

# Preparing Content for Custom AI Chatbots

Updated November 1, 2024

## Overview

A custom AI chatbot that leverages retrieval augmented generation (RAG) can help students work through course content, answer FAQs, and act as a tutor. It can save time for both instructors and students by automating routine queries. Additionally, it can facilitate continuous learning outside class hours, help to clarify concepts taught in class and can be prompted to quiz students to check their understanding of material.

For more information about creating custom AI chatbots using non-supported AI tools, including essential considerations around cost, privacy, security, copyright, and intellectual property, see CTSI's "[Engaging with Non-Supported Generative AI Tools](#)" resource.

Creating a custom AI chatbot starts with content preparation. Well-organized content promotes accurate, relevant responses and better handling of domain-specific knowledge related to course materials. Following best practices in content preparation improves the accuracy, relevancy and speed of responses.

## Sources of Content

### Supported File Types

Depending on the platform used, different file types may be supported. Check the platform's documentation for supported file types for uploading. As of October 2024, commercially available RAG-based platforms (e.g., Microsoft Copilot Studio, OpenAI GPT Builder, Google NotebookLM, etc.) have varying levels of support for different file formats. However, all support machine-readable PDF and plain text upload. Generally, images, videos, and audio files will not be processed – if sources contain media elements, they will be ignored (or at best, a text transcript of audio/video will be processed if available).

The following links will direct you to currently supported file types and limitations for various AI platforms. This is not an exhaustive list, and only represents select tools currently available in Canada:

- [Microsoft Copilot Studio](#)
- [OpenAI GPT Builder](#)
- [Google NotebookLM](#)



© 2024, Centre for Teaching Support & Innovation. Except where otherwise noted, this work is made available under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International \(CC BY-NC-SA 4.0\) License](#).

## Suggested Course Documents

The following types of course documents may be considered for uploading to a custom AI chatbot. As a reminder, never input confidential information or significant portions of intellectual property that you do not have the rights or permissions to (including U of T library licensed e-resources) into tools outside of the [“walled garden” of U of T’s protected digital environment](#). See the U of T Library’s [Generative AI tools and Copyright Considerations](#) resource for up-to-date guidelines on copyright and intellectual property.

- Course-specific materials (lecture notes, slide text, course assignments, lab manuals, etc.)
- Transcripts from lecture recordings
- Industry or subject or domain-specific documents
- Supplementary reading materials (might include articles, reports, case studies, online resources)
- Textbooks
- Research papers from your field
- Dialogue data (chats, forums, or other dialogue-heavy platforms)
- General knowledge (encyclopedias, Wikipedia)

For web links, note that the AI solution will not be able to access pages that require a login, including Quercus pages and modules.

**CTSI does not typically recommend uploading the course syllabus into a custom AI chatbot.** Although uploading course materials into RAG-based platforms will generally reduce the rate of hallucinations (i.e., factually inaccurate outputs), AI-generated text is not guaranteed to be correct. Students should be reminded that an AI chatbot should not be treated as an “authoritative source”. Students should continue to refer to the course syllabus directly for key dates and course policies.

## Content Preparation Best Practices

For content that you have the ability to edit, it is essential to prepare it for input into an AI virtual agent. In general, more contextualized and structured input will lead to more coherent output, as generative AI models are becoming increasingly sophisticated in their capabilities to parse the information architecture of source documents.

### Curate and Format Text

- **Remove noise** – Remove any irrelevant characters or information that doesn’t add value, including HTML tags, emojis, or elements that do not contribute to the information architecture of the document.
- **Fix typos and transcription errors** – Use a program’s built-in spellcheck and review any transcripts of multimedia content for errors. Automatically generated transcripts (e.g., via YouTube, Zoom or MS Teams) generally require human review and correction.
- **Remove excessive white space** – Removing extra space, which is considered noise, can lead to more efficient data storage and faster processing times.

- **Remove duplicate or redundant content** – Make data more efficient for the model to process and retrieve relevant information quickly. This ensures that the model focuses on the most pertinent data, improving the accuracy and relevance of its responses. Additionally, a cleaner dataset reduces computational overhead, leading to faster processing times and better overall performance.
- **Ensure that PDF documents are machine-readable** – If a document has been scanned, use Optical Character Recognition (OCR) to convert images into machine-readable text. This conversion is necessary because the model requires text data to process and retrieve information.
- **Avoid multi-column documents** – Current models generally struggle with parsing multi-column documents (such as PDFs of academic journal articles). If possible, source a version of the document that is structured as a single column.
- **Utilize LaTeX** – For technical, mathematical, and scientific notation, most AI models support parsing [LaTeX format](#).

## Information Architecture

- **Implement document structuring** – Utilize headings and formatting to provide an organizational overview, showing the logical development and hierarchical relationship of ideas within a document. This can be done natively in MS Word and other rich-text editors, and can be accomplished with plain text files (such as lecture transcripts) by using [Markdown](#).
- **Develop a glossary** – If a virtual agent struggles with interpreting jargon, acronyms, and other domain-specific content, develop a glossary document to ensure consistency and correct interpretation of uploaded content.
- **Annotate data** – Provide annotations for data that are not machine-readable. This includes ALT text for images, captions for charts and figures, and explanations for any other non-textual items that contribute to the reader’s comprehension of the content.
- **Anaphora resolution**: address pronouns and references (anaphora) effectively. For example, ensure the models knows who “he” refers to by using actual names.
- **Chunk content** – In cases of upload size limitations, divide very long documents into meaningful chunks. Each chunk should encapsulate a coherent topic or set of ideas. This helps improve accuracy of context-aware answers. Chunking also prioritizes efficiency by breaking down text into smaller units, making it easier and faster for an AI model to parse.

## Data Chunking Example

For a 10,000-page document on Introduction to Art History, chunk the document into smaller pieces that are each 500 words long. This size is manageable for the model to process and still has enough context to be meaningful. Include an overlap of 50 words between consecutive chunks to ensure continuity and context preservation.

- Chunk 1 might cover “Paintings and Sculpture” and the first part of “Architecture”.
- Chunk 2 would start from the last 50 words of “Architecture” and continue to the next section, perhaps “Applied Arts”.

## Content Preparation Pitfalls

1. **Over-Cleaning** – Removing too much information can lead to loss of important context or features that are necessary for the model to understand the data properly.
2. **Under-Cleaning** – Insufficient cleaning can leave in noise such as irrelevant information, special characters, or formatting issues, which can confuse the model and lead to inaccurate results.
3. **Inconsistent Standards** – Without a consistent approach to cleaning, data can end up in various formats, making it difficult for the model to process it uniformly.
4. **Neglecting Data Source Quality**: Using low-quality or unreliable data sources can lead to a garbage-in-garbage-out scenario, where the model’s outputs are only as good as its inputs.
5. **Failing to Update Data**: Not regularly updating the content can result in the model relying on outdated information, which can be particularly problematic for time-sensitive topics.
6. **Biased Data**: If the data cleaning process does not account for biases, the model may produce biased outputs, which can have serious ethical implications.
7. **Overlooking the Importance of Context**: Data cleaning that does not consider the context in which the data will be used can strip away nuances that are essential for accurate generation.

To avoid these pitfalls, it is important to have a well-thought-out data cleaning strategy that includes understanding the data, applying consistent standards, addressing missing values, ensuring the quality of data sources, and regularly updating the dataset.

## References

Anello, E. (2024). How to improve RAG performance: 5 key techniques with examples. DataCamp. Retrieved from <https://www.datacamp.com/tutorial/how-to-improve-rag-performance-5-key-techniques-with-examples>

Glantz, W. (2024). 12 RAG pain points and proposed solutions. Towards Data Science. Retrieved from <https://towardsdatascience.com/12-rag-pain-points-and-proposed-solutions-43709939a28c>

Guler, O. (2023). How to improve RAG performance: Advanced RAG patterns (Part 2). Cloud Atlas. Retrieved from <https://cloudatlas.me/how-to-improve-rag-peformance-advanced-rag-patterns-part2-0c84e2df66e6>

Guler, O. (2023). Why do RAG pipelines fail? Advanced RAG patterns (Part 1). Cloud Atlas. Retrieved from <https://cloudatlas.me/why-do-rag-pipelines-fail-advanced-rag-patterns-part1-841faad8b3c2>

Oviedo, E. R., & Lanza, E. (2024). Four data cleaning techniques to improve large language model (LLM) performance. *Intel Tech*. Retrieved from <https://medium.com/intel-tech/four-data-cleaning-techniques-to-improve-large-language-model-llm-performance-77bee9003625>

Monigatti, L. (2024). A Guide on 12 Tuning Strategies for Production-Ready RAG Applications. Towards Data Science. Retrieved from <https://towardsdatascience.com/a-guide-on-12-tuning-strategies-for-production-ready-rag-applications-7ca646833439>