A practical guide to using AI to enhance your quality of life

(and research productivity)

Courtney Balentine, MD, MPH, FACS

Director of the Wisconsin Surgical Outcomes Research Program (WiSOR)

Director of Data Science for the Department of Surgery

Department of Surgery
UNIVERSITY OF WISCONSIN
SCHOOL OF MEDICINE AND PUBLIC HEALTH

Endocrine Surgery

Disclosures

- I have no financial stake in any of the companies producing large language models
- I am not paid by any of the companies that make large language models

Motivation for this talk

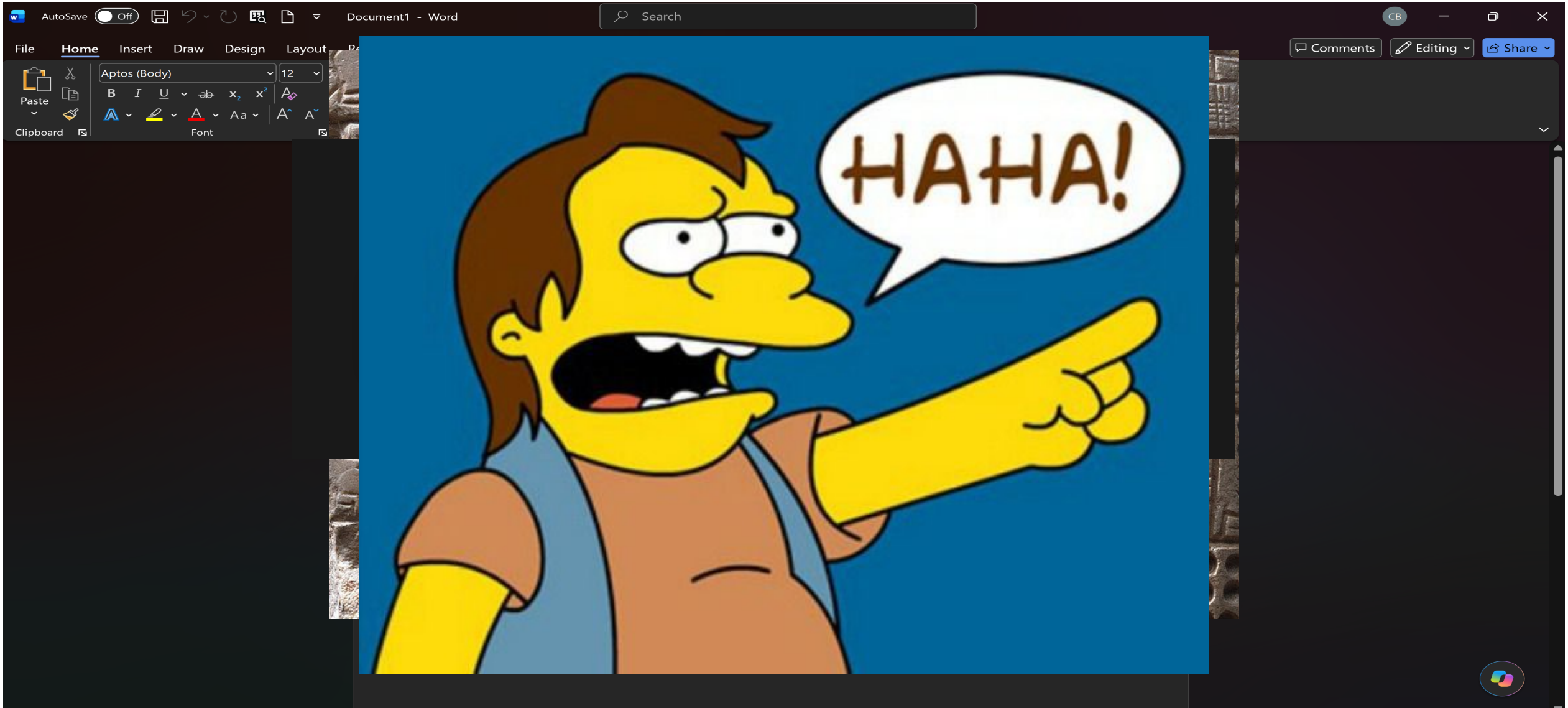
- One of the few constants in your career: work & demands on your time will increase
  - “Could you just help a little with …”
  - “It won’t be too much work …”
- Your employer will never say “Gee, we think you are working too hard and should take more time to relax”
- AI presents a rare opportunity to deflect that trajectory



Goals for this presentation: practical knowledge to enhance quality of life

- Learn how to communicate effectively with LLMs to get what you want
- Learn to work with skills and connectors in Claude
- Example of end-to-end workflows using Claude skills
- Understand potential risks of using LLMs and how to mitigate.

Key mental adjustment for successfully working with AI



Prompting: Telling the LLM what to do

Prompt engineering

- This is how you communicate your intent to the AI

Anatomy of Effective Prompts

1. **Role**: Who is the LLM?
    - "You are an expert experimental social scientist…"
2. **Task**: What do you want?
    - "Generate 3 news articles about climate policy…"
3. **Context**: What background information is relevant?
    - "These will be used in a survey experiment testing…"
4. **Constraints**: What rules must be followed?
    - "Each must be exactly 200 words, 8th grade reading level…"
5. **Format**: How should output be structured?
    - "Return as JSON with fields: article_id, text, frame_type"

How to think about prompting

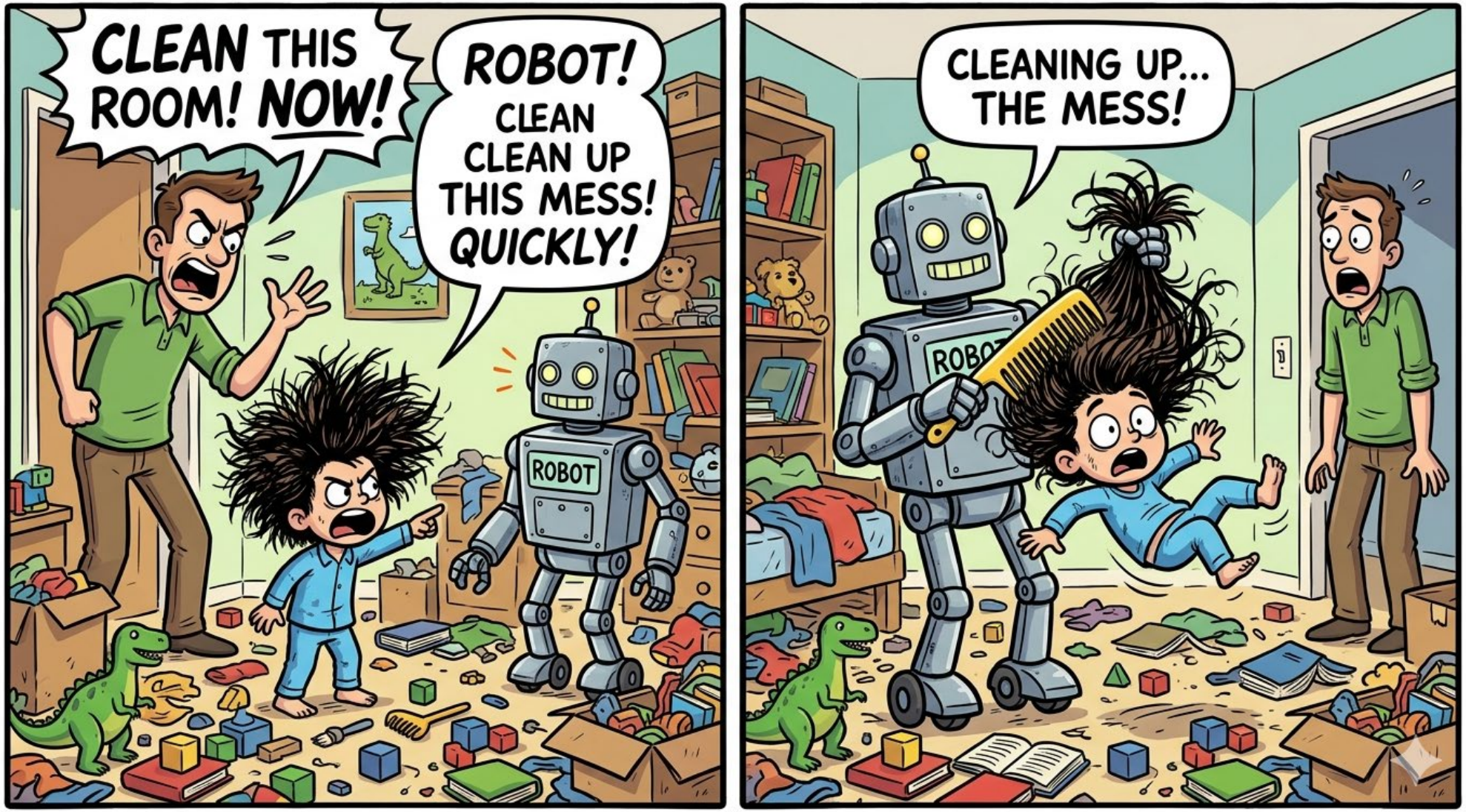
Imagine conversing with a brilliant but totally inexperienced new employee who is afraid to ask questions



Other tips for optimal prompting

- Don’t bury the lead: get better results by telling LLM what you want up front, then can provide additional information
- Formatting
  - New paragraph for new part of prompt
  - Caps for NEVER, ALWAYS, etc. can help LLM differentiate what is what

Common pitfall of prompting: ambiguity

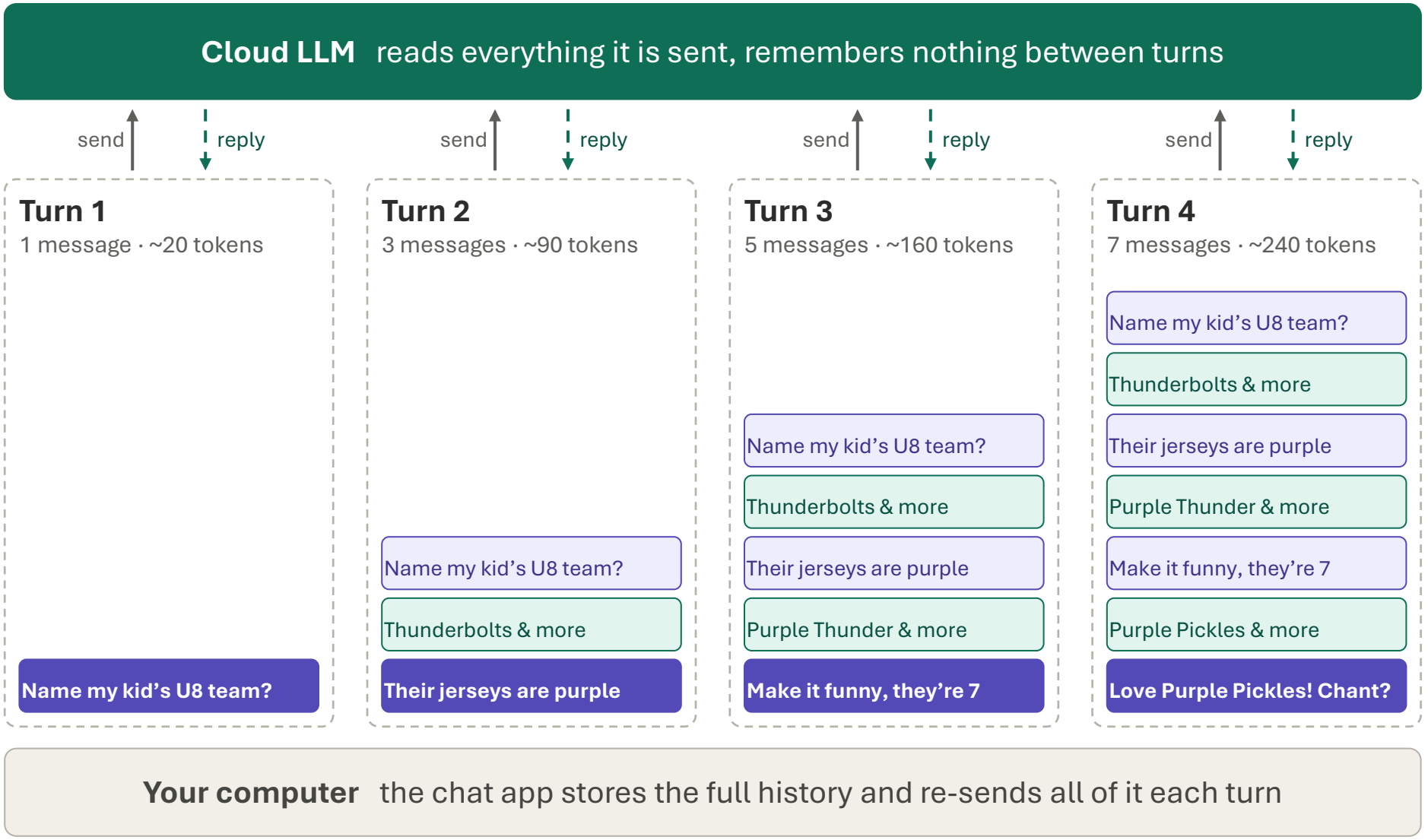


The changing meaning of context

- Used to mean mostly descriptive text in your prompt
- Documents (grants, papers, literature review)
- Images: figures for grant or paper, explanatory diagrams
- Instructions for the project
- Meeting notes & communication with teammates

Why context matters: every turn, the whole conversation is sent to the model again

Example: asking an LLM to help name a kid's soccer team



~510

tokens the model reads to write 4 short replies

Sent per turn:
20 → 90 → 160 → 240

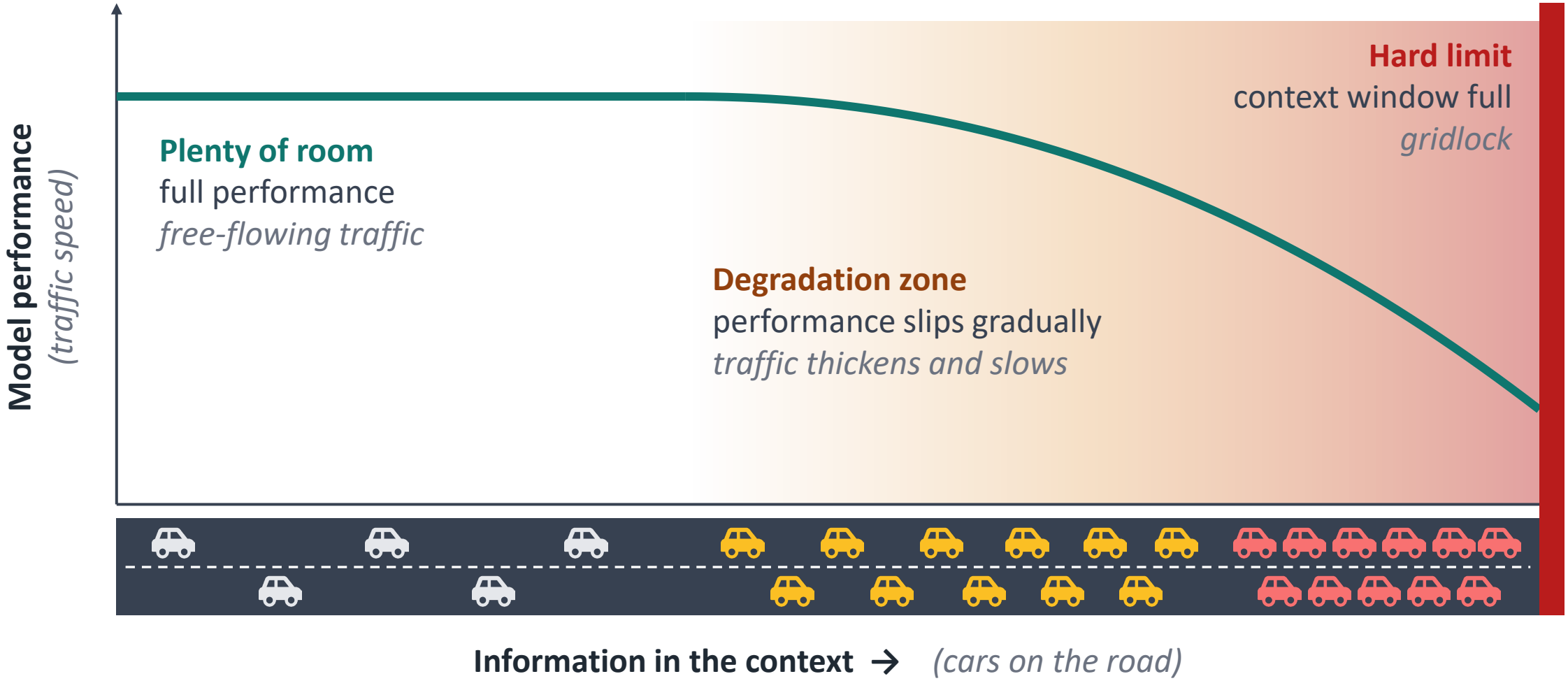
Key

- Your earlier messages
- Your new message
- LLM replies

Token counts are illustrative

AI context has two limits, not one

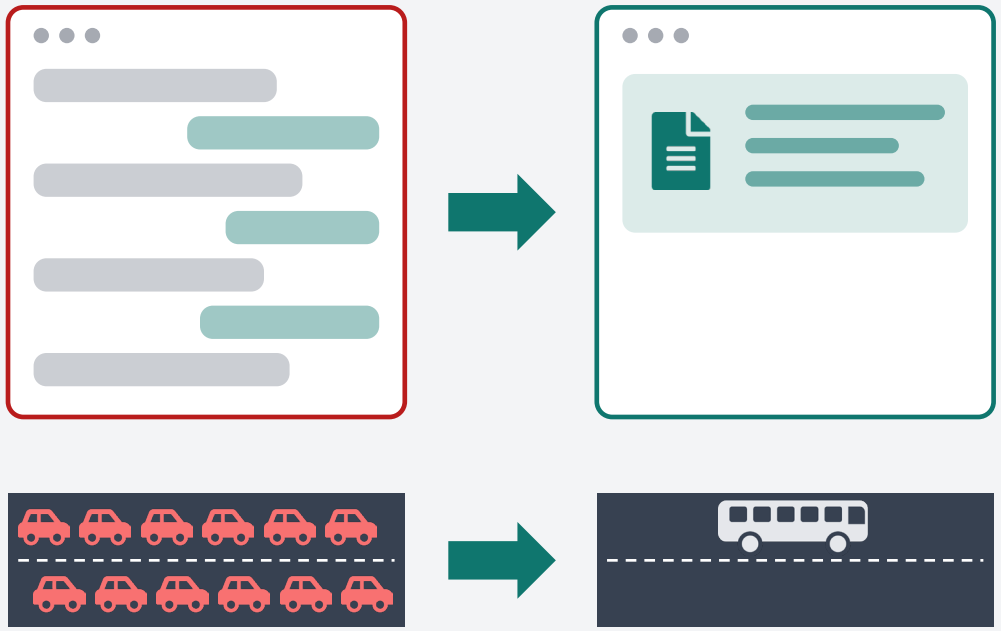
Like a highway, a model slows down well before it is completely full



Two ways out of the degradation zone

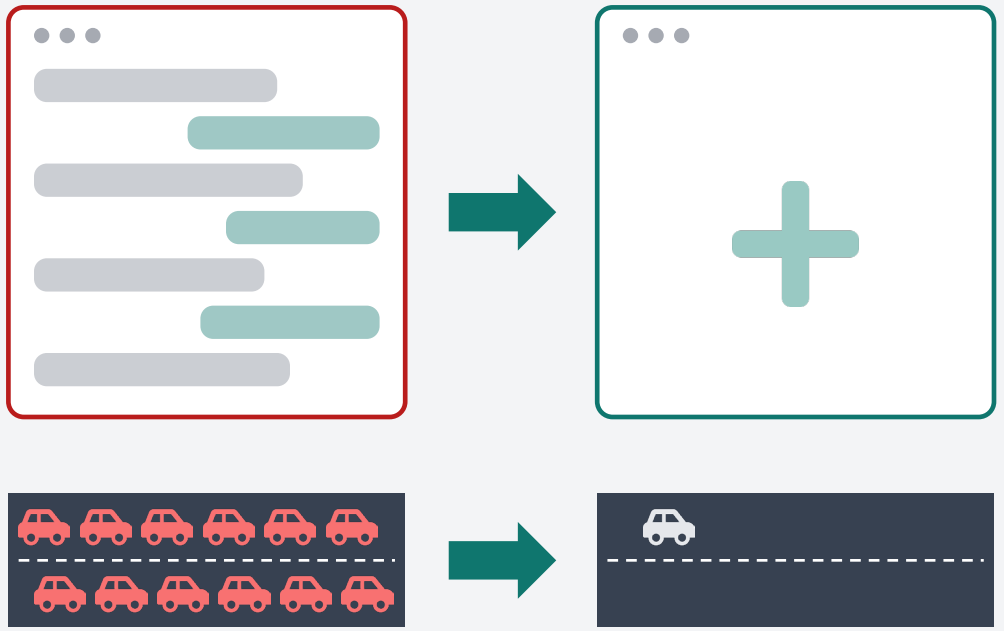
Condense the conversation

keep the essentials



Start a new chat

begin with a clean slate



Introduction to Claude: projects, skills, and connectors

Claude: an active partner in your work

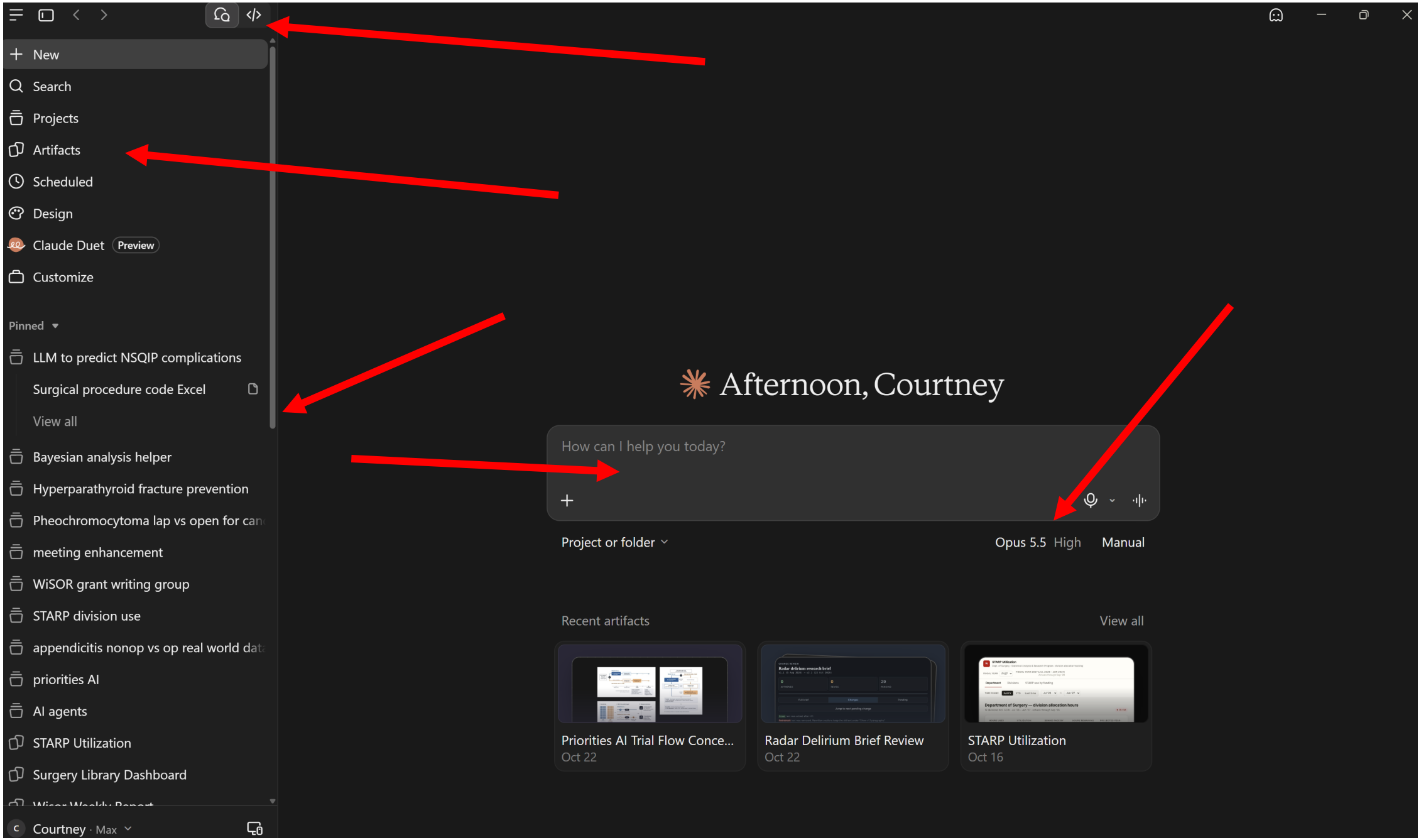
- Unlike chatbots and most (current) university LLMs, Claude can directly take action on your behalf
- Claude can
  - Access folders & files
  - Create (and destroy) files
  - Fill out forms & manipulate web pages
  - Issue instructions to other software
- Positives: makes Claude very powerful & helpful
- But …


WITH GREAT POWER COMES GREAT RESPONSIBILITY..

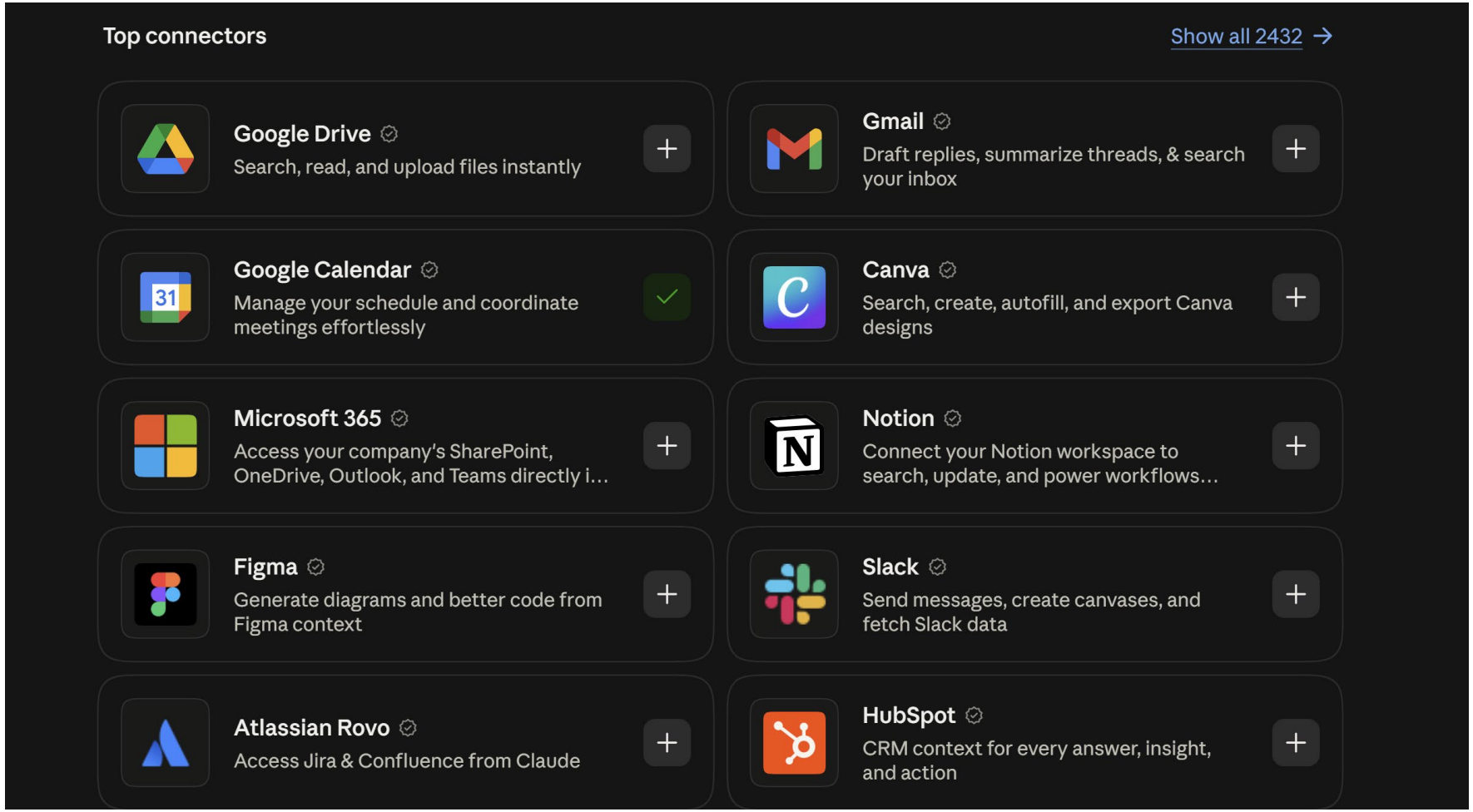
Only give LLM access to the folders and files that are absolutely necessary for the project

Meta Security Researcher's AI Agent Accidentally Deleted Her Emails

Meta's Summer Yue says she ran OpenClaw on her inbox, but its size 'triggered compaction [and] lost my original instruction' to get her permission before deleting.

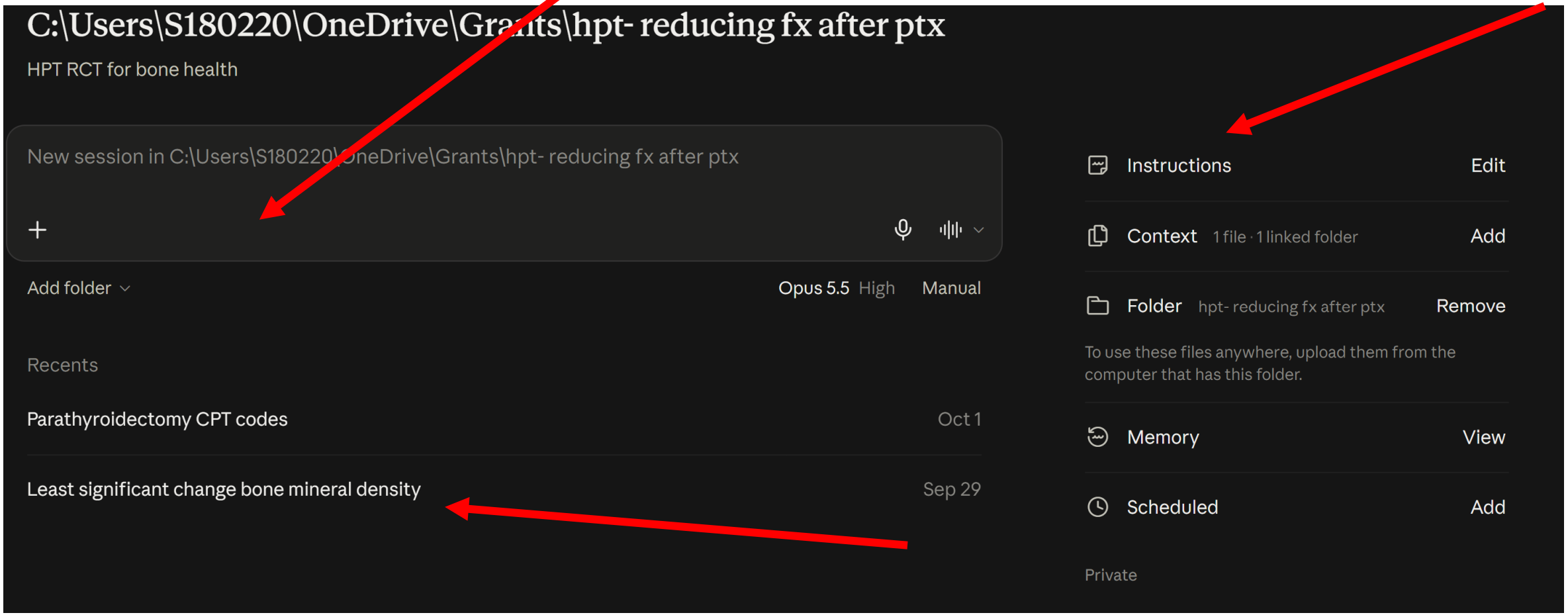

New
Search
Projects
Artifacts
Scheduled
Design
Claude Duet
Preview
Customize
Pinned
LLM to predict NSQIP complications
Surgical procedure code Excel
View all
Bayesian analysis helper
Hyperparathyroid fracture prevention
Pheochromocytoma lap vs open for can
meeting enhancement
WiSOR grant writing group
STARP division use
appendicitis nonop vs op real world dat
priorities AI
AI agents
STARP Utilization
Surgery Library Dashboard
Courtney · Max
Afternoon, Courtney
How can I help you today?
Project or folder
Opus 5.5 High
Manual
Recent artifacts
View all
Priorities AI Trial Flow Conce...
Oct 22
Radar Delirium Brief Review
Oct 22
STARP Utilization
Oct 16

Connectors: enabling Claude to work directly with other software



Projects in Claude

- Gives Claude access to **specific** folder(s) and files (i.e. context)
- Instructions & context that are applied to all future prompts within each project
  - Saves your having to add the context with each prompt
- Can monitor and make progress on your mobile devices
- Probably coming soon: central agent (chief of staff) manages all agents & tasks within a project & spins off agents/subagents to handle different tasks
  - Already active in Claude Code


C:\Users\S180220\OneDrive\Grants\hpt- reducing fx after ptx
HPT RCT for bone health
New session in C:\Users\S180220\OneDrive\Grants\hpt- reducing fx after ptx
Add folder
Opus 5.5 High
Manual
Recents
Parathyroidectomy CPT codes
Oct 1
Least significant change bone mineral density
Sep 29
Instructions
Edit
Context
1 file · 1 linked folder
Add
Folder
hpt- reducing fx after ptx
Remove
To use these files anywhere, upload them from the computer that has this folder.
Memory
View
Scheduled
Add
Private

Skills: your coworkers for projects and tasks

- A skill is a saved set of instructions, templates, and examples Claude loads when the task calls for it (i.e. skill for writing IRBs)
- Write the workflow once instead of re-explaining it in every conversation
- Good candidates: anything you do the same way every time
  - Drafting specific aims or a reviewer response
  - Running a PubMed review with a fixed verification protocol
  - Formatting an abstract to a journal's rules
- Claude calls the skill on its own when your request matches, or you call it by name with a slash command (/irb-writer)
- The skill carries your standards, so the output comes back in your voice and structure, not generic AI prose

Elements of a Claude skill bundle

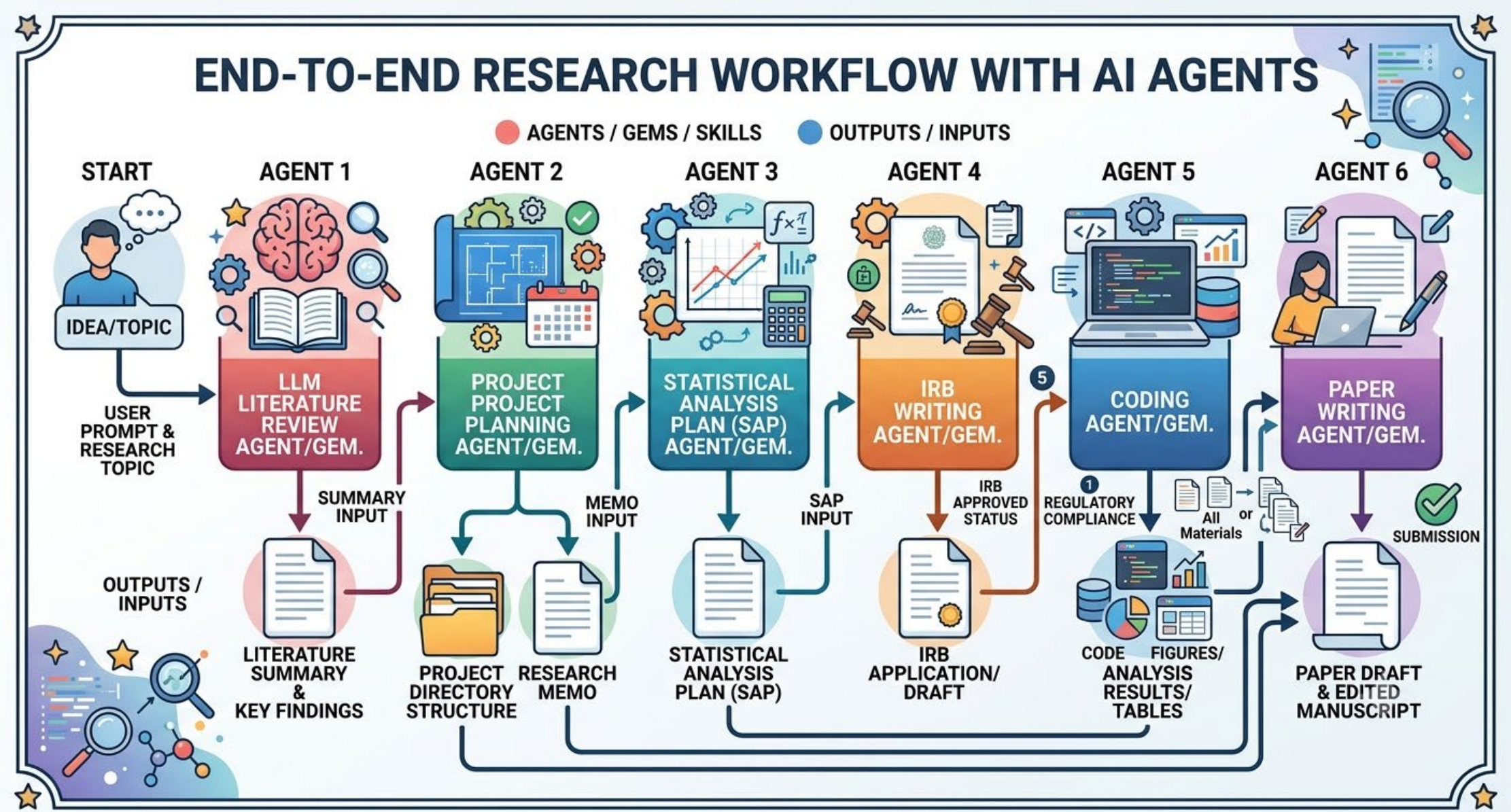
- instructions on what to do (the skill file)
- supporting instructions & files
  - Detailed instructions for tasks that are sometimes but not always used when the skill is called
  - Examples of outputs
  - Scripts/code
- description of when to call skill
- Optimizes use of your context window: only necessary instructions get read and used
  - Uses markdown format because less expensive for Claude

Ways to create a skill

- Ask Claude to help you
  - Describe what you want and ask Claude to create a draft of the skill
- Do a task in Claude and then ask it to review what you've done and create a skill out of that
- Record a skill
  - Do stuff and have Claude "watch" and learn
- Search for a comparable skill and modify
- All ways: Test and revise (and create backup copies)

How skills & connectors improve efficiency and quality

End-to-end workflows with LLMs



How end-to-end workflows help

- Efficiency + cognitive offloading: time consuming steps get automated so you spend your time thinking instead of doing the tedious stuff
- Minimize wasted time
  - Get more input on project design in the early phase so you don’t have to go back and fix things later
  - Avoid internal inconsistencies by having a project partner who can review all your study documents
  - Makes it easier to talk to your content & methods experts (and to get feedback from them)
- Planning: connections with project/task management software facilitates planning and tracking

Let’s see how partnering with AI works in practice



pubmed-lit-review

by you · updated Aug 3

Share

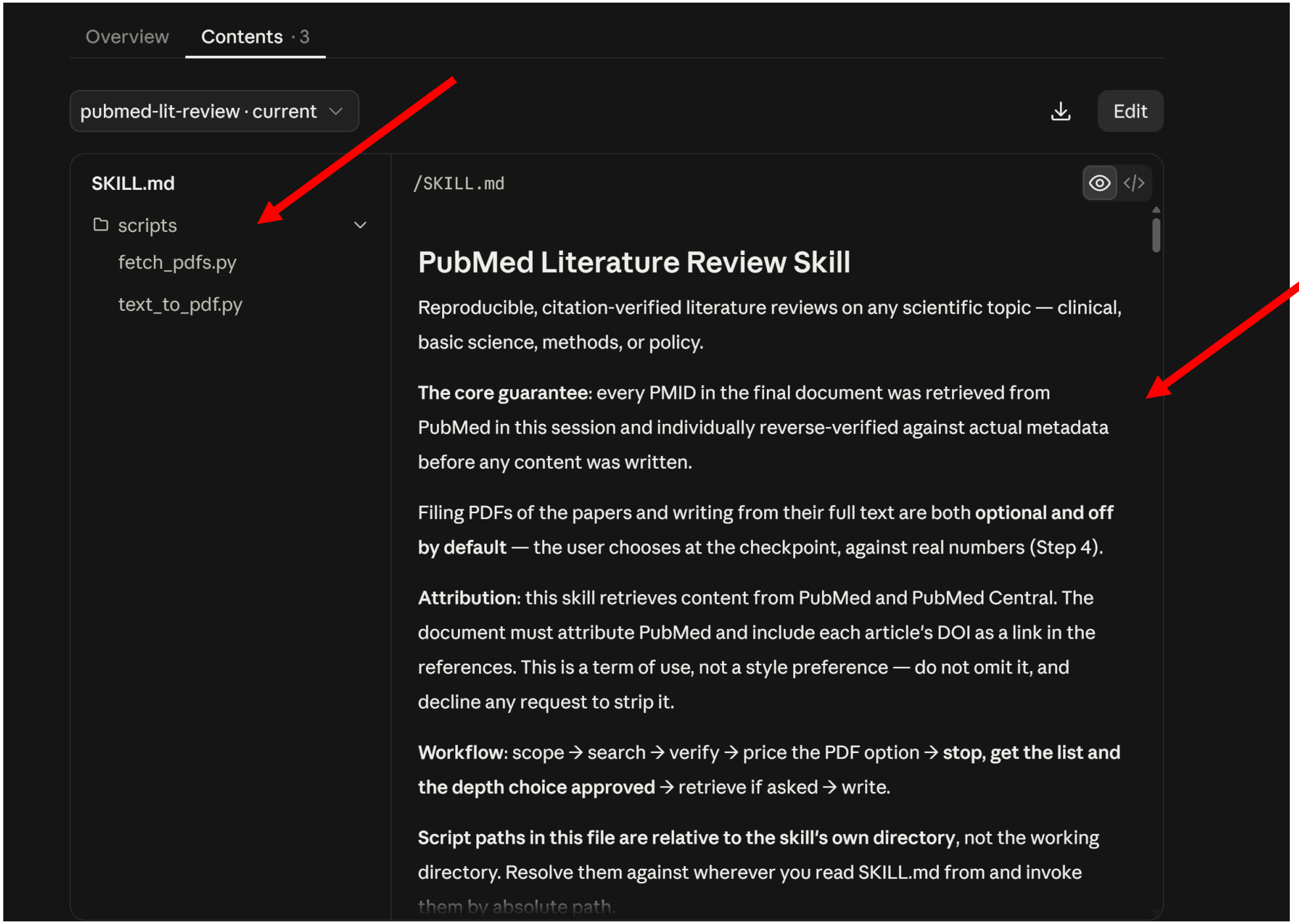
Overview | Contents · 3

Description

Conduct a rigorous, hallucination-resistant PubMed literature review on any scientific topic and produce a structured Word document (.docx) with executive summary, summary table, per-study deep dives, contradictions, references, and a PMID verification audit trail. Optionally downloads the open-access PDFs of the included papers into a folder the user chooses, and can write the deep dives from full text rather than abstracts. Use this skill whenever the user asks for a literature review, lit review, systematic search, review of the literature, or to find and summarize papers on any scientific topic. Always trigger on phrases like "do a literature review", "search the literature", "find papers on", "review studies on", "summarize what's known about", "what does the literature say about", "get me the PDFs for these papers", "download the papers you found". Never skip the verification protocol — hallucinated PMIDs are a serious research error.

Created by

You


Overview
Contents · 3
pubmed-lit-review · current
Edit
SKILL.md
scripts
fetch_pdfs.py
text_to_pdf.py
/SKILL.md
PubMed Literature Review Skill
Reproducible, citation-verified literature reviews on any scientific topic — clinical, basic science, methods, or policy.
The core guarantee: every PMID in the final document was retrieved from PubMed in this session and individually reverse-verified against actual metadata before any content was written.
Filing PDFs of the papers and writing from their full text are both optional and off by default — the user chooses at the checkpoint, against real numbers (Step 4).
Attribution: this skill retrieves content from PubMed and PubMed Central. The document must attribute PubMed and include each article's DOI as a link in the references. This is a term of use, not a style preference — do not omit it, and decline any request to strip it.
Workflow: scope → search → verify → price the PDF option → stop, get the list and the depth choice approved → retrieve if asked → write.
Script paths in this file are relative to the skill's own directory, not the working directory. Resolve them against wherever you read SKILL.md from and invoke them by absolute path.

pubmed-lit-review · current

Edit

SKILL.md

scripts

fetch_pdfs.py

text_to_pdf.py

/SKILL.md

Step 1 — Scope the Review

Clarify before searching: the **topic**, any **population/setting** restriction, the **date range** (default: last 10 years), the **study types**, and roughly **how many papers** (10–15 focused, 20–35 broad, no hard cap).

Ask only if an answer would substantially change the search.

Also get the user's **email** if you don't have it — NCBI and Unpaywall both require a contact address, the scripts hard-exit without one, and `$LITREVIEW_EMAIL` may already hold it. Cheaper to ask now than to stall at Step 3b.

Step 2 — Search PubMed (NO memory PMIDs ever)

CRITICAL RULE: never write down a PMID you recall from training. Every PMID must come from a live tool call in this session. LLMs cannot reliably recall PubMed IDs — a remembered PMID is a hallucination risk.

Run 3–6 `search_articles` queries covering the topic from different angles (different MeSH-style terms, method keywords, subpopulations). `find_related_articles` on a strong hit expands coverage, and `lookup_article_by_citation` recovers a known paper that search misses;

pubmed-lit-review · current

Edit

SKILL.md

scripts

fetch_pdfs.py

text_to_pdf.py

/SKILL.md

Step 3 — Verify Every PMID (before writing anything)

1. Call `get_article_metadata`, batching 3–5 PMIDs per call.
2. Read the result. Small results come back inline; large ones write a file. Metadata records can exceed 300,000 characters — conference proceedings volumes are the usual culprit — so query a file with `jq` (e.g. `jq '.articles[] | {pmid, title, journal, publication_date}' <file>`). **Do not use `Read` with a line limit:** these files are often a single line, so a line limit does not chunk them and you get everything or nothing.
3. Extract **Title**, **First Author**, **Journal**, **Year**.
4. Add a row to the verification log.
5. Screen on the retrieved title and abstract — not on what you expect the paper to say.

Two traps, both seen in real runs:

- **Conference abstract books rank high** in relevance-sorted searches and read like studies until you check the retrieved title. Screen on what came back, never on search rank.

pubmed-lit-review · current

Edit

SKILL.md

scripts

fetch_pdfs.py

text_to_pdf.py

/SKILL.md

Verification log format:

PMID	Confirmed Title	First Author	Journal	Year	Decision	Exclusion Reason
12345678	[title from metadata]	Smith	NEJM	2022	Include	—
23456789	[title from metadata]	Jones	JAMA	2019	Exclude	Not primary research

Any PMID whose metadata call fails or returns no title: mark **[UNVERIFIED — excluded]** and keep it out of the document.

Scope note. Six searches can return thirty candidates for a five-paper review, and verifying all thirty triples the cost for no effect on the output. Two-stage screening is fine: drop the clearly off-topic ones using the title and abstract the *search itself* returned, then fully verify everything you are seriously considering, with a comfortable margin above target so there is room to swap. Log the pre-screened PMIDs with reasons. The guarantee that must not bend runs the other way: nothing reaches the document unverified.

Step 3b — Price the Depth Option

Two cheap calls, before the checkpoint, so the offer carries real numbers. Neither retrieves article text.

1. `convert_article_ids` on the included PMIDs → which have a PMCID, and every DOI.
2. `python <skill-dir>/scripts/fetch_pdfs.py --probe --email <user's email>` → prints `MODE=direct` or `MODE=mcp-only`. **Exit code 3 means `mcp-only`, not a failure** — read stdout, not the status.

The mode changes what the options cost, and by enough to change which ones are worth offering:

	`MODE=direct`	`MODE=mcp-only`
Filing PDFs	Near-free — one script call, seconds per paper, no article text enters the conversation.	Expensive — the only way to make a file is to pull PMC full text through the conversation, ~10–20k tokens per paper.
Using full	Expensive — the PDFs have to be read, ~10–20k tokens per	Already paid for — the text came through to build the file, so using it

Step 4 — Human Checkpoint

STOP HERE. Present the verification log, the included paper list (PMID | Author | Title | Year | Journal), a brief coverage statement naming any gaps, and the depth offer.

Retrieval is the slowest and most expensive part of the review; doing it for papers the user is about to strike wastes their time, context, and disk.

The three depths

Refer to them **by name, never by number** — the menu differs by route, so numbers drift and a mis-keyed "2" silently buys the wrong thing.

- **Review only** *(default)* — write-ups from abstracts. No extra cost.
- **Review + PDFs** — a folder of papers to read yourself; write-ups still from abstracts. *Direct route only.*
- **Review + full text** — the papers filed *and* critiques built on their actual methods and results.

Present it roughly as:

Step 5 — Write the Word Document

Use the docx skill. Write it to the user's working directory as `<topic>_lit_review.docx` unless they said otherwise — Step 4b.1's `literature/` default sits alongside this file, so it needs a known home.

What you may write from

Every factual claim must trace to text retrieved this session — an abstract, or under **Review + full text**, a paper's full text. Nothing from training memory, however familiar the paper feels.

Under **Review only** and **Review + PDFs**, write from abstracts. **Do not open the filed PDFs to enrich the write-ups** — that is the other depth's cost, and quietly paying it defeats the choice the user made. If the abstracts leave you wanting the methods, say so at the end and offer to upgrade. (On `mcp-only`, if a user insisted on **Review + PDFs**, the text has already passed through to build the files — have the subagents return only their manifest rows, not the extraction, so the analysis genuinely never enters the conversation. Write from abstracts as asked and note in one line that the retrieved text was left unread, so they can change their mind cheaply.)

Document structure

1. Executive Summary — up to three paragraphs: key themes, consensus, major uncertainties.

2. Summary Table — one row per study:

First Author	Title	Journal	Year	Results Summary (1–2 sentences)	PMID	PDF

The PDF column appears only when PDFs were retrieved: the filename, *rendered (full text)*, *rendered (abstract only)*, or *not available*. Under **Review only**, drop the column rather than filling it with misses — a column of "not available" reads as failure instead of choice.

3. In-Depth Study Descriptions — three paragraphs per study under *Design & Methods*, *Results*, *Critique*. Group by theme with a Heading 2 per group, with the evidence-base label under each study heading.

4. Contradictions and Tensions — up to 10 specific conflicts across the included studies; name the studies and the findings that clash. Aim for at least three, but if a focused review on a coherent topic genuinely contains fewer, say so plainly

The output

Social Isolation, Living Alone, and Lack of Social Support as Determinants of Outcomes After Post-Operative Delirium

A Verified PubMed Literature Review

Prepared for: Courtney Balentine, MD

Project: Pilot feasibility study of iConnect for mitigation of post-operative delirium

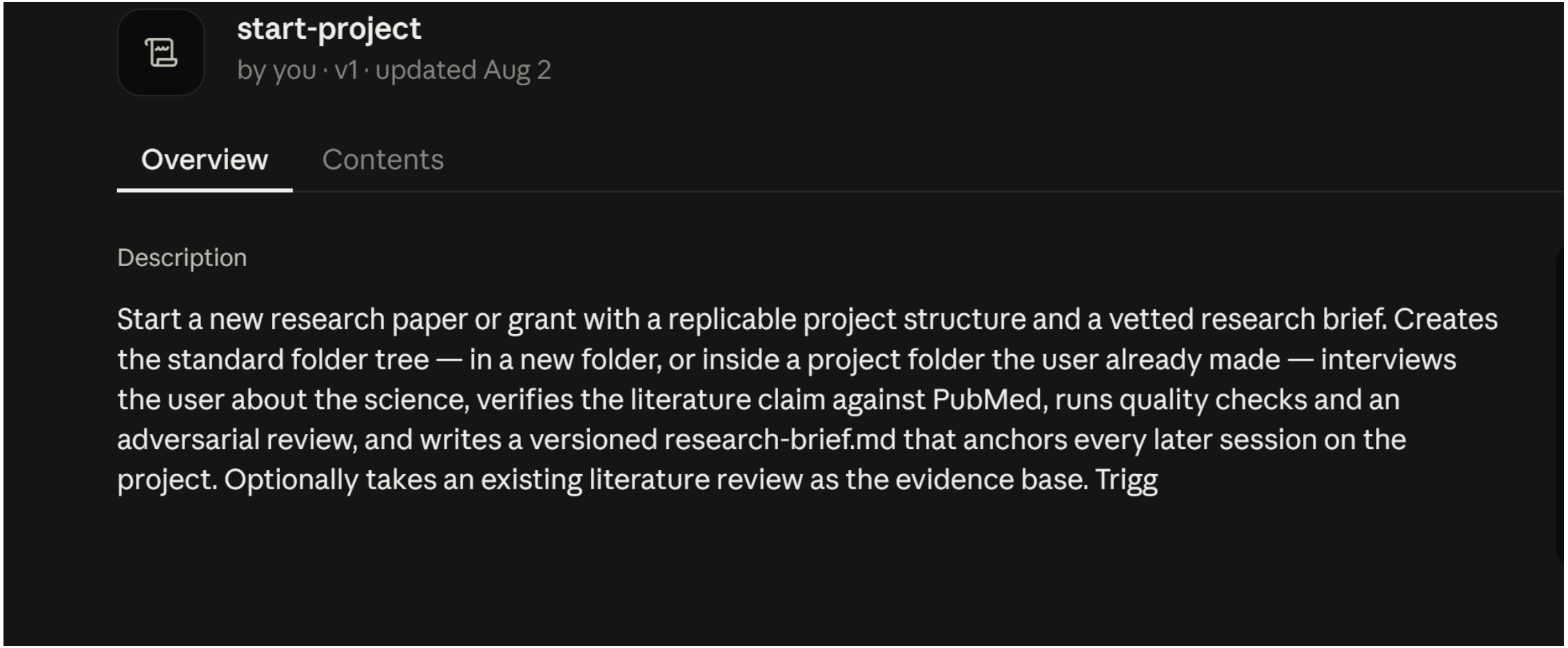
Date: August 1, 2026

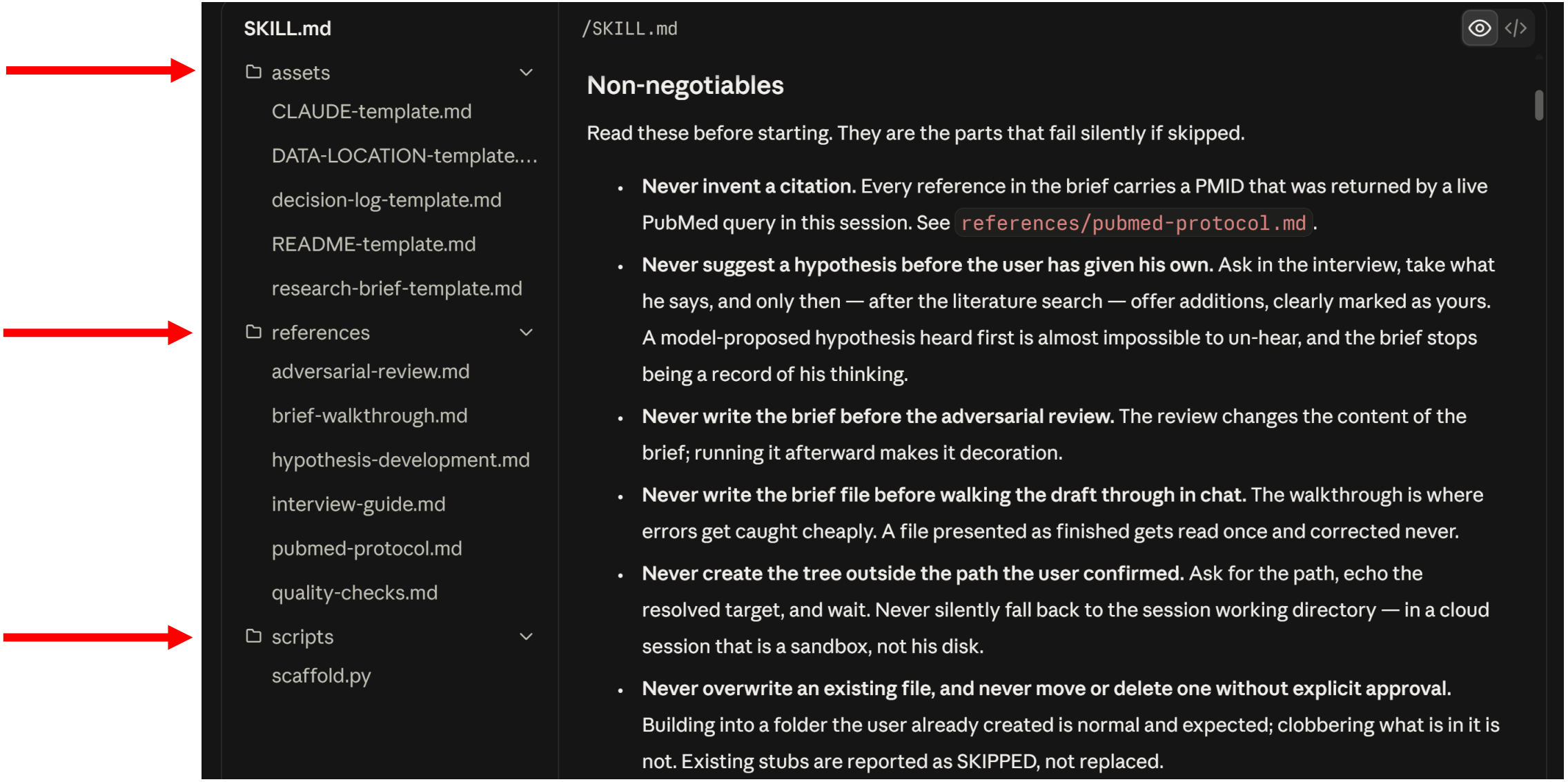
Search domain: PubMed, 2000–present. 43 studies verified and included. Every PMID in this document was retrieved live from PubMed during preparation and individually reverse-verified against its actual title, author, journal, and year before any content was written. The audit trail appears in Appendix A.

Next step: pass to the project planning tool

- Sets up folder and file architecture
- Collaborate to generate objectives, aims, and hypotheses
- Brief literature review or confirmation of completed review
- Generate preliminary outline of methods & analysis plan
- **Adversarial review of the project**
- **Identify threats in design to mitigate**
- Timeline + assigned to-dos
- Creates research brief: guides all future interactions with Claude
- Creates decision log to track important decisions & rationales

Planning tool instructions




SKILL.md
assets
CLAUDE-template.md
DATA-LOCATION-template....
decision-log-template.md
README-template.md
research-brief-template.md
references
adversarial-review.md
brief-walkthrough.md
hypothesis-development.md
interview-guide.md
pubmed-protocol.md
quality-checks.md
scripts
scaffold.py
/SKILL.md
Non-negotiables
Read these before starting. They are the parts that fail silently if skipped.
Never invent a citation. Every reference in the brief carries a PMID that was returned by a live PubMed query in this session. See references/pubmed-protocol.md.
Never suggest a hypothesis before the user has given his own. Ask in the interview, take what he says, and only then — after the literature search — offer additions, clearly marked as yours. A model-proposed hypothesis heard first is almost impossible to un-hear, and the brief stops being a record of his thinking.
Never write the brief before the adversarial review. The review changes the content of the brief; running it afterward makes it decoration.
Never write the brief file before walking the draft through in chat. The walkthrough is where errors get caught cheaply. A file presented as finished gets read once and corrected never.
Never create the tree outside the path the user confirmed. Ask for the path, echo the resolved target, and wait. Never silently fall back to the session working directory — in a cloud session that is a sandbox, not his disk.
Never overwrite an existing file, and never move or delete one without explicit approval. Building into a folder the user already created is normal and expected; clobbering what is in it is not. Existing stubs are reported as SKIPPED, not replaced.

Next step(s)

- Generate statistical analysis plan or project protocol
- Generate IRB
- Analyze data
- Write up results

Creating an IRB writing skill

- Start with your institution's manuals, policies, and forms
  - Ideally, available via web link to maintain updates
- Helpful to have some examples of IRBs
  - Include final product + early stage submission + comments + revision
- Can supplement with answers to common questions + pearls from experts
- Add additional topic specific guides and documents as needed
- Spawn subagent to review the application after it's drafted

Advantages of using LLM to write and edit your IRBs

- Automatically inserts necessary language
- Can provide guidance for complex decisions
- Advice on how to submit/categorize your application
- Can also prepare necessary documents
  - Consents
  - HIPAA waivers
- Can generate subagent “review” before your actual review
- **Consistency checking**
  - Internal: make sure you don’t break new things with each edit or response to IRB comments
  - External: make sure IRB doesn’t contradict grant, protocol, etc.

Creating a protocol/analysis plan skill

- Make the skill specific to one set of methods
  - Quantitative or qualitative/mixed methods; not both
- Encourage the LLM to suggest alternative designs and rigorously critique your proposed design
  - Last chance to catch screwups before you start working
- May be helpful to include sources for context if you are employing specialized methods
  - Less helpful for more commonly used approaches
- Clearly lay out the content and formatting of the final document
- Analysis of quantitative data
  - Can generate starter code in Stata, python, R, SAS, etc.
  - Can also pass the plan to Claude Code for execution
- Can also spawn subagent for “independent” review of methods

Analyzing quantitative data within Claude

- Can connect Claude Code and CoWork to common statistical packages via MCP
- Feed Claude the analysis plan +/- grant +/- other key context and tell it what you want
- Helpful to provide data dictionary
- For secure data (i.e. you can't give Claude access), can use data dictionary to generate simulated data to test your code
- Can ask Claude to summarize results to help interpretation
- **Always** have Claude annotate code that it generates
  - So you understand what's happening
  - So you and your team/statistician can do quality checking
- **Always** check the work

Writing and editing with LLMs

- **Don’t have it draft the paper for you**
- Works better as a brainstorming partner
  - Here’s what I want to say in the first paragraph…
  - I’m struggling to say X well, suggest some ways…
- Fantastic editor
  - Suggest alternative phrasing
  - Cut excess verbiage
- Again, helpful to give the skill exemplars of writing
  - Can be your writing
  - Can be papers/grants/abstracts from someone(s) you think does a great job

Notes on safety

Can LLMs steal my stuff?

- Sort of
- Using most LLMs involves sending information out into the cloud
  - But so does email, OneDrive, Google Drive, Box, etc
- Dangers of allowing companies to train their models on your data
  - Universities generally opt out of this for you
  - You can opt out on any personal subscriptions
- Bottom line: don’t give LLM any PHI or proprietary research, but otherwise it’s (probably) safe



Tasks that AI sucks at (currently, but may change rapidly)

- Managing references
  - My advice: don't ever let LLM touch your references
- Basic math (practical applications)
  - You're better off using a calculator for basic math
- Epidemiological data
  - Numbers may bear no relation to reality because LLM's don't actually see numbers
- Doing statistics (though they are improving rapidly)
  - But excellent performance at generating code to do statistics
- Staying up to date on evidence
  - Different if have access to PubMed tool

Bottom line: thoughtful use of AI can

- Reduce stress by taking over and automating repetitive tasks
- Help you refine and improve the quality of your outputs
- Save you time that can be used to enhance your quality of life
  - As long as you don’t fall into the productivity trap

The Dream



Useful resources

- AI Prompt Engineering by Nelson Ming
- Building Claude Skills by Pietro Montaldo
- Claude CoWork for Beginners Made Easy by David M. Patel
- Agentic Spec Driven Development by Anatoly Volkhover

Thank you

