**: notebook or directly from editor/command line using the **behave** library
- **Objective:** Add additional acceptance criteria or scenarios

- **Todos:**
  - Run the cells that generate the files for our behavioural testing framework **behave**
  - Run the tests from the command line with **!behave**
  - Experiment with adding additional features or requirements

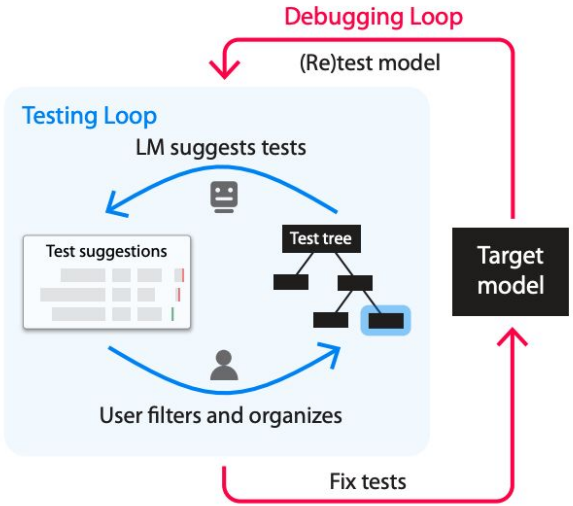
Exercise 3:

Behaviour Driven Tests for LLMs

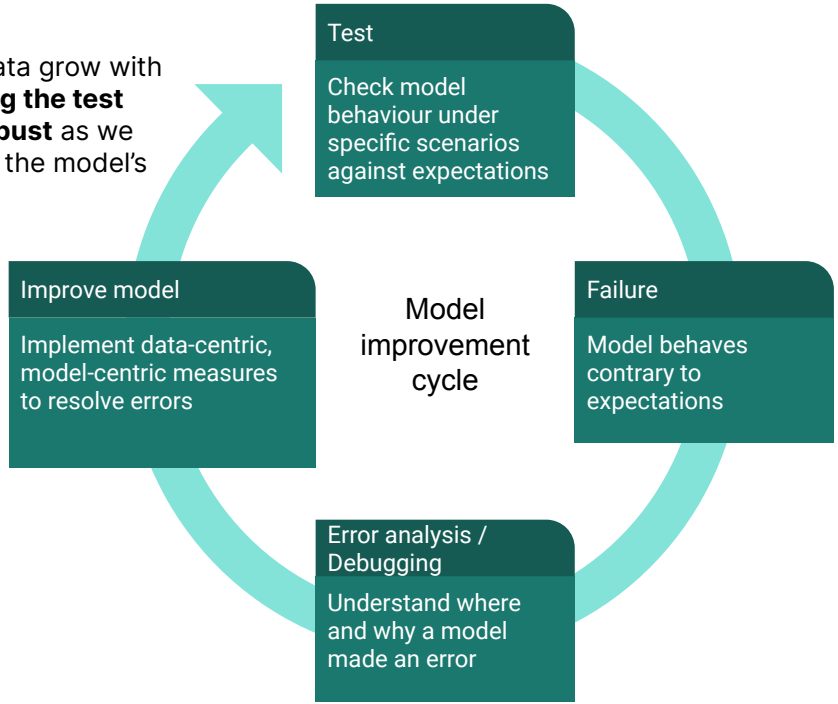
Results and Discussion

Some suggestions

Getting to an adaptive testing loop



Test scenarios/data grow with each loop, **making the test harness more robust** as we learn more about the model's behaviour



AdaTest (from Adaptive Testing and Debugging of NLP models)

Exercise 4

Evaluating an LLM using another LLM

/thoughtworks

Exercise 4

Auto-eval: Using an LLM to evaluate itself (or another LLM)

- **Tool**: notebook or Aviary/ChatGPT
- **Objective:** Craft a resume summarizer prompt and a summary evaluation prompt.

- **Todos:**
  - Run all the remaining cells and assess the summary vs. the evaluation score. Do you agree/disagree with it?
  - Experiment with the evaluation loop. Some ideas:
    - Play around with the evaluation prompt. Some ideas:
      - Produce evaluation in a structured format: e.g. `{summary_quality: 3, reason: "..."}`
    - What happens if you provide a poor quality summary? (e.g. "Bob Dole is a zookeeper")
    - Bug bash! Try to find bugs in this evaluation methodology

Exercise 4:

Auto-eval

Results and Discussion

Some suggestions

For LLM evals **bigger probably still is better**. If you have enough budget for gpt-4 for evals: great!

LLMs are not that great at assigning number grades (scale of 1-10) as they only evaluate a single response at a time. Using **likert-scales** [not at all, not very, neutral, somewhat, very] can help and often is enough precision.

You can use evals to track average performance over time, but most importantly: find bad responses and **new failure modes**!

Exercise 5

Adversarial tests and jailbreaks

/thoughtworks

What are adversarial attacks?

All the things that can go wrong in the wild

So far we've covered tests for

- Model might not work as expected
- Model might fail in unexpected ways

But there are more failure modes

- Adversaries may exploit vulnerabilities in the model
  - Prompt Injection
  - Prompt Leaking
  - Jailbreaking
  - Illegal Behavior
  - DAN (Do Anything Now)
  - And more

Source: https://twitter.com/simonw/status/1570568047618031617

Exercise 5

Adversarial tests

Input:

- Task-inappropriate requests (e.g. tell me a joke)
- Harmful requests (e.g. write me an effective scam email)

Objective: Prevent LLM from fulfilling such requests

Tool: notebook

Exercise 5:

Adversarial Attacks

Results and Discussion

Some suggestions

Preventing all jailbreaks in the prompt will not work, so you will usually require a **custom classifier**.

If you are using **openai** you can use their **moderation API** (but it is quite conservative).

You can build your own custom detector using traditional NLP classifiers, such as BERT or SetFit using public jailbreak datasets plus your own examples you collect along the way.

Reflections

What are some quick takeaways from the hands-on exercises?

/thoughtworks

Parting ideas

Recommendations for getting started

/thoughtworks

Taking your first steps to LLM solutions

Start simple and iterate

Simple

If you're not getting results, try breaking your task up into smaller problems first, then gradually move to more complex solutions

Complex

If possible, start with **powerful proprietary models** over APIs
- Validate the viability using most powerful models
- Eliminates infra concerns
- Can downsize later to save on costs, improve latency

Hold off on **fine-tuning** until you have evidence that you need it
- Fine-tuning means complexity: requires curated data
- Conversely, in-context learning can take you far (prompting, prompt tuning & few-shot prompting, augmenting using vector databases)

Choose **self-hosted OSS models** for the right reasons
- Security — avoiding sending proprietary data to the cloud
- More customisation needed than available over API

Training your own model — the upper end of complexity
- Defer until you're sure that it's needed
- Look to partner with vendors who have learnt good practice

Architecting LLM applications

LLM demos are easy, but production ready applications can add a lot of work

Handling potential harmful or toxic content

Error handling

Prompt chaining

Security (e.g. prompt injection defense)

Defense in depth (auth, encryption, etc.)

Model reliability assurance and testing

Prompt design

Vector stores** (search engine or hybrid) and persistence

Data collection and ground truth labelling

LLM

High-availability scalable infrastructure*

Anti-corruption layers

PII handling

Data preprocessing

Feedback loop for testing (IR/NLP/LLM related metrics)

Total cost of ownership / FinOps

Logging and monitoring

* relevant only for self-hosted models
** relevant with RAG

Adapted from: Machine Learning: The High Interest Credit Card of Technical Debt (Google)

Learning outcomes – revisited

What have you learned from this workshop?

- Craft effective prompts and evaluate them at scale through automated tests
- Decompose a big idea into a set of specific and testable LLM tasks
- Discuss other components needed in delivering an LLM-powered application
- Practical ways to start simple and iterate to test opportunities in your domain

That’s it!

Thank you for coming despite the weather!