Project Title: Automated Literature Review

Github Link: https://github.com/ACerullo1996/EGR404-Final-Project.git

Author: Alexander Cerullo

Overview of the Project (Pages 1-3)

Problem Addressed

In neuroscience research, especially with EEG and fNIRS studies, performing literature reviews is time-consuming and labor-intensive. Researchers must manually read, extract, and compare experimental details across many papers. This slows the process of hypothesis development and study design. The project addresses this challenge by developing an AI-driven assistant to automate the reading, summarization, and comparison of scholarly articles, making the early phases of a literature review both scalable and efficient. While not designed to fully remove the researcher, this protocol will create a 'rough draft' from where the user can edit the file and correct improperly phrased results. In addition to summarizing the experimental protocols, this project was also capable of identifying the 'future work' from each paper and using them to generate a new hypothesis and experimental protocol for the research to fine tune and implement.

Methodology

1. PDF Ingestion & Summarization

- Open PDF articles and extract the text aspects of the pdf (ignoring the figures)
- Each article is summarized using OpenAI's gpt-4o, prompted to extract:
    - Metadata (title, authors, journal, impact factor, etc.)
    - Experimental design (tasks, modalities, trial structure, sample size, statistical comparisons)
    - Results and conclusions
    - Future work suggested by authors

2. Meta-Analysis

- Individual summaries are merged and further analyzed using GPT to:
    - Extract shared experimental themes
    - Generate a synthesized overview

3. Propose a testable hypothesis and protocol based on the reviewed literature

- Review the fused summaries and summary overview to detect running themes in the 'results' and 'future work' sections.
- Use these themes to generate a hypothesis and justification
- Review the experimental tendences to generate a likely experimental protocol.

4. Presentation Automation

- Summaries and the synthesized meta-analysis are converted into a structured PowerPoint using python-pptx.
- Display the proposed hypothesis and justification in a two-slide slide-deck
- Display the overall summary of the literature
    - With a four-slide breakdown (title slide, metadata, design, results and future work), automatically formatted.
- Display a four-slide summary of each journal article using the same format as the overall summary.

Results

- Successfully processes PDFs and generates structured, high-quality summaries saved as .txt files.
- Produces a merged file (AllSummaries.txt) and a meta-summary (AllSummaries_MetaSummary.txt) for overview insights.
- Reviews the AllSummaries.txt and the AllSummaries_MetaSummary.txt to generate a Proposed_Hypothesis_and_Protocol.txt.

- Reopens each .txt file (excluding AllSummaries.txt) and automatically generates a clear, structured PowerPoint presentation, suitable for academic or lab meetings.

Challenges

- Rate Limiting: The assistant occasionally hits OpenAI's rate limits during summarization; a retry logic is implemented with delays.
- PDF Parsing: Some scientific PDFs may have poor formatting or embedded images that disrupt text extraction, reducing summary accuracy.
- JSON Decoding Errors: When converting LLM responses into JSON (for slide generation), minor inconsistencies in output formatting occasionally require error handling.
- Evaluation: No automated benchmarking was implemented to compare AI-generated summaries to human-written literature reviews, though this is mentioned as a future validation step.
- Figure Agnostic: This code does not review the figures included in the papers. Any results that were embedded in a figure were unfortunately missed by the text viewer.
- Lack of Citation: While each paper is reviewed the meta summary and proposed hypothesis do not automatically cite the papers contained in the review. Any citations in the output are a direct result of the LLM generation and need to be double checked.

Project Outcome

- Running Summary Generator code results in a streamlined generation of a series of text documents containing in depth descriptions of each scholarly journal.
- Running Power Point Generator Code results in the summary files being collected into a clean and concise manner for ease of viewing. The user is freely able to edit the text and add figures or clarifications when needed.
- This tool allows researchers to quickly generate draft PowerPoints that go over a large number of journals.

Integration of Generative AI and Automation in the Literature Review Pipeline (Pages 4-6)

The growing scale and complexity of scientific research make it increasingly difficult for scholars to manually conduct comprehensive literature reviews, particularly in fields such as neuroscience that rely on multimodal data like EEG and fNIRS. This project sought to address that burden by building a fully automated assistant that reads, summarizes, analyzes, and visualizes scholarly articles using generative AI. At its core, the system transforms dense PDF content into structured, editable formats while reducing the need for repetitive human input. By leveraging OpenAI's language models, automated file handling, and dynamic PowerPoint generation, this project creates a reproducible, scalable pipeline from document ingestion to hypothesis formation.

This section outlines five major technical components of the system: (1) the standardization of scholarly article summaries via prompt engineering, (2) the extraction of future work sections to generate testable hypotheses, (3) the implementation of automated API calls with OpenAI's LLMs, and (4) the conversion of PDF files through intermediate formats to structured presentations and finally (5) the use of PowerPoint over Google Slides,. Each component contributes to the end-to-end automation and adaptability of the tool, ultimately enabling researchers to generate ready-to-use insights from unstructured literature with minimal effort.

A major strength of the system lies in its ability to produce standardized, structured summaries of scholarly articles. The vast majority of EEG/fNIRS research papers follow the same outline of Introduction, Background, Recording Methods, Experimental Design, Results, Conclusion, and Future Work. From these sections, information related to testing, replicating, or advancing the field can be found and extracted. This is achieved through carefully designed prompts provided to the OpenAI language model, which instructs the assistant to extract specific information categories such as metadata, experimental design, brain regions of interest, key findings, and proposed future work. These categories were formulated over months of manually crafting literature reviews by hand and were found to be generally applicable to EEG/fNIRS studies. Because these prompts are consistent across all documents, the automated literature reviewer produces summaries that maintain the same format as the manual literature reviews. This uniformity enables easier downstream parsing, comparison between studies, and conversion to presentation-ready formats. By limiting variance in the output, the system minimizes interpretation errors and helps researchers focus on content rather than formatting. This limiting of variance plays

a key role downstream, therefore a temperature of zero was chosen for the OPENAI prompt to minimize the internal excitability of the LLM.

The assistant also supports early-stage hypothesis generation by synthesizing future research directions proposed in the reviewed papers. Each summary includes a section on "future work," extracted by identifying language cues commonly used by authors when discussing study limitations or next steps. These individual insights are compiled and analyzed collectively in a meta-summary phase, which identifies overlapping gaps or suggested avenues for future research. The assistant then uses these patterns to formulate a testable hypothesis and a draft experimental protocol. This includes detailed suggestions for tasks, modalities (EEG or fNIRS), brain regions of interest, statistical analysis, and sample size. This capability is especially useful for researchers developing new studies, offering them a concrete starting point grounded in published literature. Due to the non-deterministic nature of the LLM, by repeatedly running the hypothesis generator with slightly different prompts (ie. Generate the 2nd most likely hypothesis, generate 5 hypotheses, using a non-zero temperature) a researcher is able to produce multiple initial hypothesis and experimental protocols with minimal effort. While these experiments may be similar, this similarity acts as a benefit as the researcher can quickly prototype the experiments and generate a series of pilot studies.

Behind the scenes, all of this functionality is powered by automated calls to OpenAI's large language models. Rather than requiring user interaction with a chatbot interface, the system submits prompts and text blocks directly to the OpenAI API, handling everything from summarization to hypothesis generation. This standardized approach prevents user error as the only point of interaction is to select PDF files; all subsequent steps occur in the background. This architecture is made possible through secure API key management, allowing the assistant to run repeatedly and safely across multiple systems. The use of gpt-4o ensures high-quality, high-speed processing, enabling large-scale reviews to be conducted in a matter of minutes. In fact, the crucial speed limit is the built-in tokens-per-minute cost of using LLMs, in order to keep the system from accidentally generating unexpected charges, a rate limit of ~3000 tokens per minute was implemented, with built in pause functions in case the program requires extra resources.

To move data efficiently between components, the project also employs a multi-format transformation pipeline. PDF files are first parsed using the fitz library, which extracts the text

content while ignoring visual elements like figures and tables. The extracted text is summarized and saved as .txt files, which are then parsed and reformatted into structured .json dictionaries. Ensuring that the LLM did not intentionally generate corrupted .json files was a hurdle that had to be overcome. To account for this, a failsafe 'try;else' step managed to prevent the code from simply crashing, however a more permanent fix was later implemented. This fix built upon the 'try:else' code and sought to strip extraneous ' "" ' markers from the .json file. These dictionaries serve as the input for the PowerPoint generation module, where each field is mapped to predefined slide sections. This stepwise conversion—from PDF to .txt, then to .json, and finally to .pptx—ensures modularity, ease of debugging, and adaptability to future formats or platforms.

One of the final design choices in this project was the use of python-pptx to generate PowerPoint presentations instead of relying on Google Slides. This decision was motivated by several practical benefits. PowerPoint files are platform-independent, can be accessed offline, and are widely accepted in academic conferences, classroom presentations, and lab environments. Unlike Google Slides, which requires authentication and internet access for API usage, PowerPoint generation can be executed locally with full formatting control. This local generation mitigates data privacy concerns, making it a safer choice for working with unpublished or sensitive research data. Additionally, with the only authentication requirement being a working OPENAI_API_KEY, this code is easily distributed and saves all files to the same directory. If the user still requires a Google Slide to be generated, a manual upload can accommodate them.

In conclusion, this system demonstrates how generative AI and software automation streamline the literature review processes, turning it into a scalable and modular system. Running this code on either the entire dataset of >50 articles, or on subsets of only 5 articles, the researcher now has the time and resources to quickly review their field of study, develop a working understanding, and generate a novel hypothesis in only a few minutes. By combining structured prompt engineering, robust API integration, and file format handling, the assistant enables researchers to go from reading papers to designing experiments with unprecedented efficiency. While initially designed with EEG/fNIRS analysis in mind, this same protocol can be quickly adapted to other research domains. With further development, this system could serve as the foundation for lab-wide review tools, grant-writing assistants, or educational platforms in neuroscience and beyond.