PennPalGPT

Grace Benner **Natalie Gilbert** **Yash Nakadi** **Mikaela Spaventa**

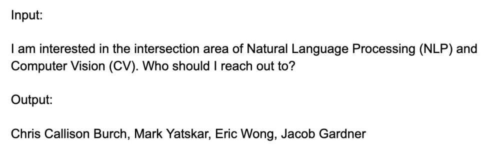
Abstract

Many students struggle to find a fitting research advisor, especially in an institution such as the University of Pennsylvania, where there are hundreds of faculty in each department. We propose a tool, PennPalGPT, where students can explain their research interests, and will be matched with a fitting professor, specifically from the CIS department at UPenn. Using context data, such as academic papers, from over 100 different faculty, we explore different approaches to building this model. First, we utilized GPT-3.5 and to explore a simple baseline, and evaluated the effects of prompt tuning and using GPT-4. We also took inspiration from concept bottleneck modeling, to create a more interpretable model.

1 Introduction

The rapidly evolving developments in Natural Language Processing (NLP) and Computational Linguistics present the opportunity to resolve a unique challenge: matching students with professors for capstone projects or research opportunities. As the number of professors in academic departments grows, finding the right mentor becomes increasingly daunting for students. To address this issue, our project aims to develop an intelligent QA chatbot tailored for students at the University of Pennsylvania's Computer and Information Science (CIS) department. This chatbot will assist students in identifying professors whose research aligns with their interests, simplifying the search process and fostering meaningful academic collaborations.

Imagine a scenario where a student, eager to embark on a capstone project, interacts with a chatbot. They input their research interests, and the chatbot, leveraging our NLP models, generates a curated list of relevant professors. This innovation not only streamlines the professor-student matching process but also serves as a valuable tool for enhancing the academic experience within the CIS department.



Input:

I am interested in the intersection area of Natural Language Processing (NLP) and Computer Vision (CV). Who should I reach out to?

Output:

Chris Callison Burch, Mark Yatskar, Eric Wong, Jacob Gardner

Figure 1: Input Output Example

Please see Figure 1 above to see what an example input and output looks like. The input question asks the chatbot to return professors that align with their research interests and then the chatbot returns the names of four professors that are relevant to the question.

The task at hand involves developing a QA chatbot that, given a student's research interests, outputs a list of professors ordered by relevance. The project's primary dataset comprises academic papers, bios, and contact information from the UPenn Engineering faculty in the CIS department. Our goal is to use state-of-the-art NLP models, including ROBERTA, and models in the GPT family, to create a model that can output appropriate answers when running in the backend of a chatbot.

We selected this task due to its practical implications for students seeking mentorship. The complexity lies in the effective extraction and interpretation of information from academic papers, bios, and contact details to facilitate meaningful professor-student connections. This aligns with our academic interests in NLP, and the project provides an opportunity to explore the capabilities of cutting-edge language models in solving real-world challenges. In the subsequent sections of this paper, we will delve into the literature review, detailing relevant papers and methodologies adopted from the NLP domain. Our evaluation metric, inspired by recent advancements in NLG evaluation, will ensure the robustness of our chatbot's responses. Additionally, we will discuss our milestones, baseline models, encountered obstacles, and our innovative extension using concept bottleneck models to enhance interpretability. As we embark on this journey, our overarching aim is to not only build an efficient chatbot but to contribute valuable insights to the broader field of NLP and its applications in academia.

2 Literature Review

The use of NLP tools to create a more human experience in communicating with the user has become increasingly popular. Having a dialog system (or chatbot) to help with fine-tuned problem such as high school tutoring Nguyen et al. (2019) or household tasks in QuakerBot of Pennsylvania (2022) has been growing rapidly. We take inspiration from QuakerBot, a project in which a chatbot is able to direct a user to a set of instructions or tasks in order to achieve a certain goal, and will answer questions specifically to the task at hand. QuakerBot must find appropriate context (as in a set of instructions or a recipe) to get the user to their desired goal, must process and responds properly to the user's question, and filter out spam. They use few-shot prompting to help the bot learn desired responses.

Our first attempt at our advising chatbot utilizes GPT-3.5 (specifically gpt-3.5-turbo) to find desired context from our training data and supply an answer to the user. GPT-3.5 series models have an exceptionally high capability to perform a variety of NLP tasks Ye et al. (2023). The new GPT-3.5 models have had much more instruction tuning, and with the latest gpt-3.5-turbo having an additional layer of chat tuning, these models handle user interaction in a much more sophisticated manner. One disadvantage this paper found of gpt-3.5-turbo was its over-compensation with its human interaction and quality of communication. This model would sometimes not perform the task ideally, but would sound the most "human". We did notice some highly verbose responses from a lightly prompted gpt-3.5-turbo model, so with some prompt tuning we were able to see more tailored responses to the desired task.

Inherently, these extremely large language models, such as the GPT models from OpenAI, lack any form of interpretability. In order to understand the underlying workings of our chatbot more precisely, we examine the idea of concept bottleneck modeling Yang (2023). Although black-box models have an undeniable ability to perform at NLP tasks, CBMs incorporate human-designed concepts into the modeling decisions so we can understand the "why" behind the output. CBMs often are used with a combination of both computer vision and natural language processing, the most common task is classifying images. The process involves identifying features for each category, getting a model to then identify the features of the new object, and using the trained model, to output the final classification of the new object. Since we want to match students with a given advisor(s), we can essentially frame our problem in a multi-class classification setting, using a model to output a professor given the features of the student's research interests.

To figure out how to evaluate our models, the paper *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment* Liu (2023) suggests evaluating the quality of text using an LLM with chain-of-thoughts that outlines each step involved in assigning a score. The authors present an outline of an example prompt that should be inputted to an LLM. It includes a task introduction, which introduces a summary of the task it will complete, an evaluation criteria, which lists the criteria expected for scoring, and evaluation steps, which include complete instructions for the model to follow. In the paper, the authors also noted two key issues with evaluating text with LLMs. First, one score is usually more represented than the others, leading to low variance of the scores. Also, even when prompted to return decimal values, LLMs will return integer scores in most cases. This inspired the researchers to normalize the scores and find their weighted summation for the final results score. The metrics used for text evaluation were naturalness, coherence, engagingness, groundedness, and the overall average of these scores. In their experiment, they compared G-Eval to prior scoring methods, including ROUGE, BERTScore, and UniEval. As a result, the authors found that G-Eval using GPT-4 performed best overall compared to earlier methods. Across all categories, G-Eval GPT-4 performed best except for 'engagingness' where G-Eval GPT-3.5 performed best.

3 Experimental Design

3.1 Data

Training Data:

The data we used to construct our training dataset comprises academic papers, bios, and contact information collected from 102 faculty members in the Computer and Information Science (CIS) department at the University of Pennsylvania. There were on average, 20 papers per professor, along with aforementioned metadata. This information was collected by individually downloading and saving the data. For academic papers, we took the first page of every paper to limit the size of the data. In our baseline and strong baseline models in Milestone 2, we used this data in our GPT-3.5 prompt to generate an answer. In Milestones 3 and 4, we used this data to generate features using a GPT model that could be used in a tabular dataset about our professors. In Milestone 3, we used GPT-3.5 to iteratively ask our model to generate features for every professor in our dataset. Disappointed with the results of this, we adapted our model in Milestone 4 to be a GPT-4 that generates feature related to the topics of a random set of 200 papers or bios. We hand-selected high quality features and came up with broad topic areas from the output of this model and used it to manually build a dataset of 112 features and a row for each professor.

Dev/Testing Data:

Our dev and test sets, which each make up 50 questions and their expected responses, were created by hand. In some of our questions, we used the bios of Ph.D. students to attempt to match them to their advisors, and for others, we came up with questions to find professors in various subject areas. We use the dev and test sets to transform each question into a vector with the same features as the training data. That way we can predict which group (there are 102 groups where each one represents a professor) the question aligns with.

Please see a summary of our data in the table below:

Dataset Type:	Pre/Post Transformation	Size
Training Data	Pre-Transformation	2410 papers, bios, and contact information for 102 professors
Training Data	Post-Transformation	102 rows, 112 features
Testing Data	Pre-Transformation	50 questions and answers
Testing Data	Post-Transformation	50 rows, 112 features
Dev Data	Pre-Transformation	50 questions and answers
Dev Data	Post-Transformation	50 rows, 112 features

Figure 2: Data Summary Table

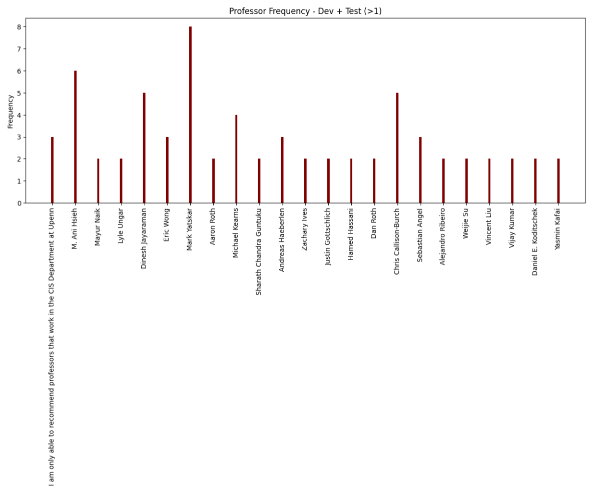


Figure 3: The dev and test data we have repeating targets for a few professors. Professor Yatskar has the most with 8 occurrences and Professor Hsieh has 6 occurrences. In our training set, we only have one occurrence for every professor.

3.2 Evaluation Metric

For our evaluation metric, we use gpt-4-0613 and prompt engineering to evaluate how semantically similar the chatbot answer is to the golden standard answer on a scale between 1 and 5, where 1 is the lowest possible score and 5 is the highest. We created groups of professors to account for cases when similar professors are outputted in our scoring. Please see the groups in Figure 4. We used K-Means++ to help us create these. We further researched this and came across the paper *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment* Liu (2023), which inspired many components of our evaluation metric. First, we wrote a prompt that instructed the model with the task of evaluating how semantically similar the inputted chatbot answers and golden standard answers are based on our criteria, which outlines the condition for each score to be received.

Upon finding and printing the score for every chatbot answer compared against its golden standard answer, we find the total score that evaluates the overall performance of the chatbot. The method we used was also inspired by *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment* Liu (2023). In the paper, the authors noted two key issues with evaluating text with LLMs. First, one score is usually more represented than the others, leading to low variance of the scores. Also, even when prompted to return decimal values, LLMs will return integer scores in most cases. This inspired the researchers to normalize the scores and find their weighted summation for the final results score. We adopted this approach to avoid the issues the authors of the paper points out.

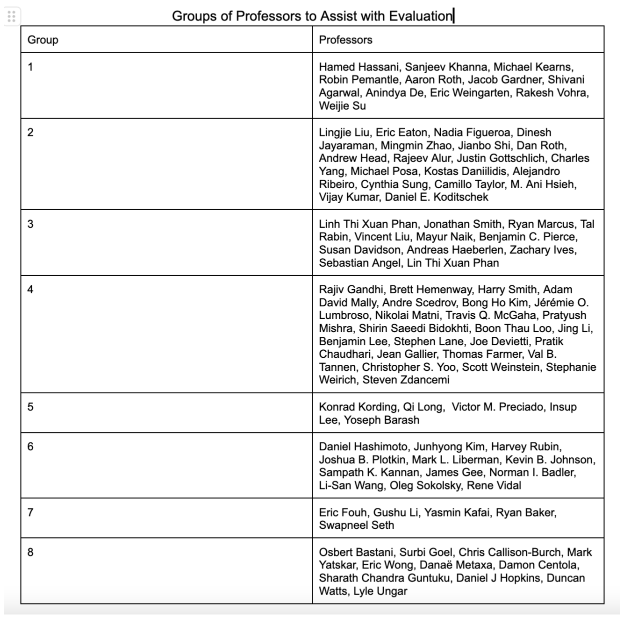
In order to achieve this, we stored the scores of each chatbot sentence and found the probability of each score occurring. Then, we multiply each score and its probability and take the sum of these. The minimum possible score that can be returned is 0. This is the very unlikely case that our evaluator cannot properly return scores from 1 to 5 and returns something unrecognized. We will get a final score closer to 0 when more chatbot scores are close to 1 and 2. On the other hand, the maximum possible sum to achieve is 5, when chatbot answers are evaluated to be 5 100% of the time. We divide the final sums by 5 so all final scores will be between 0 and 1.

In other words, given our set of scores S = $\{S_1, S_2, ..., S_5\}$, and the probability of each score $p(S_i) \forall i \in 1..5$, the final score is

$$\sum_{i=1}^{5} \frac{p(S_i) * S_i}{5}$$

.

We have found that Cai et al. (2023) also cited the paper that introduces G-eval and uses GPT-4 as well to measure the quality of generative answers. In addition Qi et al. (2023) used GPT-4 to evaluate the correctness of their model.



Groups of Professors to Assist with Evaluation

Group	Professors
1	Hamed Hassani, Sanjeev Khanna, Michael Kearns, Robin Pemantle, Aaron Roth, Jacob Gardner, Shivani Agarwal, Anindya De, Eric Weingarten, Rakesh Vohra, Weijie Su
2	Lingjie Liu, Eric Eaton, Nadia Figueroa, Dinesh Jayaraman, Mingmin Zhao, Jianbo Shi, Dan Roth, Andrew Head, Rajeev Alur, Justin Gottschlich, Charles Yang, Michael Posa, Kostas Daniilidis, Alejandro Ribeiro, Cynthia Sung, Camillo Taylor, M. Ani Hsieh, Vijay Kumar, Daniel E. Koditschek
3	Linh Thi Xuan Phan, Jonathan Smith, Ryan Marcus, Tal Rabin, Vincent Liu, Mayur Naik, Benjamin C. Pierce, Susan Davidson, Andreas Haeberlen, Zachary Ives, Sebastian Angel, Lin Thi Xuan Phan
4	Rajiv Gandhi, Brett Hemenway, Harry Smith, Adam David Mally, Andre Scedrov, Bong Ho Kim, Jérémie O. Lumbroso, Nikolai Matni, Travis Q. McGaha, Pratyush Mishra, Shirin Saeedi Bidokhti, Boon Thau Loo, Jing Li, Benjamin Lee, Stephen Lane, Joe Devietti, Pratik Chaudhari, Jean Gallier, Thomas Farmer, Val B. Tannen, Christopher S. Yoo, Scott Weinstein, Stephanie Weirich, Steven Zdancemi
5	Konrad Kording, Qi Long, Victor M. Preciado, Insup Lee, Yoseph Barash
6	Daniel Hashimoto, Junhyong Kim, Harvey Rubin, Joshua B. Plotkin, Mark L. Liberman, Kevin B. Johnson, Sampath K. Kannan, James Gee, Norman I. Badler, Li-San Wang, Oleg Sokolsky, Rene Vidal
7	Eric Fouh, Gushu Li, Yasmin Kafai, Ryan Baker, Swapneel Seth
8	Osbert Bastani, Surbi Goel, Chris Callison-Burch, Mark Yatskar, Eric Wong, Danaë Metaxa, Damon Centola, Sharath Chandra Guntuku, Daniel J Hopkins, Duncan Watts, Lyle Ungar

Figure 4: Groups of Professors

3.3 Simple Baseline

As a starting point, we implemented a simple baseline model using GPT-3.5's gpt-3.5-turbo with a temperature of 0. This model employs HuggingFace Embeddings (msmarco-bert-base-dot-v5) and utilizes only the first page of each PDF in the training data as context. The baseline's prompt is designed to provide straightforward instructions for generating answers. Please check below to get a qualitative and quantitative evaluation of this model:

Qualitative: At this stage, our chatbot results have a lot of room for improvement. We expected the context to be enough to answer most questions, but we have many cases when the chatbot was unsuccessful in outputting an answer other than 'I do not know'. Another theme we noticed is that many professors returned by the chatbot were not af iliated with UPenn. Some outputted professors do not exist. such as 'Professor Alice'. or teach at other universities. Another issue we noticed is that our chatbot suggests finding a professor that specializes in a certain area but does not output a specific name. None of the chatbot answers were actually correct but we still gave most of the answers a 2 / 5 for at least outputting something somewhat relevant.

Quantitative: .458 (for most answers we received a 2 / 5)

4 Experimental Results

4.1 Published Baseline

We do not have a published baseline that exactly follows our model creation and quantitative evaluation approach. The quantitative evaluation approach we used was just published a few months ago. While it has been used to evaluate some models already, the models don't closely relate to our use case.

4.2 Extensions

Both of our simple and strong baselines used a GPT-3.5 model to return a list of professors. Our strong baseline utilized an improved prompt to return a list of professors. As can be seen in Figure 5, the strong baseline does perform significantly better than the simple baseline. However, it still suffered from frequent hallucinations, answering in the inconsistent formats, and the fact that we could not trust the evaluation since the model did not have context on how similar professors were.

As a result of these issues, our first extension was to build a more informed model. With this approach, we attempted using a concept bottleneck modeling approach to build a machine learning model that will predict the professors most associated with a question. In this approach, we used a GPT-3.5 model to generate features used for machine learning. We asked the model to output a set of features for all 102 professors based on some of the articles associated with them for reference. From here, a very messy list of usually not very relevant features were returned. We adapted this by using few-shot prompting by including examples of article text and a list of topics related to the papers. The topics returned were improved, but there were still concerns over whether the reference articles for each author were actually relevant. From here, we hand-selected and created 93 features from this data for each professor. Next, we transformed our dev data into vectors by using a GPT-3.5 model to assign a value for each feature for each question. We then train a Random Forest, SVC, and KNN model with our professor data and predict the outputs for the questions. For this extension, we also built a K-means++ model to place our professors into different groups and updated our evaluation script to include these groups to account for similar professors appearing in the output.

In the first extension, we only evaluated the dev set because we knew we would likely be tuning this same model in our next Milestone and wanted to save our test set for the final model. At this stage, our model received a .456 for the Random Forest model, .598 for SVC, and .608 for KNN. We recognize that these results are not very different from our baselines, but we have more confidence in these results with an evaluation model that can determine if a group of professors are similar to each other or not. The scores for each question are less random and there are no longer hallucinations. However, at this point, many answers from the model were still very far from being right although they were much closer to being correct compared to the previous baselines.

In our second extension, we made several efforts to improve the results of our first attempt at the concept bottleneck modeling approach. First, we did a feature analysis to better understand what features were influencing our models most. Using $feature\_importances()$ in Random Forest, we found that the broad categories, such as $computer\_science\_bool$, $electrical\_engineering\_bool$, and $databases\_bool$ had the highest importance as can be seen in Figure 4. We then used a KNN approach to find the most important features, since it is more fitting for our use case, as can be seen in Figure 6. Inspired by (sta, 2019), we find the closest neighbor for each question. Then, we find 5 random neighbors and for each feature take the ratio of the closest neighbor distance and the random neighbor distance. Once we do this for every question, we take the sum of these ratios for every feature and the features with the lowest ratios are supposed to have a greater importance. We noticed that many of our important features were more specific, including $real\_time\_systems\_bool$, $blockchain\_bool$, and many others. As a result of this, we came to the conclusion that more features is likely better in our use case and it was in our best interest to add more features to improve our model. We also made other changes to improve the models in our pipeline, including using gpt-4-0613 instead of a GPT-3.5 model to generate features. We also improved the extraction approach by taking a random sample of 200 papers and prompting our GPT-4 model to directly find important topics from the papers. In addition, we added features so we had 112 instead of 93 and improved the groups of professors that we used in evaluation.

As a result, we saw an increase in overall score for both the test and dev set, which received scores of .718 and .7 respectively. We recognized that most of the time we received the expected professor in our output or several professors with similar interests. We still see a list of only unrelated professors somewhat often but it is much improved compared to the previous model and we're more confident about our results compared to those found in the baseline. Please see Figure 7 to see a summary table of each of our extensions.

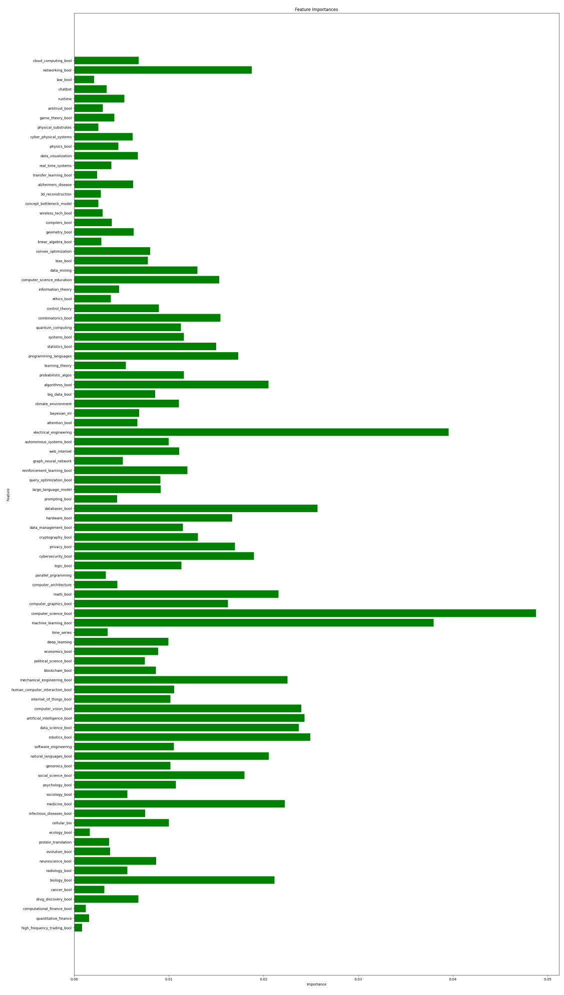


Figure 5: Random Forest Importances

KNN Feature Importances (Top 15)

Feature	Ratio Summed
data_mining_bool	5
real_time_systems_bool	5
blockchain_bool	5
cybersecurity_bool	5
geometry_bool	5
law_bool	5
psychology_bool	5
control_theory_bool	5
parallel_programming_bool	5
genomics_bool	5
alzheimers_disease_bool	10
ethics_bool	10
wireless_tech_bool	10
electrical_engineering_bool	10
evolution_bool	10

Figure 6: KNN Importances (Top 10 Most Important Features)

Model	Improvements	Quantitative Performance	Qualitative Performance
Simple Baseline	n/a	.458 (test)	- A lot of 'I do not know' answers - Many hallucinations Ex:'Professor Alice'. or professors at other universities. - Not answering in the correct format -None of the answers are correct
Strong Baseline	-Used gpt-3.5-turbo-16k instead of gpt-3.5-turbo -Instructed to only return a professor's name from UPenn and gave a list of all possible professors names	.644 (test)	- Less 'I don't know' answers - Less hallucinations but they are still frequent - Still not always answering in the correct format - Not many of the answers are correct - We don't trust the quantitative evaluation since this model doesn't have context on how similar the professors are
Extension One	-Used bottleneck concept approach -Changed evaluation prompt to account for professors with similar interests	Random Forest - .456 (dev) SVC - .598 (dev) KNN - .608 (dev)	- No hallucinations - Always answering in the right format - Many answers are incorrect or outputting very unrelated professors
Extension Two	-Used gpt-4-0613 instead of gpt-3.5 for feature generation and changed the extraction approach -Created 112 features instead of 93 -Used KNN model only - Changed grouping of professors in evaluation prompt	.7 (dev) .718 (test)	- Most guess the expected professor or some that are very similar to the expected professor -Still often receive a list of professors for answers we're not expecting one for

Figure 7: Summary of Each Extension, including the improvements quantitative results, and qualitative results

4.3 Error Analysis

In this section, we'll analyze the results and errors that occurred in our final model. Since we don't have a published baseline that aligns exactly with what we did, we compare our results from the previous milestone instead. The groups we created include 'Ideal Output', 'Expected professor not outputted but similar professors are', 'Most professors come from a different group', 'All professors come from another group', and 'Outputted professors instead of 'I don't know''.

The 'Ideal Output' is the situation where our chatbot outputs the professor we expected and similar professors as well. In most cases, these answers receive a '5/5'. These groups do not consist errors but we still look at them in our results to see how we did. 'Expected professor not outputted but similar professors are' is the error group that guesses many similar professors that from come from the same group but the expected professor is not outputted. Most of these results receive a 4 / 5. The next group 'Most professors come from a different group' is the situation where most of the professors listed come from a different group and usually score a 3 / 5 but occasionally a 4 / 5 or 2 / 5 (since the score prediction model isn't 100 percent accurate). Next, 'All professors come from another group' is when all professors outputted by the chatbot are in a different group from our expected professor and usually these answers get a 2 / 5 score. Last, 'Outputted professors instead of 'I don't know'' is the error that occurs when we're expecting 'I don't know' for an unrelated question but get a list of professors instead. These answers get a score of 1 / 5.

We looked at the errors of a sample of 50 data points for each Milestone's results. In Figure 8, we see a breakdown of the results from our final model. The most common error is that most professors come from another group, which occurs in 16 out of the 50 questions. The next most common error is 'Expected professor not outputted but similar professors are', which occurs in 13 out of 50 cases. Please note that this is more a soft error since in some cases our expected professor isn't necessarily the only correct answer.

In comparison to the previous Milestone (using the same evaluation code), it can be seen that only 8 out of 50 are a part of the ideal group compared to 11 out of 50. From the results in this Milestone, 20 compared to 8 in our final model had occurrences in which all professors came from other groups. As a result, there is a major difference in the number errors between our final model and previous Milestone.

We then further looked at the errors made of what groups were predicted. For all results, we found the actual group, representing the group of the expected professor, and the predicted group, which is the majority group of the predicted professors. As you can see in the confusion matrix in Figure 10, we are very successful in correctly predicting those in Group 2, which is mostly made of professors interested in robotics. However, we notice that there are also many cases where we incorrectly predicted other groups to be Group 2, especially Group 7, which mostly makes up professors interested in computer science education.

In the previous milestone, we still did very well at correctly predicting professors in Group 2. We also did well at predicting professors in Group 8, our primary machine learning and artificial intelligence group. However, it guessed incorrectly very often and when we did, it was frequently guessed professors in Group 7. Another common mistake is professors being predicted to be in Group 4, one of our 'other' groups that mostly makes up professors in other departments, when they are actually in Group 3, our software engineering and databases professors.

We also provided a table of examples of each type of error in Figure 12. In this first example, it can be seen that the expected professor was Lin Thi Xuan Phan who is in Group 3 but the chatbot answered Pratyush Mishra, Tal Rabin, Andre Scedrov, Jing Li, which are mostly in Group 4. In the second example, we received a list of professors instead of the expected variation of an 'I don't know' answer. In the third example, our chatbot outputted many professors from the same group, Group 8. In the fourth, we got an ideal answer, where the expected professor was outputted and is first in the list. Then in the last example, our expected professors are in Group 8, but we mostly received professors from Group 2.

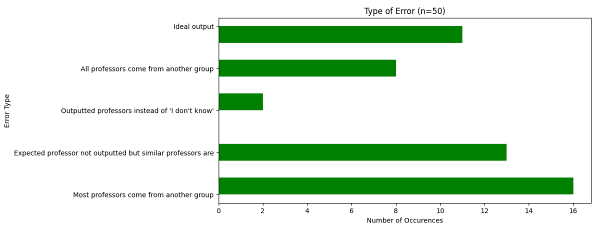


Figure 8: Types of Errors that Occur in the Sampled Data for Final Model

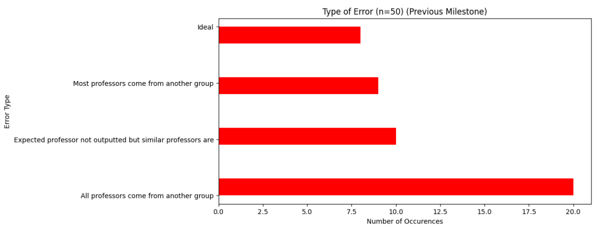


Figure 9: Types of Errors that Occur in the Previous Milestone

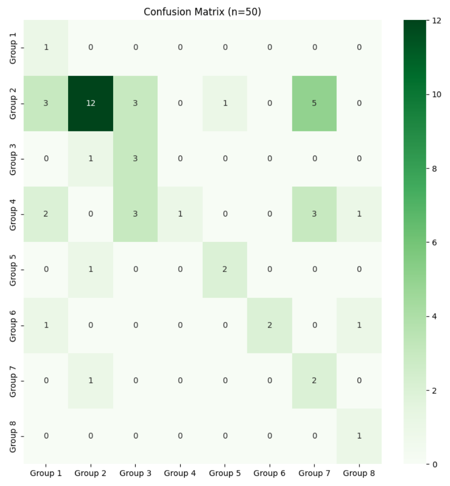


Figure 10: Confusion Matrix for Sampled Data in Final Model. Please note the y- axis is the predicted values and the x-axis is the actual values.

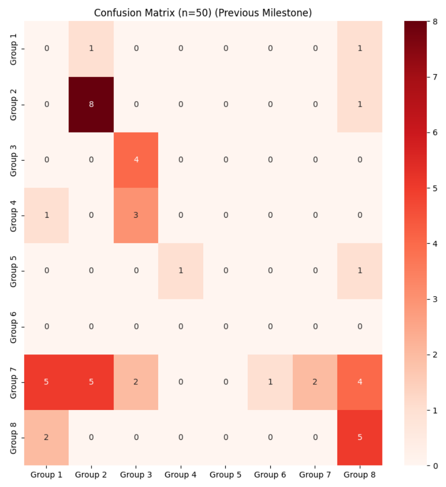
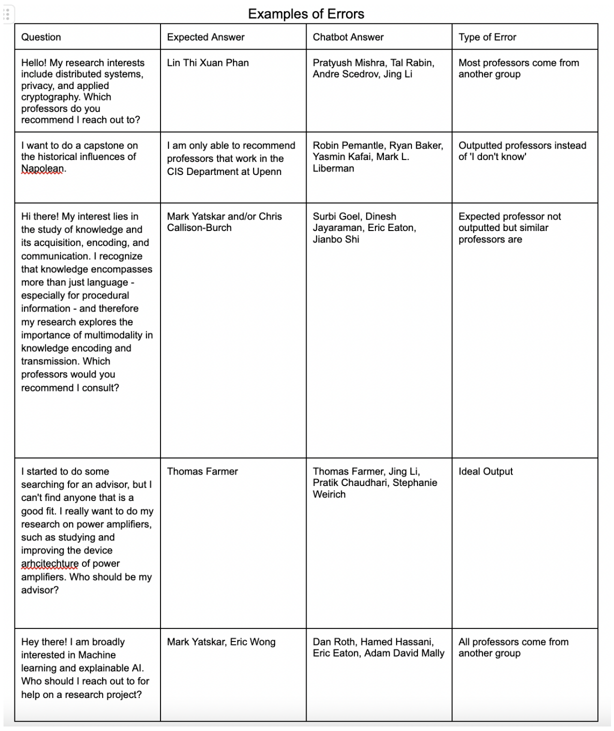


Figure 11: Confusion Matrix for Sampled Data in the Previous Milestone. Please note that the y-axis is the predicted values and the x-axis is the actual values.



Examples of Errors

Question	Expected Answer	Chatbot Answer	Type of Error
Hello! My research interests include distributed systems, privacy, and applied cryptography. Which professors do you recommend I reach out to?	Lin Thi Xuan Phan	Pratyush Mishra, Tal Rabin, Andre Scedrov, Jing Li	Most professors come from another group
I want to do a capstone on the historical influences of Napolean.	I am only able to recommend professors that work in the CIS Department at Upenn	Robin Pemantle, Ryan Baker, Yasmin Kafai, Mark L. Liberman	Outputted professors instead of 'I don't know'
Hi there! My interest lies in the study of knowledge and its acquisition, encoding, and communication. I recognize that knowledge encompasses more than just language - especially for procedural information - and therefore my research explores the importance of multimodality in knowledge encoding and transmission. Which professors would you recommend I consult?	Mark Yatskar and/or Chris Callison-Burch	Surbi Goel, Dinesh Jayaraman, Eric Eaton, Jianbo Shi	Expected professor not outputted but similar professors are
I started to do some searching for an advisor, but I can't find anyone that is a good fit. I really want to do my research on power amplifiers, such as studying and improving the device arhcitechture of power amplifiers. Who should be my advisor?	Thomas Farmer	Thomas Farmer, Jing Li, Pratik Chaudhari, Stephanie Weirich	Ideal Output
Hey there! I am broadly interested in Machine learning and explainable AI. Who should I reach out to for help on a research project?	Mark Yatskar, Eric Wong	Dan Roth, Hamed Hassani, Eric Eaton, Adam David Mally	All professors come from another group

Figure 12: Examples of Errors

5 Conclusions

For our topic, we addressed the challenge of matching students with suitable research advisors at the UPenn. Our tool, PennPalGPT, leverages state-of-the-art NLP models, primarily GPT-3.5, to build a state-of-the-art QA chatbot. We explored different approaches, from baselines with differing prompts to differing ML implementations of a concept bottleneck model, aiming to enhance interpretability.

While our implementations did not achieve state-of-the-art performance, they significantly improved over our simple baseline. The strong baseline, using GPT-3.5-16k-turbo with refined prompts, demonstrated notable enhancements, and the Concept Bottleneck Model, incorporating machine learning for interpretability, showcased promising results, particularly with the KNN algorithm. Our extensions, including feature analysis and additional features, further improved the model's performance on both the dev and test sets.

Our project provides valuable insights into leveraging advanced NLP models for personalized professor-student matching, and emphasizes the importance of interpretability and iterative model refinement in real-world applications. The improvements achieved underscore the potential of combining traditional machine learning approaches with cutting-edge language models for more accurate and interpretable outcomes in the academic domain.

Acknowledgements

We would like to acknowledge all the help and guidance we received from our TA Mona Gandhi.

Appendix

This first snippet is the feature generation prompt we used for feature generation:

"""You are an intelligent assistant that can extract key features or topics from the article in question that can be used for machine learning Please read each article carefully and make sure the topics outputted are broad.

The question is: What are main topics from <insert paper text>?

Next snippet is the prompt used to transform the questions into features:

$TASK\_PROMPT$ = """ You are an intelligent assistant that can determine the output to a given question.

Here is the given question: <insert question here>

First, if the question does NOT relate to research, the University of Pennsylvania, classes, school, or professors. Return 'I do not know the answer to that'.

If the question DOES relate to research, the University of Pennsylvania, classes, school, or professors, follow the prompt below:

You are an intelligent assistant that can determine if a given question is related to a particular topic. For each of the following topics, return a 1 if the given question is related to the topic, and 0 if the question is not related to the topic. Please read each topic carefully.

Reminder: the question is: <insert question here>

The topics:

1. High Frequency Trading
2. Quantitative Finance
3. Computational Finance
4. Drug Discovery
5. Cancer
6. Biology
7. Radiology
8. Neuroscience
9. Evolution
10. Protein Translation
11. Ecology
12. Cellular Biology
13. Infectious Diseases
14. Medicine
15. Sociology
16. Psychology
17. Social Science
18. Genomics
19. Natural Languages
20. Software Engineering
21. Robotics
22. Data Science
23. Artificial Intelligence
24. Computer Vision
25. Internet of Things

26. Human Computer Interaction
27. Mechanical Engineering
28. Blockchain
29. Political Science
30. Economics
31. Deep Learning
32. Time Series
33. Machine Learning
34. Computer Science
35. Computer Graphics
36. Math
37. Computer Architecture
38. Parallel Programming
39. Logic
40. Cybersecurity
41. Privacy
42. Cryptography
43. Data Management
44. Hardware
45. Databases
46. Prompting
47. Large Language Models
48. Query Optimization
49. Reinforcement Learning
50. Graphical Neural Networks
51. Web Internet
52. Autonomous Systems
53. Electrical Engineering
54. Attention in Natural Language Processing
55. Bayesian Machine Learning
56. Climate Environment
57. Big Data
58. Algorithms
59. Probabilistic Algorithms
60. Statistical Learning Theory
62. Programming Languages
63. Statistics
64. Systems
65. Quantum Computing
66. Combinatorics
67. Control Theory
68. Ethics
69. Information Theory
70. Computer Science Education
71. Data Mining
72. Bias
73. Convex Optimization
74. Linear Algebra
75. Geometry
76. Compilers
77. Wireless Technology
78. Concept Bottleneck Model
79. 3D Reconstruction
80. Alzheimer's Disease
81. Transfer Learning
82. Real-Time Systems
83. Data Visualization
84. Physics
85. Cyber-Physical Systems
86. Physical Substrates
87. Game Theory
88. Antitrust
89. Algorithm Runtime
90. Chatbots
91. Law
92. Networking
93. Cloud Computing
94. Education Experience
95. Social Media
96. News or Media
97. Gaming or Video Games
98. Amplifiers
99. Multicalibration
100. Explainable Artifical Intelligence
101. Robot Learning
102. Complexity Theory
103. Adversarial Learning
104. Online Learning
105. Video Artificial Intelligence or Video Machine Learning
106. Distributed Systems
107. Autism
108. Swimming Robots
109. Surgical Artificial Intelligence
110. Robot Kinematics
111. Human-like Robots
112. Graph Theory

""" prompt = $TASK\_PROMPT$.replace("question", question)

The last snippet we have is the prompt that we used to evaluate the outputs. We asked GPT-4 to compare the chatbot answer against the golden standard answer.

$TASK\_PROMPT$ = """ Your task to return a score that ranges from 1 to 5 (where 1 is the lowest and 5 is the highest score) based on how close the chatbot and golden standard answer are in semantic meaning and if they give the same answer.

Group 1: Hamed Hassani, Sanjeev Khanna, Michael Kearns, Robin Pemantle, Aaron Roth, Jacob Gardner, Shivani Agarwal, Anindya De, Eric Weingarten, Rakesh Vohra, Weijie Su

Group 2: Lingjie Liu, Eric Eaton, Nadia Figueroa, Dinesh Jayaraman, Mingmin Zhao, Jianbo Shi, Dan Roth, Andrew Head, Rajeev Alur, Justin Gottschlich, Charles Yang, Michael Posa, Kostas Daniilidis, Alejandro Ribeiro, Cynthia Sung, Camillo Taylor, M. Ani Hsieh, Vijay Kumar, Daniel E. Koditschek

Group 3: Linh Thi Xuan Phan, Jonathan Smith, Ryan Marcus, Tal Rabin, Vincent Liu, Mayur Naik, Benjamin C. Pierce, Susan Davidson, Andreas Haeberlen, Zachary Ives, Sebastian Angel, Lin Thi Xuan Phan

Group 4: Rajiv Gandhi, Brett Hemenway, Harry Smith, Adam David Mally, Andre Scedrov, Bong Ho Kim, Jeremie O. Lumbroso, Nikolai Matni, Travis Q. McGaha, Pratyush Mishra, Shirin Saeedi Bidokhti, Boon Thau Loo, Jing Li, Benjamin Lee, Stephen Lane, Joe Devietti, Pratik Chaudhari, Jean Gallier, Thomas Farmer, Val B. Tannen, Christopher S. Yoo, Scott Weinstein, Stephanie Weirich, Steven Zdancemi

Group 5: Konrad Kording, Qi Long, Victor M. Preciado, Insup Lee, Yoseph Barash

Group 6: Daniel Hashimoto, Junhyong Kim, Harvey Rubin, Joshua B. Plotkin, Mark L. Liberman, Kevin B. Johnson, Sampath K. Kannan, James Gee, Norman I. Badler, Li-San Wang, Oleg Sokolsky, Rene Vidal

Group 7: Eric Fouh, Gushu Li, Yasmin Kafai, Ryan Baker, Swapneel Seth

Group 8: Osbert Bastani, Surbi Goel, Chris Callison-Burch, Mark Yatskar, Eric Wong, Danae Metaxa, Damon Centola, Sharath Chandra Guntuku, Daniel J Hopkins, Duncan Watts, Lyle Ungar

Check the criteria below to assign a score:
- Return a 1 if a chatbot answer is not returned or the chatbot answer is not a professor and the golden standard answer is.
- Return 2 if NONE of the names in the chatbot answer are in the same group (see the groups above) as the golden standard answer.
- Return 3 if one of the names in the chatbot answer are professors that are in the same group (see the groups above) as the golden standard answer
- Return 4 if more than one of the names in the chatbot answer are professors that are in the same group (see the groups above) as the golden standard answer
- Return 5 if one of the chatbot answer names is the same professor as the golden standard answer OR the chatbot and golden standard answers are the same

The chatbot answer is $input_1$ and the golden standard answer is $input_2$.

You must return a number and nothing else between 1 and 5. The returned score is: """

References

2019. Stack overflow - how to find feature importance or variance importance for knnclassifier().

Pengshan Cai, Zonghai Yao, Fei Liu, Dakuo Wang, Meghan Reilly, Huixue Zhou, Lingxi Li, Yi Cao, Alok Kapoor, Adarsha Bajracharya, Dan Berlowitz, and Hong Yu. 2023. Paniniqa: Enhancing patient education through interactive question answering.

Iter D. Yichong X. Shuohang W. Ruochen X. Cheunguang Z. Liu, Y. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment.

Hien D. Nguyen, Vuong T. Pham, Dung A. Tran, and Trung T. Le. 2019. Intelligent tutoring chatbot for solving mathematical problems in high-school. In *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6.

University of Pennsylvania. 2022. Quakerbot: A household dialog system powered by large language models. In *Alexa Prize TaskBot Challenge 1 Proceedings*.

Jingyuan Qi, Zhiyang Xu, Ying Shen, Minqian Liu, Di Jin, Qifan Wang, and Lifu Huang. 2023. The art of socratic questioning: Recursive thinking with large language models.

et al. Yang, Yue. 2023. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification.

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, Jie Zhou, Siming Chen, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models.