integreat

Advance chatbot use

- with RAG example

Håkon A. Taskén

Generative AI Chatbots

- ChatGPT reached 100M users faster than TikTok after launch in 2022
- Rapidly improving
- Today that are super useful!
- Will learn: **How to get the most out of chatbots**
- Tips mostly based on Andrej Karpathy's video explaining his use pattern

Overview

- This session will be mostly exercises
- Red thread: Run a RAG example
  - Very ambitious
  - Demonstrate power: Chatbot allow you to do difficult things you don’t know
  - Learn to handle things that don’t work right away
- Process:
  - I present some usage tips or feature of current chatbots
  - Exercises displayed then on powerpoint
  - You will test out on your own
  - I will give an example solution
  - We will all use my example as starting point for next exercise

Many Good Chatbots

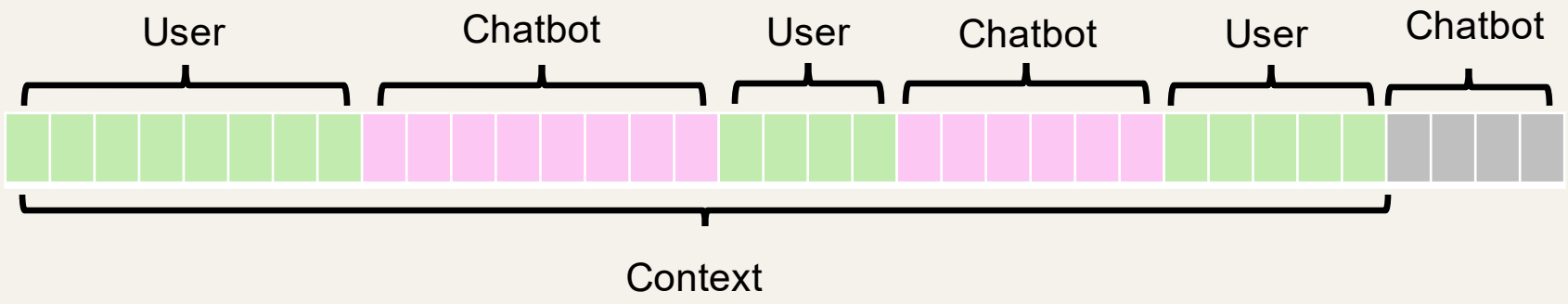
- **ChatGPT** – most popular and feature rich
- Gemini by Google
- Meta AI by Meta
- Copilot by Microsoft
- Claude by Antropic
- Grok by xAI
- Perplexity
- DeepSeek from Chinese company
- Le Chat by French Mistral AI

Intro Exercise

You have just been told about RAG. Let the chatbot be your personal tutor and get it to explain you something you don’t understand about RAG.

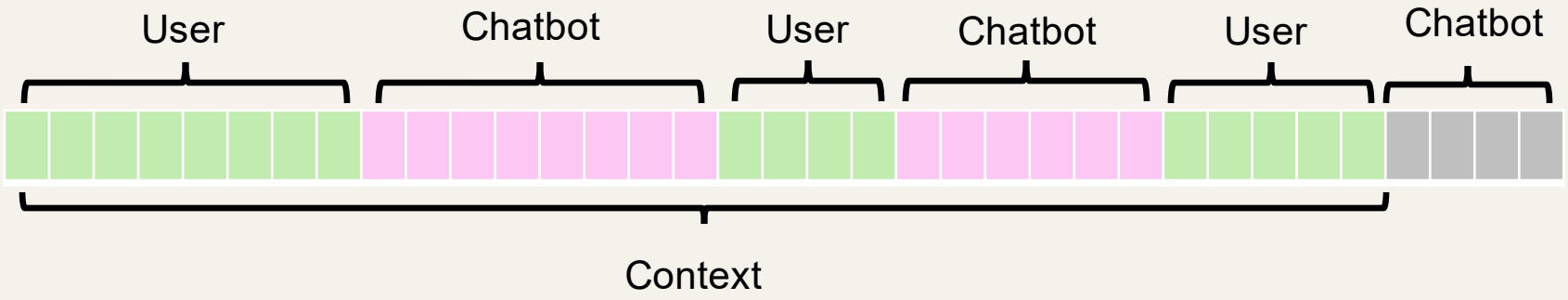
Mental model

- User and chatbot collaborate to make in a single piece of text taking turn writing.
- Chatbot tries to make the text look like the internet up to some cut-off date and examples of how humans want answers based on context.
- Chatbot have imperfect memory and makes **random,** best effort guesses.



Take-away from the mental model

- Need to provide **all** needed information in precise manner
- Much unnecessary information may confuse it
- It does not know about things after a cut-off date (without search)
- It hallucinates – makes stuff up
- Exists a maximum context length



When not appropriate to use?

- High stakes situation (it can be completely wrong)
  - Situations can often be made low stakes by checking its answers
  - Useful when it is faster to check answer than create it yourself
  - Pitfall: Easy to don't check properly
    (Sounds good and confident even when wrong)
- Recent information (without search)
- Things not written much about on the internet
- Private/sensitive data – unless using local LLM
- For simple google searcher – it uses more power

Usage Tips

- Explain all useful information precisely
  - State specifically what you are after
- Start new chat when start new topic
- For multiple question it is better to ask them separately
- Check results before trusting it
  - Evaluate: is it common on the internet
- Ask it for help!
  - State things that are wrong or lacking
  - Iterate and refine answer until satisfied

Exercise: RAG problems

Learn about the shortcomings of RAG.
Iterate if answer not to your liking.

Overall exercise goal

Find 10 events where first responders was killed in armed conflicts in Europe after 2000 using data from Uppsala Conflict Data Program and RAG.

Exercise: Feasibility

Find 10 events where first responders was killed in armed conflicts in Europe after 2000 using data from Uppsala Conflict Data Program and RAG.

Is this possible when one don’t know how to code, but one can copy paste code and has the help of a chatbot?

If possible were, is it easiest to run the RAG solution?

Web search + other tool use

- ChatGPT can search the web
- Can fix problems of recent information
- Can improve results for less frequently information
- Automatically or force manually web search
- ChatGPT can use calculator and make plots

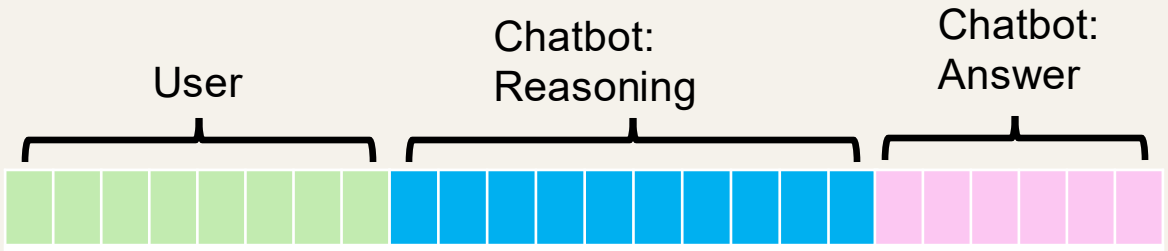
Exercise: UCDP size

Find the size of the Uppsala Conflict Data Program database.

For ChatGPT plus users: Try to plot the size by year.

Reasoning models and model choice

- Smaller models are less capable, but faster
- Reasoning models perform better for task requiring reasoning
  - Structured problem solving
  - Multi-step logic
  - Conceptual understanding
  - Planning and strategy
  - Math, coding, etc.



Exercise: Generate some code

Find 10 events where first responders was killed in armed conflicts in Europe after 2000 using data from Uppsala Conflict Data Program and RAG. Generate self containing code for pasting into Google Colab that download the data and uses RAG to find entries we are after. Make sure code run fast and download fast so it can be used as a quick and free demo without any user input.

Then copy the code and run it in Google Collab.
Ask the chatbot for solutions to hiccups (there will be several).

Deep Research

- Combine web search and reasoning models
- Asks clarifying questions
- Gives a long report
- Takes a long time

Deep research example

Prompt: Explain the typical problems one encounter when using RAG.

Clarify: Legal domain

Used over 3 min and 48 sources

Output: 9 pages PDF

ChatGPT

Challenges in Retrieval-Augmented Generation (RAG) for the Legal Domain

Introduction

Retrieval-Augmented Generation (RAG) is an AI approach that combines a large language model with an external knowledge base or document repository to "ground" the model's outputs in relevant source material. Instead of relying solely on the model's internal training (parametric memory), a RAG system first retrieves pertinent documents (e.g. case law, statutes, regulations) and provides them as context for the model to generate a response [1]. This technique has strong appeal in the legal domain, where answers must be precise and supported by authority. Legal applications are well-suited to RAG given the abundance of high-quality legal databases and the need to incorporate up-to-date information (for example, newly decided cases or amended laws) without retraining the model [2]. In theory, grounding a generative model in authoritative texts should improve accuracy and reduce the risk of "hallucinations" – the model making up facts or citations [3] [4]. Indeed, legal technology providers have marketed RAG-powered research tools as *"hallucination-free"* or highly reliable assistants [5] [6]. In practice, however, deploying RAG in the legal field brings a host of challenges. This report analyzes the common problems encountered with legal RAG systems, including issues in retrieval precision, model hallucination of legal content, maintaining legal context, scalability, jurisdictional and temporal pitfalls, data privacy and compliance concerns, and the difficulty of evaluating output correctness. Each challenge is discussed with examples and findings from recent research and case studies.

Retrieval Precision and Relevance in Legal Research

One fundamental challenge is ensuring the retrieval component of a RAG system finds the *truly relevant* legal documents or passages with high precision. Legal queries often require pinpointing specific rules or precedents buried in lengthy texts. If the retrieval step fails to fetch the right authority, the generation step will be answering on a flawed foundation. Unfortunately, **legal retrieval is hard** – the law is not a simple database of facts, but a complex web of opinions and context that can make it difficult to identify a definitive answer [7]. Studies indicate that even advanced legal RAG systems sometimes retrieve documents that are only superficially relevant to the query (e.g. containing similar keywords) but not applicable in context (for example, a case from the wrong jurisdiction or an outdated statute) [8]. In a recent benchmark, researchers emphasize that retrieving **precise snippets** of text (as opposed to large, imprecise chunks) is crucial: broad retrieval results can overflow the context window and even introduce irrelevant information that confuses the model [9]. An insightful analogy from Thomson Reuters compares this to a law student using a bad casebook – *"just as a law student using an outdated textbook will give wrong legal answers, an LLM using RAG without good sources will generate unreliable content"* [10]. In other words, poor retrieval or low-quality context can nullify the benefits of RAG. Legal RAG systems therefore must prioritize **high recall and high precision** in search, often trading speed for thoroughness. For instance, legal-specific retrieval models use domain-tuned text embeddings and may perform exhaustive re-ranking over the entire corpus to ensure the most on-point results are selected [11] [12]. This focus on precision reflects the legal profession's willingness to tolerate higher latency for better accuracy [13]. Ultimately, a

1

Exercise: Fix Error in Code

Copy errors messages into the chat and generate new code. Repeat.

Multimodal Models and Image Generation

- Multimodal Models like ChatGPT 4o understands images and sound
- Voice mode: talk directly to the chatbot
- Excellent for producing summaries even of PDFs
- Generate images and illustrations

Summary of Deep Research Report

Summary: Common Challenges of RAG in the Legal Domain

Using RAG in legal applications offers benefits like grounding answers in legal texts, but it also presents serious challenges:

Retrieval Accuracy: Legal queries require highly precise retrieval of statutes or case law. Standard search may return superficially relevant but legally inapplicable content (e.g., wrong jurisdiction or outdated law).

Hallucinations: RAG reduces but doesn't eliminate hallucinations. AI may fabricate citations or misrepresent what retrieved documents actually say.

Loss of Legal Context: Breaking documents into small chunks can strip critical legal definitions or relationships, leading to misinterpretation of laws or cases.

Scalability and Latency: Large legal corpora and long documents slow down retrieval and generation. More relevant text often increases cost and confusion.

Jurisdiction and Timeliness: Law is jurisdiction-specific and changes over time. RAG systems may surface irrelevant or outdated legal authorities without recognizing their invalidity.

Privacy and Ethics: Handling sensitive legal data raises confidentiality risks. AI tools must comply with data protection laws and ethical standards.

Evaluating Accuracy: It's hard to measure whether RAG outputs are legally correct. Legal professionals must often verify AI responses manually.

Exercise: Try to make it work

Find 10 events where first responders was killed in armed conflicts in Europe after 2000 using data from Uppsala Conflict Data Program and RAG. Generate self containing code for pasting into Google Colab that download the data and uses RAG to find the entries we are after. Make sure code run fast and download fast so it can be used as a quick and free demo without any user input.

Then copy the code and run it in Google Collab.
Ask the chatbot for solutions to hiccups (there will be several).

Thank you for your attention and effort!