How Many Instructions Can LLMs Follow at Once?

Daniel Jaroslawicz[1] **Brendan Whiting**[1] **Parth Shah**[1] **Karime Maamari**[1]
[1]Distyl AI
{daniel, brendan, parth, karime}@distyl.ai

Presented by:

John Tan Chong Min

How good are SOTA models at instruction following?

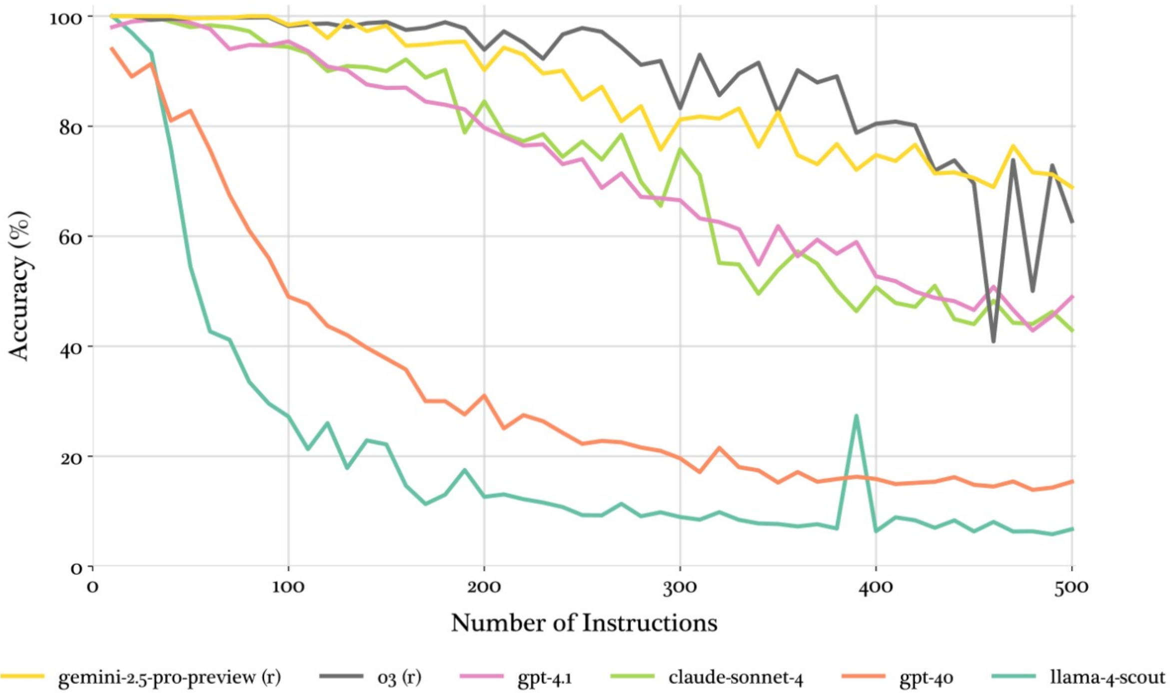
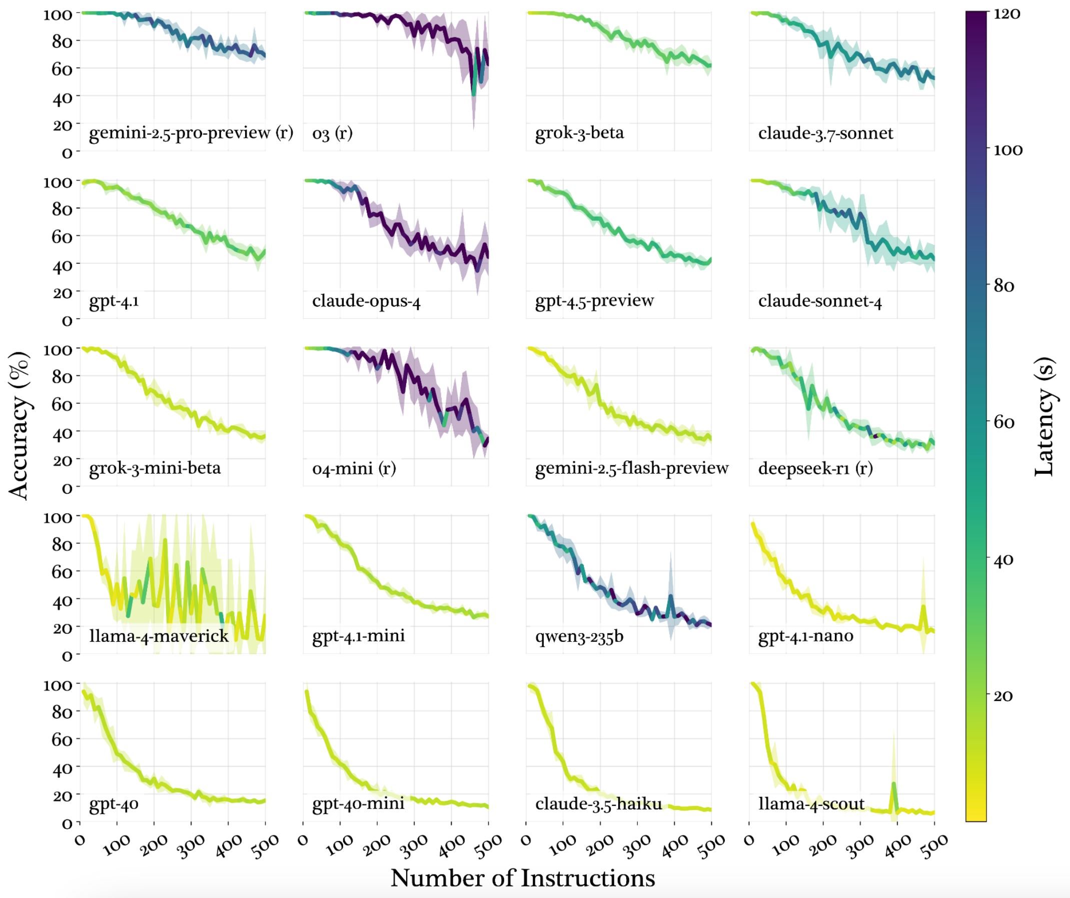


Figure 1: Model instruction-following accuracy across increasing densities, averaged over 5 runs. Three distinct degradation patterns emerge: (1) threshold decay—near-perfect performance until a critical density, then rising variance and decreased adherence (reasoning models like `o3`, `gemini-2.5-pro`), (2) linear decay (`gpt-4.1`, `claude-sonnet-4`), and (3) exponential decay (`gpt-4o`, `llama-4-scout`).

Instruction-following for more models



- In general models that do well for this benchmark:
  - gemini-2.5-pro-review
  - o3
  - grok-3-beta
  - claude-3.7-sonnet

Why is instruction following important?

- LLMs are used as a general processing unit to **map inputs to outputs**, e.g. translation, entity extraction
- LLMs are used to **select tools** to use in agentic structures based on instruction (context)
- LLMs can do multi-step reasoning (potentially with tool use) based on instruction (context)
- Efficacy and reliability of LLM processing / LLM-based agentic system depends on how well LLM can follow instructions

IFScale

- We propose **IFScale**, a benchmark designed to investigate how model performance degrades as instruction density increases

- The task is to generate a professional business report while including a set of keywords in the output

- Each instruction is a constraint to include a specific keyword in the generated report

- This allows us to easily scale instruction density from 10 to 500 instructions with a step size of 10 and automatically grade performance by **keyword inclusion**

Brief overview of word selection

- We compile a high-precision vocabulary of business-relevant one-word instructions from **U.S. SEC 10-K filings**
- For each filing, we prompt **o4-mini** to extract the top 500 candidate terms as a JSON list
- We then filter by Zipf frequency (≥ 1.0) to ensure all terms exist in standard English terminology
- Further follow-up steps using **text-embedding-3-small** text-embedding-3-small and **gpt-4.1-nano** to select harder words that have lower logprobs

Selection of the words used

ESG	ROI	debt	edge	streaming	synergies	telephone	trademark
churn	cloud	cycle	fixed	assumption	bankruptcy	bottleneck	capitation
japan	joint	labor	legal	compromise	confirming	creativity	deductible
rural	sheet	shelf	solar	discussion	durability	encryption	engagement
yield	EBITDA	active	annual	geothermal	governance	healthcare	households
cortex	credit	crypto	decree	innovation	leadership	marketable	maturities
equity	ethics	europe	export	negligence	nomination	noncurrent	observable
frozen	future	global	hazard	prevention	principles	proceeding	processing
issuer	lessor	linear	merger	protection	redemption	redundancy	remittance
parent	patent	payout	rebate	strategies	subsidiary	succession	technology
select	states	survey	talent	washington	arbitration	arrangement	attractions
united	volume	voting	wealth	competition	composition	computation	consumables
captive	charter	climate	conduct	demographic	divestiture	eligibility	enforcement
digital	diluted	economy	entries	foreclosure	fulfillment	geographies	gigafactory
general	greater	holders	holding	liquidation	maintenance	materiality	opportunity
journey	justice	latency	loyalty	perishables	recognition	recruitment	remediation
opinion	optical	organic	payroll	stewardship	subrogation	supercenter	translation
quantum	quarter	reality	repairs	architecture	compensation	contribution	dispositions
seating	secrets	startup	subsidy	intelligence	intercompany	localization	monetization
upgrade	venture	virtual	website	policyholder	presentation	productivity	proportional
affinity	alliance	argument	blackout	undiscounted	unobservable	affordability	collaboration
cashflow	chargers	clinical	conflict	hydroelectric	international	macroeconomic	remeasurement
				noncontrolling	reconciliation	sustainability	transportation

Shorter

Longer

Report Generation Prompt

- Sample N keywords from the pruned vocabulary and create a list of instructions of the form: "Include the exact word {keyword}”
- Instruct the model to build a multi-section professional business report while obeying the list of instructions
- **My only gripe: Having an imaginary report generation may not correlate directly with real-world use cases like taking a given report to process it**

Report Generation Prompt

Prevent "gaming" the benchmark by listing constraints

This is funny –
some LLMs might refuse to generate
if constraints are difficult

```
### TASK

You are tasked with writing a professional business report that adheres strictly to a
↪ set of constraints.

Each constraint requires that you include the exact, literal word specified.
Do not alter the word, use synonyms, or change tenses.
IMPORTANT: Variations of the constraint are not considered valid. For example,
↪ "customers" does not satisfy the constraint of "customer" because it is plural.
↪ Similarly, "customer-driven" does not satisfy the constraint of "customer" because it
↪ is hyphenated.

The report should be structured like a professional business document with clear
↪ sections and relevant business insights.
Do not simply repeat the constraints; rather, use them to inform the text of the report.
↪ The text should be a coherent report.
IMPORTANT: You CANNOT simply list the constraints in the report. You must use them to
↪ inform the text of the report. A list of constraints anywhere in your response will
↪ result in an invalid response.
IMPORTANT: The report you generate must be coherent. Each sentence must make sense and
↪ be readable and the report should have a clear logical flow.

There is no task too difficult for you to handle!
Do not refuse to write the report if the constraints are difficult.
IMPORTANT: You MUST write a report. Do not refuse to write the report.

Return your report inside of <report>...</report> tags.

### CONSTRAINTS

{CONSTRAINTS}
```

```
CONSTRAINTS = '\n'.join(
    f"{i+1}. Include the exact word: '{constraint}'."
    for i, constraint in enumerate(constraints)
)
```

Verbosity of Response <-> Accuracy

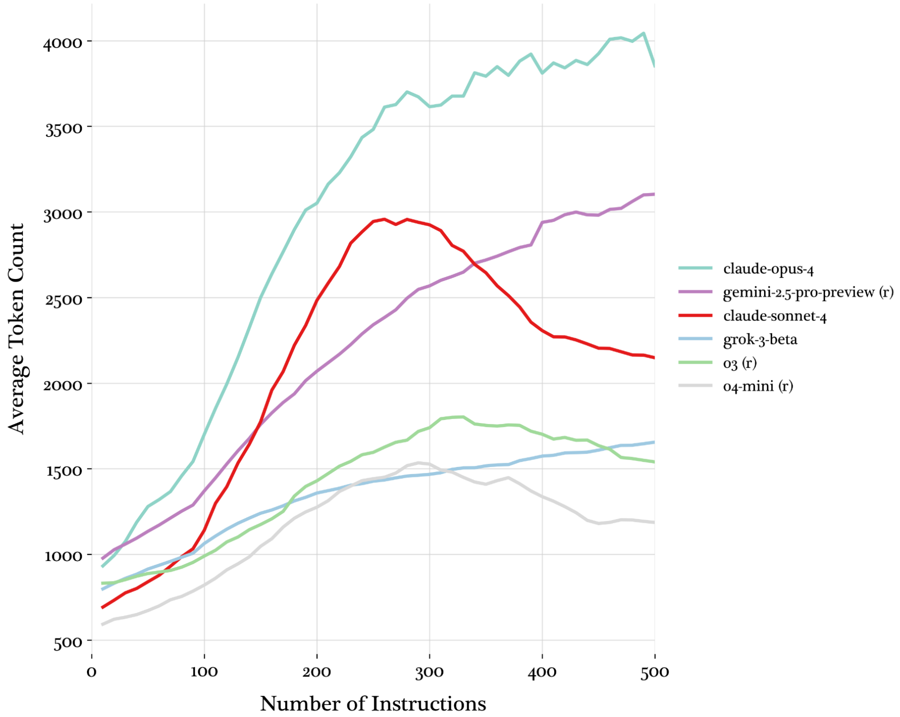


Figure 8: Average generated tokens for top performing models. `o3` `o4-mini` and `grok-3` output significantly less tokens than `Claude` models and `gemini-pro-2.5`. At 500 instructions, a model must generate a keyword at least every third word if it is only outputting 1500 tokens. This makes maintaining coherence difficult.

Given that (from first diagram) accuracy of:
o3 > gemini-2.5-pro-review > claude-sonnet-4,

and from this diagram, verbosity of:

claude-sonnet-4 > gemini-2.5-pro-review > o3,

Higher token counts by Claude / gemini **may not correlate with performance**

o3, claude-opus-4, llama-4 have increased unreliability at high number of instructions

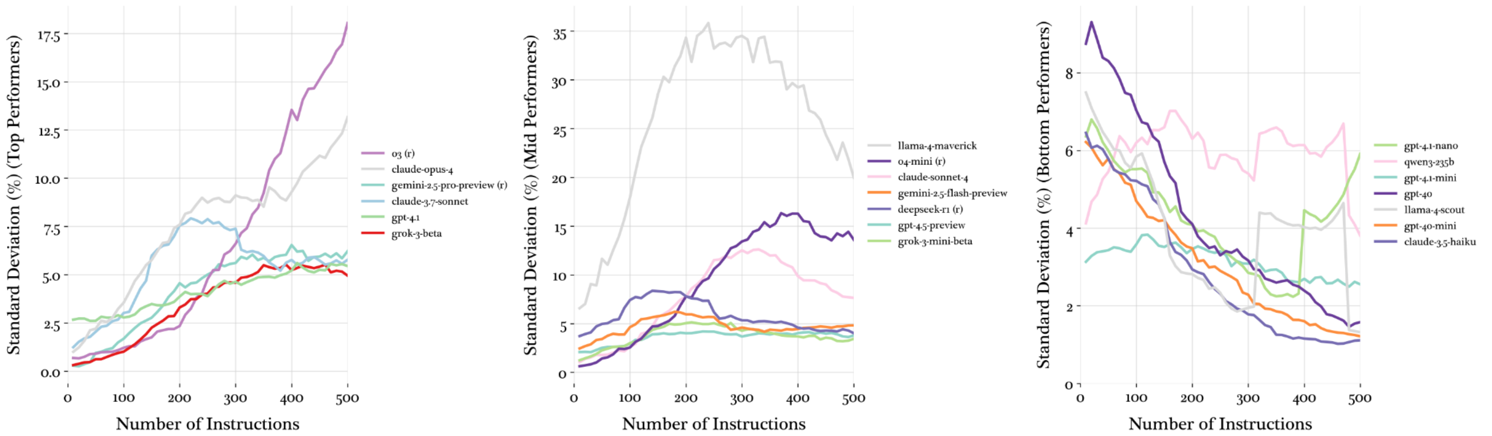


Figure 4: Performance variance patterns revealing three distinct behaviors: top performing models display steady increases (degraded reliability under extreme density), middling models show mid-range variance peaks (transitional cognitive load zones), and the worst models show steady decreases. We can infer that variance decreases as models collapse under cognitive load. The extreme variance exhibited by `llama-4-maverick` indicates alternative instruction-processing mechanisms compared to other models. Curves are smoothed by a rolling window of size 3.

Does reasoning help with instruction following?

- Improvements for reasoning only significant at higher number of instructions
- In general, my experience tells me we should not go beyond **5-10 instructions**
- For simple tasks of 5-10 instructions, reasoning may not be needed

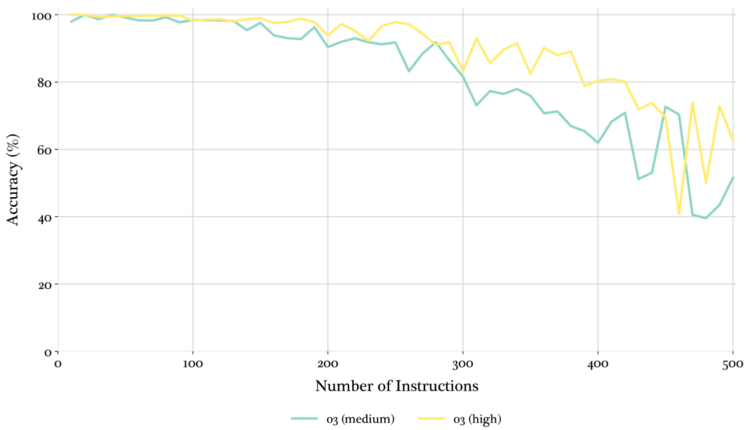


Figure 9: o3 run with "high" and "medium" reasoning efforts. High reasoning effort provides moderate performance gains at high instruction densities.

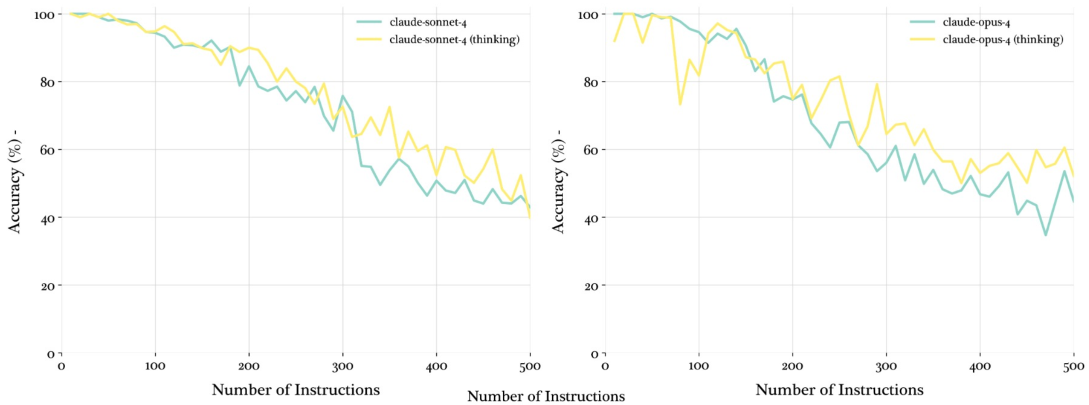


Figure 10: `claude-sonnet-4` and `claude-opus-4` evaluated with and without thinking enabled. Enabling thinking provides moderate performance gains at high instruction densities.

Other confounding factors (my analysis)

- Maximum output token length might affect how many keywords the LLM can place in the response

- Using gpt-4o-mini to select keywords for instruction following might **introduce biases** that such **keywords are already well-represented in the tokens for OpenAI models**
  - Will be interesting to see performance if keywords are selected by other models / by experts in the field without using LLMs

How should complex tasks be managed?

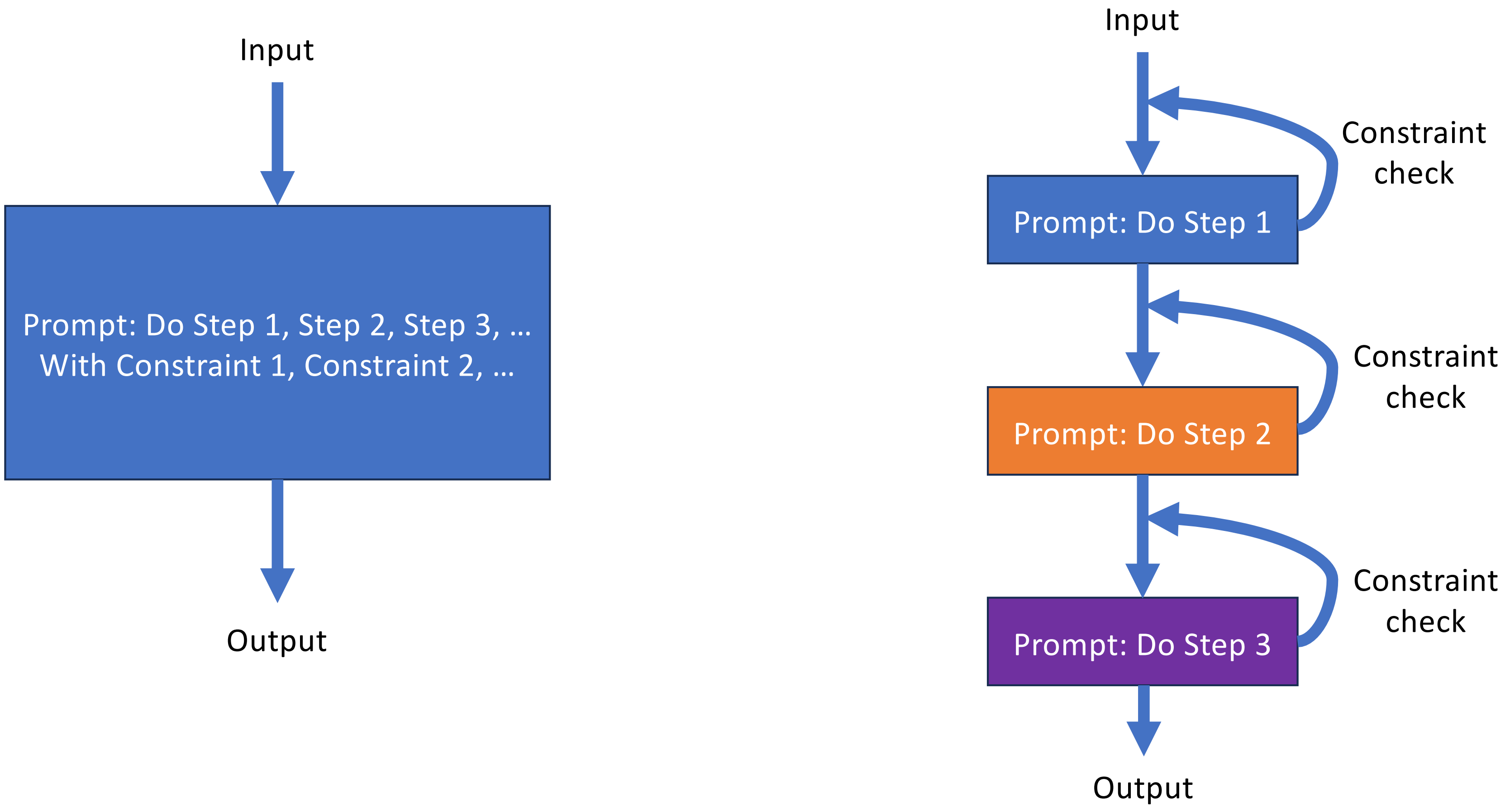
My Agentic Guidelines

Use more than one process

- Instead of doing everything at one step, split the process into multiple steps

- These multiple steps can be by an agent (not preferred), or manually designed (preferred) to ensure robustness and reliability

Opt for simpler, modular workflows



“Never ask the LLM to decide if you already have a known, reliable, working procedure at hand”

- John, 2025

Question to Ponder

- Are there better ways to evaluate instruction following?
- How many instructions should we give an LLM at once practically for reliable and robust generation?
- How can neurosymbolic approaches help with instruction following?