James Nielsen

CMSC 35360: Autonomous Laboratories

Final Report: Optimizing User Interface Research with LLMs

Research Goal

The main question I wanted to investigate with the use of Large Language Models was: In the realm of Human Computer Interaction, how can we optimize spacing and positioning of User Interface elements in order to maximize customer satisfaction and spending.

My primary motives for choosing this research question are in part because of my own interest in Human Computer Interaction and interface design; ideally, by going through the multitude of steps in the research process I will pick up on insightful strategies and tools for my own work – given the highly variable nature of the field of HCI, additional considerations are always valuable. Another reason why this research question felt worth exploring was because it would be particularly interesting to investigate how a Large Language Model processes and considers research that is largely psychological in nature. Because so much HCI research is conducted on human participants, whatever LLM is used in this research process will have to consider the most optimal ways to influence the human psyche, something I am personally curious if it can be effective at.

As such, I began the steps of the Automated Discovery Pipeline.

Part 1: Literature Retrieval and Processing

For this part of the research process, I followed the README from the CMSC35360 github quite closely. I started off with generating my list of keywords utilizing ChatGPT 4 using

the recommended prompt, altered slightly to suit my research area. My exact prompt was as follows:

"I want to retrieve documents from Semantic Scholar that relate to Optimal Spacing/Position of UI Elements to best facilitate user experience and Encourage Purchases. Please suggest to me 100 sets of possible keywords. Output each set of keywords on a separate line, without any numbers."

The exact list of the 100 sets of keywords can be read in the attached file opt.txt: https://github.com/JamyVIII/auto_lab_report/blob/main/opt.txt

I then used the provided bulk_fetch_abstracts_from_SS_via_keyword.py file to retrieve relevant articles from Semantic Scholar from 2016 to early May 2024, when I first collected my articles. After eliminating duplicates, I was left with 748 articles having been collected, as can be seen in opt.jsonl: https://github.com/JamyVIII/auto_lab_report/blob/main/opt.jsonl

Of note, I had to use Windows Subsystem for Linux in order to make use of GNU parallel, and I continued to use Windows Subsystem for Linux for the rest of the research pipeline.

From there, I ran the llm_check_relevance.py file to leverage our LLama 3 server to determine the most relevant and useful articles to eventually synthesize. Naturally, I altered the create_prompt function's gpt_user_prompt string to be asking whether the article abstracts relate not to finding a new and improved catalyst for the conversion of CO2 to ethylene, but instead about optimizing user interfaces to incentivize user spending. Unfortunately, while trying to update my CMSC35360 github with Professor Foster's new files/code, I accidentally

permanently deleted my altered string that I used in the relevance check, so I cannot give the exact wording with which I altered the original string.

The relevance check ultimately resulted in 15 articles being rated a full 4, 221 articles a 3, 191 articles a 2, and the other 321 articles a 1 or 0. The description of each article's abstract generated by the LLM can be seen in the resulting file opt_scores.csv https://github.com/JamyVIII/auto_lab_report/blob/main/opt_scores.csv

I opted to not use any of the papers rated 0 or 1; this left me with 427 relevant papers on the subject from the initial count of 748.

Finally, with the 427 papers from Semantic Scholar identified, I requested an API key from Semantic Scholar, and ran the retrieve_documents_from_SS,py. Notably, I am uncertain if this has been changed already, but in the version of retrieve_documents_from_SS.py that I ran, the paper_ids were not getting parsed correctly from the csv file. Instead of taking just the paper_id, it would read in a full line, so every single paper would result in the “Failed to download paper” error exception from the download_paper function. So, I altered line 114 from the main function to instead do:

paper_ids = [id.split(',')[0].strip() for id in paper_ids]

This would properly break at the first comma of each line. With this change, I was able to properly start downloading PDFs.

Ultimately, after execution, 110 articles out of 427 were successfully downloaded, a success rate of around 25.8%. This was a pleasant surprise, as I expected only about 50 papers to get through.

Part 2: Information Extraction with LLMs

With the 110 article pdfs downloaded, I now tried to convert the pdfs to txt. I initially attempted to use the recommended extract_txt_from_pdf.py file that utilized the txtai library to convert the pdfs, but I was encountering a strange bug. I noticed several others from our class had trouble with this step, and the error was very challenging to get to the bottom of, having to do with trouble encoding utf-8 to handle Unicode characters correctly. I believe the bug may have been because the highest version of Python that Windows Subsystem for Linux can easily download is version 3.8, but some of the libraries used in extract_txt_from_pdf.py need Python version 3.11; however this is only a theory, I was not able to fix this file myself.

To resolve the issue, I instead followed the recommendation of fellow classmate Sep, who wrote an alternative file that utilized the PyMuPDF library instead of txtai. This worked perfectly, and thus I was able to convert all of my scholarly articles to .txt files.

From there, I used the llm_summarize_all.py from the github to generate a summary for each paper utilizing our LLama 3 server. Then, I used the compact_summary_files.sh and generate_json_file.py to compact the summary files and arrange them all in the all.jsonl file: https://github.com/JamyVIII/auto_lab_report/blob/main/opt.jsonl
For some reason, during this step, 2 of the papers erred when generating summaries, leaving me with a total of 108 summaries.

I then used fellow classmate Joshua's version of extract_hypotheses_from_llm_output,py to generate a file listing the id's, hypotheses, and experiment descriptions of each article. I later adjusted my extract_hypotheses to not bother saving the id of each paper to save context window space, and saved that in a file called hypotheses_experiment.jsonl.
https://github.com/JamyVIII/auto_lab_report/blob/main/hypotheses__experiment.jsonl

For the purposes of next steps, I also generated an alternate version of that jsonl file with only the hypothesis of each paper, and saved that in a file called hypotheses.jsonl. https://github.com/JamyVIII/auto_lab_report/blob/main/hypotheses.jsonl

After looking through the hypotheses manually, though, I noticed that unfortunately, despite being supposedly analyzed by Llama 3, a large amount of the hypotheses were completely irrelevant to Human Computer Interaction, discussing user experiences with things like medication and treatments. There were also duplicates, which must have been due to different publications posting the same article.

Part 3: Hypothesis Generation and Extension

My first idea at generating the most useful hypothesis was to take my hypotheses_experiment,jsonl file and feed it to a Large Language Model so that it can synthesize all 108 hypotheses, then generate its own hypothesis as to the best ways to optimize user interfaces for commercial sales.

Around the time I was undertaking this step, ChatGPT's newest language model was released, ChatGPT-4o, and from what I had heard, the general consensus is that it was both extremely fast and extremely effective. So, instead of leveraging our Llama 3 servers for this hypothesis synthesis, I opted to buy a subscription to ChatGPT in order to experiment with the supposedly newest most powerful Large Language Model.

Thanks to the file attachment feature offered by ChatGPT-4o, I was able to directly feed my hypotheses_experiment.jsonl file. Thankfully, with an impressive 128k context window, I did not need to worry about the context window for this task. Pasted below is the exact prompt I initially provided ChatGPT-4o:

- "You are a super smart AI that knows about science. You follow directions and you are always truthful and concise in your responses.

  Here is a research question: In the realm of Human Computer Interaction, how can we optimize spacing and positioning of User Interface elements in order to encourage purchases?

  Attached is a file called hypotheses_experiment.jsonl. It is a compiled list of research paper hypotheses that have investigated the topic of optimizing user interfaces for digital consumers.

  Please synthesize the 108 hypotheses within hypotheses_experiment.jsonl, while taking note of the experimental procedures utilized in each study, to inform yourself on the subject of optimizing user interface elements for maximizing consumer spending. Then, to the best of your ability, generate a new hypothesis or conjecture about the best strategy for optimizing user interfaces in order to maximize consumer purchases."

From there, ChatGPT-4o proceeded to give a summary of the key insights it gleaned from the 108 papers. This summary was quite in-depth, but in brief, the key points were about the importance of Visual Design and Layout, Personalization and Customization, Attention and Cognitive Load, Emotional and Psychological Factors, and Interactivity and Feedback.

It then gave an original Proposed Hypothesis, which was:

"Implementing a dynamically adaptive user interface that combines personalized recommendations, real-time feedback, and simplified navigation will significantly increase consumer purchase rates on e-commerce platforms."

After this, without me even prompting the model, it proceeded to give a somewhat brief experimental procedure about how to test this hypothesis, and how to conduct Data Analysis, which I will describe and discuss in more detail in the upcoming sections of this report.

Overall, this first hypothesis was certainly built with good ideas, however, it seemed a little bit generic, and it was difficult to determine whether genuine synthesis of the research papers took place, or if ChatGPT-4o used its previously trained knowledge to give a general hypothesis that is relevant to the research question. Furthermore, I was slightly bothered that it gave experimental procedures without being prompted.

As a result, in order to improve the hypothesis generation, I made two significant changes. As mentioned in the previous section, I generated a new jsonl called hypothesis,jsonl that did not include the experiment description, so that the model would only work with the 108 hypotheses.

The next change was to utilize a Chain of Thought approach in order to affirm that the model is properly synthesizing the content of the 108 research papers, and not just its pretrained knowledge. To do this, I made a new chat, and asked the model to read and rate the 108 hypotheses on novelty and whether the hypothesis is likely to be true, on a scale of 0-10.

At this point, I encountered a bit of a hiccup. After looking through the hypotheses manually, I noticed that despite being supposedly analyzed by Llama 3, some of the hypotheses were completely irrelevant to Human Computer Interaction, instead discussing user experiences and customer satisfaction with topics like medication and treatments. There were also duplicates, which must have been due to different publications posting the same article.

To rectify this, I initially requested the model to both remove duplicates, and also remove anything that does not seem relevant to Human Computer Interaction, however, that request

ended up removing an astounding 84 hypotheses, and after cross-referencing, it removed ones that I would personally reckon are still rather valuable. So, I settled on just removing duplicates, which was only 4, and figured the model could sift through less relevant hypotheses, then sorted the remaining 104 hypotheses in a file called sorted_hypothses,jsonl. https://github.com/JamyVIII/auto_lab_report/blob/main/sorted_hypotheses.jsonl

Finally, I began the process of synthesizing a hypothesis using a chain of thought. The first prompt was similar to the initial one described earlier, but now just asked to rate the hypotheses, and select the best 5 and their scores, which are shown below:

1. Hypothesis: The integration of industrial design principles with user interface elements can enhance user engagement and purchasing behavior.
    a. Relevance: 9
    b. Likelihood: 9
    c. Novelty: 9
    d. Total Score: 27
2. Hypothesis: Personalized service and better display of product information can lead to increased user satisfaction and higher purchase rates.
    a. Relevance: 9
    b. Likelihood: 9
    c. Novelty: 9
    d. Total Score: 27
3. Hypothesis: A patient-centered, multifaceted intervention in health-related apps improves user compliance and overall satisfaction, potentially increasing in-app purchases.

   a. Relevance: 8
   b. Likelihood: 9
   c. Novelty: 9
   d. Total Score: 26

4. Hypothesis: Personalized and targeted marketing strategies through UI design can significantly boost user engagement and conversion rates.
   a. Relevance: 8
   b. Likelihood: 9
   c. Novelty: 9
   d. Total Score: 26
5. Hypothesis: Sustainable web design can positively influence user perception and purchasing behavior, promoting a more eco-friendly shopping experience.
   a. Relevance: 8
   b. Likelihood: 9
   c. Novelty: 9
   d. Total Score: 26

Overall, based on my own personal interpretation, the hypotheses listed are decent at addressing the research question, but fall on the side of being a little generic. Interestingly, ChatGPT-4o seemed highly resistant to giving out any 10s, which somewhat calls into question the validity of the scoring system. However, for my purposes, the importance of this stop is not the accuracy of scoring the top 5 hypotheses, but priming the LLM so that it can better create its own hypothesis in the next prompt.

Finally, in the same chat, I once again asked the model to generate a potential hypothesis based on the 104 provided hypotheses. The result of this prompt was the hypothesis and the rationale behind it, as such:

- Hypothesis: Integrating adaptive user interface elements that dynamically adjust based on real-time user behavior and preferences, coupled with personalized product recommendations and streamlined checkout processes, will significantly enhance consumer experience and increase overall spending in digital marketplaces.
- Rationale: This hypothesis is derived from key themes found in the analyzed hypotheses, emphasizing the importance of personalization, real-time adjustments, and ease of use. By leveraging adaptive elements, the user interface can provide a more tailored and efficient shopping experience, which is likely to lead to higher satisfaction and increased purchases.

It seems that even after the extra steps, the generated hypothesis was extremely similar to the initially generated hypothesis. This could be an indicator that the initial generation was already the best synthesize possible, or more negatively, it could be an indicator that the model is just using its pretrained knowledge of HCI to create a general hypothesis, and not relying on the provided article data. To rectify this, my final hypothesis generation experiment was to utilize a Tree of Thought approach.

Within the same chat, I asked ChatGPT-4o to generate 4 more different possible hypotheses, then, to act as a council of 4 HCI experts, 2 against each hypothesis, 2 for each hypothesis, and have the council discuss and select the best of its 5 generated hypotheses.

The full output is too long to paste in full, but can be seen in tree_of_thought.md.

https://github.com/JamyVIII/auto_lab_report/blob/main/tree_of_thought.md

For some reason, each expert speaking was considered to be a new expert. In the end, the council chose Hypothesis 1, the first one generated by the Chain of Thought approach.

Part 4: Experimental Plan Generation

Having learned from the previous section, I knew I needed to do some cleanup of hypotheses_experiment.jsonl before generating experimental plans. So, after again removing the 4 duplicates, reformatting, and sorting the data, I saved the hypotheses and experimental plan data in sorted_hypotheses_experiment.jsonl.

https://github.com/JamyVIII/auto_lab_report/blob/main/sorted_hypotheses_experiment.jsonl

In a prompt mirroring the initial prompt for hypothesis generation, I provided sorted_hypotheses_experiment.jsonl, and now the ideal hypothesis, and asked ChatGPT-4o to create a specific experimental plan.

The resulting experimental plan is too long to paste in this document, but can be read in plan1.md.

https://github.com/JamyVIII/auto_lab_report/blob/main/plan1.md

Overall, it is already impressively specific in the experimental plan, mentioning details such as the number of participants, the experimental and control group, participant randomization, and particular quantitative and qualitative evaluation metrics.

Part 5: Experiment Execution Plan

In the same chat window as Part 4, I then utilized a Chain of Thought approach to try and improve the initial experimental plan outlined in plan1.md, so as to perfect an experiment execution plan.

My strategy was to request the model to imagine that a robot is going to perform each step of this study and run it tomorrow. The robot needs specific steps on how to conduct each part of the experimental plan, an estimation about the budget required for each step, and the estimation as to the time frame that each step of the plan will take.

The result of this improved experiment execution plan is once again too long for this document, but can be read in full in plan2.md.

https://github.com/JamyVIII/auto_lab_report/blob/main/plan2.md

Overall, this experimental execution plan is extremely specific, roughly 3 times the number of lines as plan1.md, and I believe is a rather effective plan. If I had the necessary time and money, I could see myself running an experiment akin to the one described in these steps.

Part 6: Data Analysis and Hypothesis Testing

Continuing the same Chain of Thought approach, for this part I utilized the same chat window as parts 4 and 5.

This time, I asked for specific steps the robot should take to analyze the data collected from participants, and come to a conclusion about whether to reject the null hypothesis or not.

The output of this prompt was an extremely specific set of steps that the robot can take in order to determine whether or not to reject the null hypothesis. It is, as usual, too long for this document, but can be read in full in analysis.md

https://github.com/JamyVIII/auto_lab_report/blob/main/analysis.md

As with part 5, I am impressed with the steps outlined here by ChatGPT-4o, and believe the process here would be sufficient to analyze the results of the hypothetical experiment. From providing specific Null and Alternative Hypotheses, to measuring Effect Size, to performing Chi-Square tests, the Data Analysis is very rigorous and suitable for an HCI study.

Reflection on Automated Discovery via LLMs

In the end, throughout the process of this automated pipeline, numerous strengths and weaknesses of Automated Discovery via LLMs have been made clear.

To start off with strengths, LLMs are extremely effective at providing near instant synthesis and generation of whatever material is desired. It is incredibly effective at producing a vast quantity of ideas, from which we can synthesize ourselves and iterate on. If that is too much work, it can even perform its own somewhat shaky synthesis, and arrive at what is likely a fairly good answer, as shown by my experimentation with Tree of Thought, or more importantly the ability to gather 108 papers on a topic in a relatively fast amount of time. In general, proper prompting of LLMs using strategies like Chain of Thought and Tree of Thought can result in highly synthesized and thought out responses.

Moreover, p[ossibly one of the greatest benefits of the utilization of LLMs for research is the capability to restructure code and documents that humans can easily utilize. While I can personally sort something like the hypotheses.jsonl manually, or even write code to sort it for me, the time it takes me to do so is significantly more burdensome than leveraging an LLM.

However, LLMs are clearly not perfect. As shown by steps in this research pipeline like my manual analysis of my hypotheses, LLM synthesize can completely misread or misinterpret

the relevance of certain documents or ideas. For instance, if it happens to find the right keywords, it can completely misjudge the value of a particular paper. Even when synthesis appears successful, it can be hard to determine if the model actually engaged in synthesis of the provided material, or just relied on its pretrained knowledge.

Furthermore, the most scary aspect of relying on an LLM is its capacity to hallucinate. Because it is always attempting to satisfy the user, it can easily make up data or conclusions that are not correct if it is otherwise unsure of an answer. LLMs will not be clear on what they do not know: anything unknown oftentimes will be filled in with a best guess.

Ultimately, as of the time being, fully automating a discovery process with LLMs seems extremely challenging. Despite their many strengths, there is clear capacity for mistakes or failure, so for the time being, human oversight seems necessary. Yet, I believe that in the years to come, it is inevitable that LLMs continue to prove more and more useful to all things, discovery included. I am excited to see what the future holds for automated discovery.