Peptide Digest

Elizabeth Gilson - UC Berkeley

Collaborators

Jeffrey Jacob - UC Berkeley

Joshua Daniel Blomgren - UC Berkeley

Dr. Gregory Bryman - Merck

Dr. Ge Wendong - Merck

Dr. Chen Tianchi - Merck

Dr. Jennifer Johnston - Merck

Dr. Jessica Nash - UC Berkeley, Molecular Sciences Software Institute

Abstract

Project Summary

The "Peptide Digest" capstone project is a collaborative effort between UC Berkeley and Merck aimed at revolutionizing the analysis of scientific papers, with a particular focus on computational peptide research (Merck 2024). The project brings together experts in machine learning, computational chemistry, and data science to develop a Large Language Model (LLM) that can efficiently extract metadata from scientific publications, assign priority scores to papers based on their relevance, and provide a user-friendly interface for accessing the model (Vaswani, 2017). There is a significant time investment required for researchers to sift through large volumes of scientific literature. To address this challenge, the team proposes the development of an LLM trained on a diverse dataset of scientific papers. The LLM will be trained to identify key metadata such as the number of compounds, series diversity, presence of unnatural amino acids, and cyclicity in computational peptide research papers. On top of this, it will provide a short summary of the article and relevance score.

Project Deliverables

The major project deliverables include:

LLM for Metadata Extraction: The primary deliverable is the development of an LLM capable of accurately extracting relevant metadata from scientific papers. The LLM will use machine learning techniques to analyze textual data and identify key information pertinent to computational peptide research. This includes parameters such as compound numbers, series diversity, and other essential features relevant to peptide modeling.

Priority Scoring System: In addition to metadata extraction, the LLM will assign priority scores to papers based on their relevance to specific inputs provided by Merck researchers. This scoring system will help researchers prioritize papers for further analysis, saving time and resources.

User-Friendly Interface: To facilitate easy access to the LLM, the project will develop a user-friendly front-end interface in the form of a web-app. This interface will allow researchers to input their criteria and retrieve prioritized papers efficiently. The interface will be designed with usability and accessibility in mind to cater to a wide range of users within Merck.

Web-Scraping Tool: The project also aims to automate the process of finding and analyzing new relevant articles through a web-scraping tool. This tool will continuously search for new publications and provide summaries and relevant information to researchers, keeping them updated with the latest developments in computational peptide research.

The project timeline spanned sixteen weeks, with clear milestones and objectives for each phase of development. Collaboration with Merck's expert scientists, access to Perlmutter, a supercomputer at the National Energy Research Supercomputing Center in Berkeley, and mentorship from experienced researchers ensured the project's success and relevance to real-world applications. In conclusion, the "Peptide Digest" capstone project represents a cutting-edge endeavor to streamline scientific paper analysis using advanced machine learning techniques. By delivering an efficient LLM and accompanying tools, the project enhances productivity, accelerates research processes, and contributes to advancements in computational peptide research at Merck.

Gap Analysis

The initial project description and the final summary exhibit several notable changes and refinements.

Scope Clarification:

Initial: Our initial project description mentions a focus on computational peptide research and the intention to potentially expand into other fields.

Final: The final summary explicitly focuses on computational peptide research. This shows that we refined and clarified the scope. This was mostly based on the feasibility of fine-tuning for one specific field vs many. Although the well-documented code means that with more data, the model could easily be expanded to a wider variety of fields.

Project Deliverables:

Initial: The initial description outlines the creation of a Large Language Model (LLM), a front-end user interface (UI), and a web-scraping tool.

Final: The final summary provides more detailed descriptions of the deliverables, emphasizing the LLM's capabilities for metadata extraction and priority scoring, as well as the development of a user-friendly interface and a web-scraping tool. This indicates a more structured and detailed approach to deliverable development.

LLM:

Initial: We were planning on using Llama-2 as our base model (Touvron, 2023).

Final: We decided on using Gemma, as it came out part-way into our project and showed better performance and had a larger maximum token length (Gemma Team, 2024).

Collaboration and Resources:

Initial: The initial description mentions the use of UC Berkeley computing resources, specifically Savio and NERSC.

Final: The final summary specifies the use of NERSC's Perlmutter supercomputer at the Lawrence Berkeley National Lab (NERSC, 2024).

Overall, this final project summary reflects a more refined and detailed understanding of project objectives, deliverables, and execution strategies compared to the initial abstract. The changes suggest a progression from conceptualization to a more concrete and well-defined final project execution.

Methodology

LLM Background

The GEMMA 7B model is part of the Gemma family of LLMs, which are publicly available models released by Google. Gemma-7B is a neutral network with seven billion parameters. This size allows it to capture a high level of detail with text data. The 7B model is optimized for deployment on Graphic Processing Units (GPUs) and Tensor Processing Units (TPUs), compared to the 2B model which is designed for Central Processing Units (CPUs). Past research has shown that Gemma-7B demonstrates strong performance across many benchmarks and excels in language understanding and reasoning (Gemma Team, 2024). We decided to use the GEMMA 7B model because of its high performance, large context window size, and because it is open source.

Fine-tuning

We first ensured the availability of a GPU and installed necessary Python packages as prerequisites for our experiments. The Gemma model was then loaded with a configuration optimized for memory usage through Quantization-aware Learning Rate Adaptation (QLoRA) (Dettmers, 2023). Subsequently, we developed a function to generate code completions based on user prompts using the Gemma model, with particular attention paid to prompt engineering to enhance result quality. We created a dataset, with human created metadata and formatted it accordingly into the requisite Gemma instruction format. Additionally, we applied Low-Rank Adapters (LoRA) to the model using the Performance Enhancement for Training (PEFT) library (Sourab, 2022). The fine-tuning process was then initiated with predefined training arguments, incorporating qLora and the Supervised Fine-Tuning Trainer (SFTTrainer) from the trl library (Leandro, 2020).

Front-end

The front-end component of our project serves as the user-facing interface, providing access to the functionality and insights generated by the Large Language Model (LLM). Our front-end is designed as a web application, leveraging the Dash framework for its development. This choice allows for the creation of interactive and visually appealing interfaces that users can easily navigate. One key aspect of our front-end architecture is its integration with a database of pre-computed entries. This approach minimizes the need to call the LLM every time a user inputs an article. Instead, the front-end retrieves relevant data from the pre-computed entries stored in the database, optimizing performance and user experience. To facilitate seamless communication between the front-end and back-end, we utilize FastAPI for our API implementation (Ramírez, 2023). FastAPI offers efficient request handling and enables rapid development of RESTful APIs, ensuring smooth data exchange between the user interface and the underlying computational components (Song, 2023).

Additionally, we use Docker for containerization to streamline deployment and scalability of our front-end application (Docker, 2024). Containerization allows for the packaging of our application and its dependencies into portable, isolated containers, simplifying deployment across different environments and enhancing reproducibility. By incorporating these technologies and practices into our front-end development, we aim to deliver a robust, user-friendly interface that effectively harnesses the capabilities of our LLM while ensuring scalability and maintainability.

Validation

Survey Results

Some articles already provide short summaries, however they are often lacking in the detail readers may want. To see if these author summaries were more or less useful than the model summaries, we surveyed 14 individuals with backgrounds in science. They were asked to rate summaries created by both our model and a human author on a scale from one to five. The results from our initial round of surveying reveal an impressive achievement: our model surpassed the Human Summary for every analyzed article. This noteworthy accomplishment is further reinforced by a low p-value, indicating statistical significance, and passing the 95% confidence interval. Our meticulous documentation of the project's progress and outcomes can be found in the comprehensive notebook, ensuring transparency and replicability of our methods. Moreover, we advocate for openness and collaboration by encouraging the accessibility of all surveys conducted throughout the project. These findings underscore the effectiveness and reliability of our model while emphasizing our commitment to rigorous scientific inquiry and open science principles.

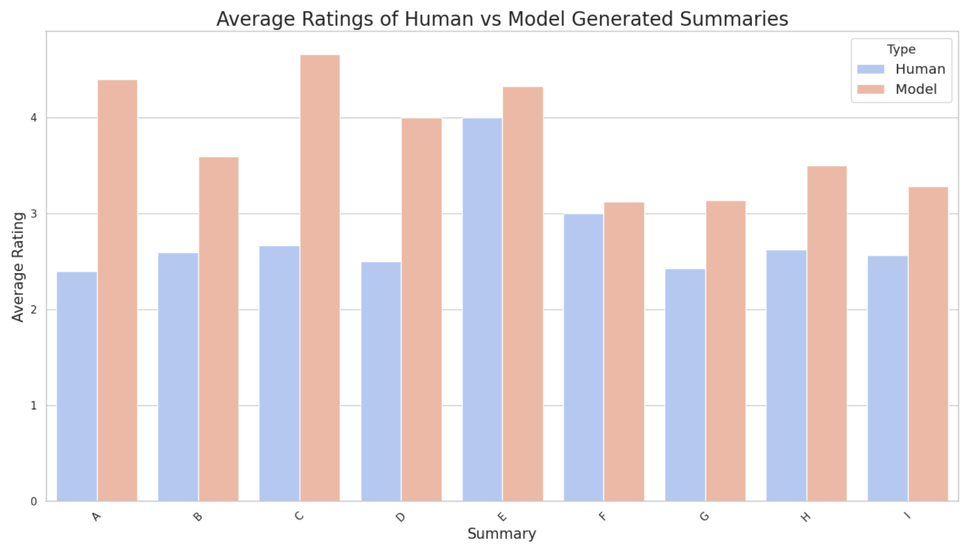

Figure 1. Average Rating of Human vs Model Generate Summaries

Figure 1. illustrates the comparison between human-generated and model-generated summaries based on their average ratings. The x-axis represents different summaries, while the y-axis displays the corresponding average rating. Every data point on the graph represents the average rating assigned to a specific summary, providing a clear visualization of the performance of both human and model-generated summaries. This comparison offers insights into the effectiveness of the model in generating summaries that are rated similarly to those produced by humans, highlighting the model's ability to capture and convey key information effectively.

Fine-tuning

We have also accomplished fine-tuning on the model to collect the metadata. The fine-tuning is based on 15 articles that have been individually analyzed and the metadata has been extracted. The process of fine tuning involves: modifying the data to fit the required Gemma instruction format, tokenizing the data, organizing it into batches, apply Quantized Low-Rank Adapters (QLoRA) using the PEFT (Parameter-Efficient Fine-Tuning) library, then use the Supervised Fine-tuning trainer (SFTTrainer) to train the model (Dettmers, 2023, Sourab, 2022, Leandro, 2020) . Then the loss is measured using the cross-entropy loss. Below is a graph of our Training Loss vs Epochs.

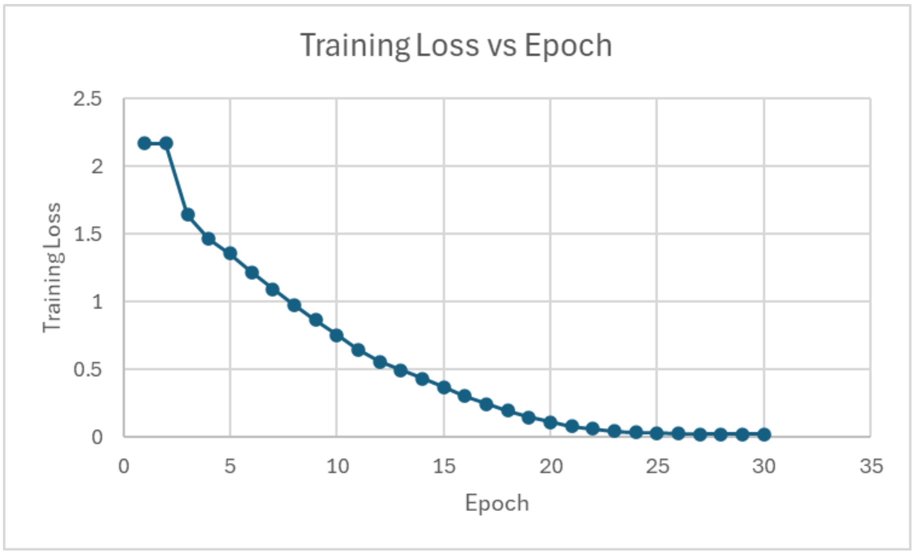

Figure 2. Training Loss Curve for Fine-tuning

As you can see after thirty epochs, the loss reaches an equilibrium and no longer significantly decreases.

To measure the performance of the fine-tuned model vs the original model we used two separate metrics. One is the “Correctness Score”, which is the number of results that the model produces that are in the ground truth, also known as true positives. Another metric is the "Hallucination Score" which is how many times the model identifies a piece of data not in the ground truth, or false positives. Below you can see the results of these two metrics before and after fine-tuning.

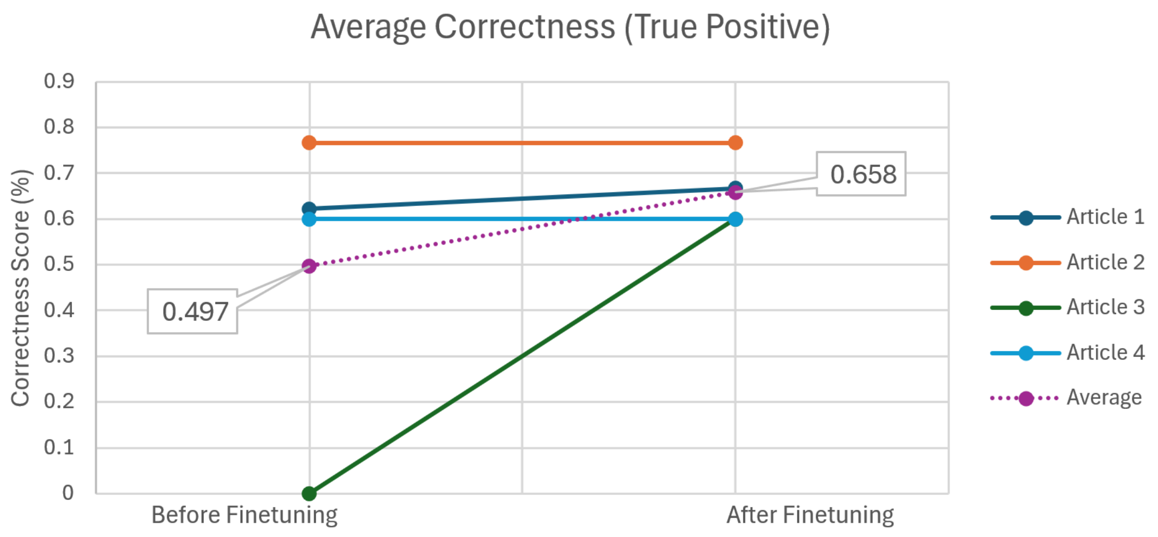

Figure 3. Average Correctness Before and After Fine-tuning

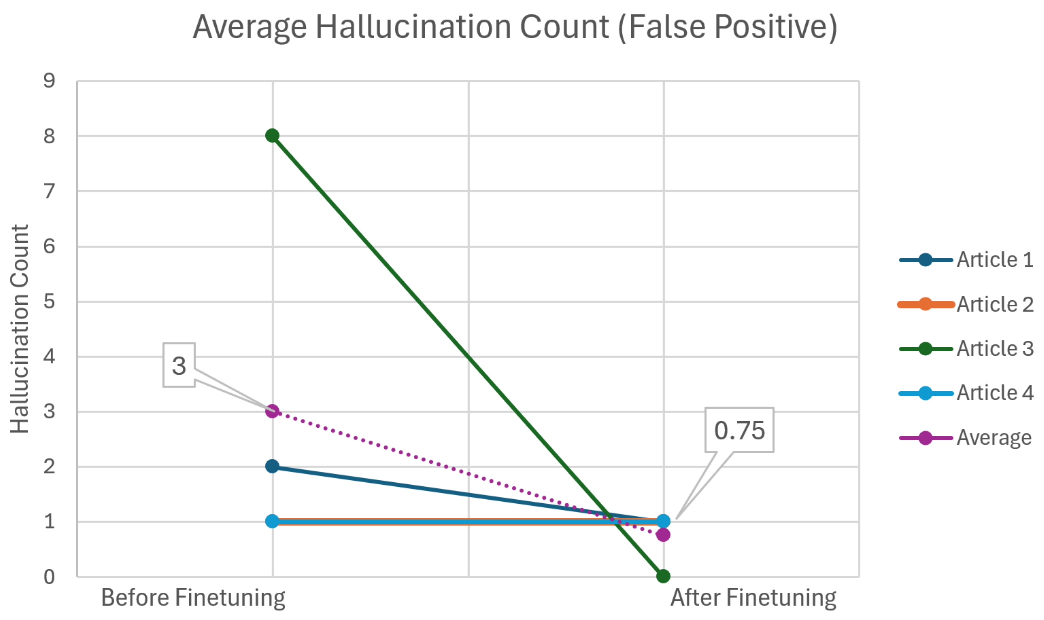

Figure 4. Average Number of Hallucinations Before and After Fine-tuning

From these graphs, you can see that for some articles there was no difference in performance after fine tuning, but for some articles there was a drastic difference. After fine-tuning the model gave more correct results and decreased the average number of hallucinations. This was just with a small dataset of fifteen articles, so these results are very promising.

Future of Software Project

Our end goal is to make our code open-source and have a github repository where anyone can access it. Additionally we are going to host the fine-tuned LLM on HuggingFace so the model can be directly connected. Merck has GPUs, so this way they could directly download it and run it from their machine. Furthermore, we are committed to documenting our project comprehensively and aim to complete a paper detailing our work on this project. We intend to submit a paper about this project for publication, thereby contributing to the academic discourse in our field and disseminating our findings to a broader audience.

Capstone Project Self-Reflection

Project Management

We used Notion to keep track of our notes, important dates, and progress throughout the project. This was useful, especially for a team project because we all had access to everything. Additionally it allowed us to look at the long-term scope of the project, but also we able to see weekly tasks, so we could ensure we weren't falling behind or committing too much time to a task that was not central to our main deliverables.

Our communication was also very solid throughout the semester, with weekly meetings with our MSSE mentor, bi-weekly meetings with the Merck Team, and weekly/more than weekly meetings with just our team. This allowed us to have frequent communication and get feedback on all of the progress that we have been making. Our team was pretty cross-functional, in the sense that we were rarely working on overlapping tasks, however it was still essential to update everyone because occasionally one change would impact the work of others.

Best Software Engineering Practices

We had a clear proposal in the beginning of the project, but were also flexible as we tried new things and faced unexpected challenges. Overall, this method allowed us to iteratively develop the project so we could be successful but also provide an end product that had value. We have also tried to make our code as modular as possible, so it can be easily adapted in the future. For example, our web-app can fetch articles in multiple different ways (PubMed ID, URL, DOI, etc.). This adaptability means that our code can hopefully be re-used for a variety of different tasks.

Our code is well-documented and parts of it are organized into different python packages for easy re-use. Additionally, our github will be public by the end of the project, meaning that all of our code will be open-source. This allows others to re-use it, which encourages reproducibility and transparency, which are important things in any science or engineering field.

Skills from MSSE

Note: The content in this section is re-used from Assignment 2

Python Skills from Chem 274A: Throughout Chem 274A, I acquired a strong foundation in Python programming, which will be invaluable in developing the various components of the project, such as data retrieval, data cleaning, working with the LLM, and potentially the web-scraping tool. Python's versatility and extensive libraries make it an ideal choice for implementing machine learning algorithms and data analysis tasks. Additionally, as a reach goal our team hopes to implement our own Python package, so the Python skills from this class are very essential for many aspects of this project.

Data Cleaning and Analysis Tools from Data 200: In Data 200, I learned essential skills for data cleaning and analysis, which will be crucial in preprocessing the scientific papers' textual data before feeding it into the machine learning model. Effective data cleaning ensures that the model receives high-quality input, leading to more accurate results and insights. Additionally, I was able to perform some EDA (Exploratory Data Analysis) and show the results to our mentors at Merck to confirm that the data we are collecting matches their expectations.

Best Software Engineering Practices from Chem 274B: Chem 274B provided me with insights into best software engineering practices, including code organization, version control, and documentation. I will apply these principles throughout the project to ensure that our code is well-structured, maintainable, and easily understandable by collaborators. If we do decide to publish our own Python package, these skills with be even more essential as future developers might potentially be using our code, so having it be efficient, clean, and well-documented is essential.

Background Knowledge on Machine Learning Models from Chem 277: In Chem 277, I gained a solid understanding of how machine learning models work, including their underlying principles, algorithms, and applications. This knowledge will help us in selecting appropriate machine learning techniques for the project, fine-tuning model parameters, and interpreting the model's outputs effectively.

By transferring these technical concepts and techniques learned in the MSSE program to my Capstone project, I hope we can produce a product that is helpful to our core organization, Merck. Also this integration of skills and knowledge acquired throughout the program with a real-world project will better prepare me for future work.

Personal Reflection

I am very proud of all that the team has accomplished. We were all relatively new to working with LLMs and with front-end development. Throughout the course of the project, our deliverables have slightly changed and our methods of achieving those deliverables have changed a lot. It was a big learning curve, but we have done well in being responsive to feedback. From the Merck team, we have gotten a lot of technical feedback, along with what would make the end product most useful for them. And our MSSE mentor's feedback was vital in keeping us on track and with reasonable goals. Additionally, I will definitely use my peer feedback to emphasize new things in the final presentation. We have all learned plenty of new skills throughout this project and I believe that the final deliverable will be very useful, even if it is just a starting point for a longer future project.

Sources

Alves, D., Guerreiro, N., Alves, J., Pombal, J., Rei, R., de Souza, J., Colombo, P., and Martins, A. (2023).Steering Large Language Models for Machine Translation with Finetuning and In-Context Learning.In Bouamor, H., Pino, J., and Bali, K., editors, Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore. Association for Computational Linguistics.

Dettmers, Tim, Artidoro, Pagnoni, Ari, Holtzman, Luke, Zettlemoyer. "QLoRA: Efficient Finetuning of Quantized LLMs". *arXiv preprint arXiv:2305.14314*. (2023).

Docker. "Enterprise Application Container Platform | Docker." *Docker*, 2024, www.docker.com/.

Gemma Team, et al. "Gemma: Open Models Based on Gemini Research and Technology." *ArXiv.org*, 13 Mar. 2024, arxiv.org/abs/2403.08295.

Leandro von Werra, , Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang. "TRL: Transformer Reinforcement Learning." . (2020).

"Merck | Home." *Merck.com*, Merck & Co., 2024, www.merck.com/.

Moslem, et al. “Fine-tuning Large Language Models for Adaptive Machine Translation”. 2023. arXiv:2312.12740v1 [cs.CL]

NERSC. “Perlmutter.” *NERSC*, 28 Mar. 2024, www.nersc.gov/systems/perlmutter/. “Using Perlmutter - NERSC Documentation.” *Docs.nersc.gov*, docs.nersc.gov/systems/perlmutter/.

Ramírez, S. (2023, October 26). FastAPIVersion (1.2.0). Retrieved from https://fastapi.tiangolo.com.

Song, Yifan, et al. “RestGPT: Connecting Large Language Models with Real-World RESTful APIs.” *ArXiv.org*, 26 Aug. 2023, arxiv.org/abs/2306.06624.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, & Benjamin Bossan. (2022). PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. .

Touvron, Hugo, et al. “Llama 2: Open Foundation and Fine-Tuned Chat Models.” *ArXiv.org*, 19 July 2023, arxiv.org/abs/2307.09288.

Vaswani, Ashish, et al. “Attention Is All You Need.” *ArXiv.org*, 5 Dec. 2017, arxiv.org/abs/1706.03762.