Hi, My name is Nat Gustafson-Sundell. I’m a Collections Librarian at Minnesota State University Mankato (MNSU), a regional, public, comprehensive university of about 14,000 students.

Today, I will present the steps we have taken to develop a prototype AI assistant for license review. I’ll explain our criteria for the selection of an AI tool for this project. We reviewed ChatGPT, Claude 2, Bard, and PDF readers. My goal was to develop an initial prototype in a Jupyter Notebook environment so I could easily re-load context information, including a license checklist, but I’ll explain why I revised this goal, instead to linger over license review interactions with ChatBots. I’ll discuss early results, demonstrate example license review interactions, and outline my next steps.

This type of project would be easy for others to customize to meet their own needs, so, in this video, I won’t dwell on licensing as such. Instead, I’ll focus on the implementation issues.
Sometimes it can be difficult to catch everything in a video. If you’d like to download my slides, as well as my notes, you can find them at “link.mnsu.edu/ailicensedeck”
I will describe the basic idea for my project, but first, I should provide some background. Hopefully, you are familiar with ChatBots, such as ChatGPT. At their most basic, ChatBots generate text responses to natural language queries. They have been trained on enormous quantities of text, including crawls of the internet over many years. ChatBots don’t have any long-term memory, although the ChatBot interface might retain old conversations.
The Library & Generative AI
August 25, 2023 (for Minitex/ ERMN librarians)

- ChatGPT demonstration & prompts
- Prompt generator demonstration
- Quick overview of ChatGPT as an information source
- ChatPDF demonstration for article summarization
- Consensus demonstration for article discovery

See link.mnsu.edu/aideck

By the way, if you haven’t had much time yet to investigate ChatBots, you might want to take a look at a presentation I gave to librarians in my region. The slides are available at “link.mnsu.edu/aideck.” It was an introductory presentation covering basic prompting and a couple of other AI tools. I also incorporated a section on the limits of ChatGPT as an information source which I prepared for a professional development session to returning faculty through our University’s Center for Excellence in Teaching & Learning.
When prompting ChatBots, it’s possible to engineer very specific kinds of responses in an array of formats. If you’ve spent anytime studying prompt engineering, you’ve probably run across approaches such as chain-of-thought and skeleton-of-thought prompts, preference maps, and so on. The point of these approaches is to define clearly what is expected of the ChatBot, so that the ChatBot responds with the right kind of information, in the right format, with the right voice, etc.
It is also possible to interact with ChatBots through a Jupyter Notebook. A Jupyter Notebook is an environment for programming, initially developed, I think, so that code and data could be shared for the replication of research. On the screen, you can see the first few cells of a Jupyter Notebook I developed several years ago for data processing. A Jupyter Notebook can be a repository for consistent inputs to a ChatBot. So, for example, you could create a preference map to load every time you use the notebook. A preference map is basically just a list of settings and expectations.
My inspiration to use Jupyter Notebook to interact with a ChatBot came from an online class I took over the summer, entitled “ChatGPT Prompt Engineering for Developers.” I learned that it’s possible to interact with ChatGPT from within Jupyter Notebook using an API key. An API key is just a programmatic means of access to the ChatBot.
So then, my initial idea was pretty straightforward. I would create a Jupyter Notebook to interact with a ChatBot to review licenses. I would adapt our licensing toolkit as a checklist of terms which I could re-load every time I used the notebook. The Jupyter Notebook would only require a few pieces, first the API key, second a checklist of terms, third a preference map or set of instructions for the responses. For any given session, I’d also load a license. After loading the license, I’d simply interact with the ChatBot to review terms. Over time, I could refine the checklist and preferences. This process would not replace my own review of the license, but it could complement that review. The new process might help me perform more consistent, careful reviews.
I began the project by investigating which of the leading ChatBots I should use to implement the project. This investigation was very simple. I just asked ChatGPT, Claude, and then Bard how I would go about working with them. On the screen, you can see the start of my conversation with ChatGPT.
Basically, they each said they’d love to help me and they each provided information about how to proceed, including suggested code to set up Jupyter Notebook. Here you can see the same start to the conversation with Bard.
These 3 ChatBots are definitely not all the same. They have distinct advantages and disadvantages for a project like mine. My first impression was that ChatGPT would not be the best option for my project. At this time, it is only possible to input PDFs into the subscription version of ChatGPT using plug-ins and then only through a URL. Text inputs are limited in size, less than 2000 words. Although ChatGPT offered a work-around, the work-around was not optimal and ChatGPT admitted it, as you can see on the screen. Basically, I would have needed to break text inputs into chunks to enter them into ChatGPT.
All of the ChatBots offered tips about how to implement the project. In all 3 cases, the ChatBots said I could access them via Jupyter Notebook using an API key. I don’t have time now to go through each of these conversations in detail, even though they were each pretty brief, but if you’re interested in reading the transcripts, I’ll provide the links on the next screen.
Claude seemed to me to be the best ChatBot to use. Claude makes it easy to upload PDFs. Claude’s instructions for implementation were very clear. Claude also was really enthusiastic about the project. I’ve worked the most with ChatGPT in the past, and I like it, but Claude just seemed best for the purposes of this project.
After getting a sense of how each of the ChatBots could work, I proceeded to practical steps, but here I ran into a problem. Although Claude had made it sound easy to get an API key, it turned out I couldn’t. In fact, for both Claude and Bard, I have had to apply for an API key, which is a beta feature right now for each of these ChatBots. I haven’t yet received a reply from either Anthropic or Google, the companies behind Claude and Bard respectively.
It is possible for me to get an API key to ChatGPT, but only by paying more money. I’m currently a subscriber to ChatGPT+ at $20 per month. This version does not provide an API key. It’s necessary to add a pay-as-you-go plan to another version. Before doing this, I wondered if there were any other options I hadn’t tried?
Because I’ve been very busy on several projects, my thinking has been compartmentalized. It took me a little while to realize that I didn’t need to proceed to the full Jupyter Notebook implementation to make progress on my project. I could instead just interact with a ChatBot over a license. And actually, as I started doing so, I realized this probably should have been part of the plan all along – it’s important to test and refine one’s prompts. Later on, when I can proceed with the Jupyter Notebook implementation, I’ll be better prepared to design my inputs.

By the way, the image on the screen is meant to illustrate the notion of compartmentalized thinking. I prompted Bing Image Creator with the phrase “numerous small furry monsters in compartments.” I don’t know why I naturally thought of my thoughts as small furry monsters, but now that I see them on the screen, I know this is definitely what my thoughts look like...
I started testing with ChatPDF. I was already familiar with this tool because it is popular for journal article summarization, an issue I addressed in a presentation to regional librarians, which I previously mentioned. Although ChatPDF is optimized for PDFs, it’s actually not a great tool for my license review project. ChatPDF is limited to prompts of about 90 words or less. The limit on words made the prompts quite lean.
The license is divided internally into sections. These sections might be demarcated by numbers, letters, or spacing. Extract any sections that refer substantially or mainly to the following concepts or terms, which I’ve demarcated with angle brackets. Wait after presenting each term. I will either ask questions or ask you to continue presenting terms. (1) <definition of authorized user>, (2) <definition of site>, (3) <renewal>, (4) <term>, (5) <termination>, (6) <financial hardship>, (7) <confidentiality> or <non-disclosure>, (8) <accessibility> or <ADA compliance>, (9) <transfer> or <previous subscriptions>.

On the screen, you can see the first draft prompt I developed when I was still working with ChatPDF. The prompt incorporates some preferences for the format and delivery of the response, but the approach is far simpler than I initially envisioned because I wrote it to conform with the limits of ChatPDF. In Jupyter Notebook, I probably would have started with something much more verbose, so I’m glad I had to rethink the project because I might have frontloaded a lot of effort trying to get ideal responses, but the reality might not have been better than I can get with simple interactions.
After my initial tests with ChatPDF, I paused the project to prepare an internal presentation for my colleagues on a variety of AI topics. I wanted to share what I was learning from this project, as well as other AI projects – one of which was a 5-hour workshop for high school students, which was way more fun, as you can see on the screen. One reason for the pause was that I wanted to solicit feedback and hear concerns about the license review project before getting too far ahead of myself. I should mention in this context that I have been careful to test only with licenses that are publicly available on the internet.
The pause was helpful because when I came back to the project, I saw that I had been compartmentalizing again. I had thought to use ChatPDF because of my previous familiarity with the tool, but, of course, Claude can also accept PDF inputs, as I previously mentioned. I knew this in one part of my mind, but when I ran into the problem with API keys, I started looking in different directions altogether.

Working with Claude, I’ve continued building on what I learned from ChatPDF. I haven’t done anything fancy yet, but I have learned there’s value simply interacting with Claude over a license. In a very basic sense, this gamifies the work, if only because the interaction makes the work a little more enjoyable and because it causes me to see the licenses somewhat differently. Let’s move onto a live-action demo of Claude so you can see how this works.

(The following slides are screenshots from Claude. In the video I demonstrate Claude in live-action, but I also wanted to provide my notes in these slides, in case they can be helpful to anyone.)
This demonstration will be very brief and basic because I’m limited for time, but it should provide a good enough sense of how anyone can get started. First, I’ll start by inputting a license into Claude. I’ll then enter one of the current versions of my starting prompt.
This prompt is still based on the lean version I developed for ChatPDF. Basically, I’ve revised the prompt so that the mechanics of my interaction are correct. With earlier versions, Claude sometimes presented term references incorrectly or presented all terms at once. My goal was for Claude to pause over each of my terms of interest, which it now does. Curiously, the responses have improved as I’ve simplified the prompts so far, but I’m glad to know I’m not limited to 90 words as I was in ChatPDF, because, in the future, I will probably expand the checklist and associated context.
I've gone back and forth deciding whether I prefer Claude to pause over each term on my checklist or whether I want Claude to provide a full list. I can see virtues in both approaches. For now, however, I’m pausing over each term, so that I can ask for follow-up – this helps me learn about the license and it also helps me think about how much I trust Claude to catch everything related to the terms in question.
I’d like to review a license. I’m interested in certain concepts. Please extract references to these concepts. After you present the first concept, please pause and ask me if I have any questions. Present each concept one at a time, pausing after each for questions. Wait for me to ask questions or say continue. Here are the concepts: (1) Definition of authorized user, (2) Definition of site, (3) Renewal, (4) The term period of the license, (5) Termination requirements, (6) Financial hardship, (7) Confidentiality or Non-disclosure, (8) Accessibility or ADA compliance, (9) Journal transfers or Previous subscriptions.

To this point, I have only tested and tweaked my basic prompt and associated follow-up. So far, I think it’s helpful to interact with Claude to read licenses because it helps me slow down and check for the terms that typically need extra attention. I definitely could not replace my own careful reading of the license at this time, but the interaction is enjoyable and sometimes enlightening. These interactions are also leading me to think about how I can improve my checklist. An additional next step will be to work on a formatted output. If Claude outputs a report of terms requiring follow-up, I could revise and augment the report based on my own careful reading.
I'd like to review a license. I'm interested in certain concepts. Please extract references to these concepts. After you present the first concept, please pause and ask me if I have any questions. Present each concept one at a time, pausing after each for questions. Wait for me to ask questions or say continue. Here are the concepts: (1) Definition of authorized user, (2) Definition of site, (3) Renewal, (4) The term period of the license, (5) Termination requirements, (6) Financial hardship, (7) Confidentiality or Non-disclosure, (8) Accessibility or ADA compliance, (9) Journal transfers or Previous subscriptions. At the end of this conversation, I'll ask you to produce a list of concepts with your comments. As we converse, if I want one of the concepts included on the final list, I'll type <Please note>

And, actually, as I was working on this video, I realized how easy it would be to have Claude produce a list of concerns, at least as a first step toward a report. I added two sentences to the prompt...
As you can see on the screen, I simply needed to ask Claude to note concepts as we proceeded.
At the end of the conversation, I just asked for a list and Claude provided it. I’ll probably play around with this a little more, but I like where it’s going so far...
I still don’t know when or if I’ll return to the idea of using a Jupyter Notebook for license review specifically, although I do think Jupyter Notebook would be a good environment for some other AI projects. This has been a sort of hobby project which I have advanced in little bits and pieces when I’ve had time, but right now, I do feel confident utilizing Claude in my real-life license review process. I also think others could easily implement Claude for their own purposes, but maybe not help with jokes to end presentations. As you can see on the screen, Claude isn’t quite ready for stand-up.
Anyway, thank you for bearing with me... and I want to thank NASIG for the opportunity to present what I’ve done so far.

If you’d like to talk, my email address is on the screen.