

QuickAns: A Virtual Teaching Assistant

ISHIKA AGARWAL, ishikaa2@illinois.edu
SHRADHA SEHGAL, ssehgal4@illinois.edu
VARUN GOYAL, varun15@illinois.edu
PRATHAMESH SONAWANE, pks10@illinois.edu

QuickAns is a virtual teaching assistant designed to help course staff who use Campuswire as their Q&A platform. It reads Campuswire posts from digest emails, and sends a potential answer to the course staff. At this stage, the course staff can review the answer for any logistical issues, and answer a student's question in a matter of minutes. Our code can be found [here](#).

1 INTRODUCTION

As the course staff gets busy with course responsibilities, especially toward the end of the semester, it becomes difficult to answer Campuswire questions promptly. Additionally, many questions tend to be repetitive or fundamental. Our application software aims to reduce the time and effort spent on answering Campuswire questions for TAs and instructors by generating potential responses. The tool relieves them from typing answers to basic questions so they can focus their attention and expertise on more challenging tasks. Additionally, it provides them with relevant content relating to the topic of the question, along with a skeleton answer that they can elaborate on.

The functionality it offers includes automated and timely response suggestions to questions on Campuswire, made available through email. Specifically, our application will read emails received from Campuswire, generate a potential response, and communicate the response via email to the course instructors. They can subsequently review the response for any errors or omissions before posting it. This streamlined process will save the course staff valuable time that would have been spent crafting and refining a detailed reply. The end-users of this toolkit are expected to include all course staff i.e. instructors and teaching assistants.

This report will outline the pipeline in section 3, show examples of how to use QuickAns in section 4, and finally, demonstrate some lightweight evaluation on our current class questions in section 5. Below are our contributions:

- Ishika: Worked on the question-answering module, helped in evaluation
- Shradha: Set up Microsoft Graph API, research the question-answering module, helped in evaluation
- Varun: Set up the email scraping system, helped in evaluation
- Prathamesh: Set up the email scraping system, helped in evaluation

2 RELATED WORK

While this problem is somewhat new, multiple groups have started working on this.

Authors' addresses: Ishika Agarwal ishikaa2@illinois.edu; Shradha Sehgal ssehgal4@illinois.edu; Varun Goyal varun15@illinois.edu; Prathamesh Sonawane pks10@illinois.edu.

PVTA. For example, Benedetto and team have worked on a similar system that uses IBM's NLP engine called Watson Assistant (i.e., Assistant) [Benedetto et al. 2019] and hosts the bot on Slack. Some of their challenges include enabling long-term memory as IBM's Assistant mostly deals with short-term memory tasks. They aim to personalize it to their students in a recommendation systems course.

Kwame. Boateng has created a virtual TA for SuaCode - an online platform that teaches students to code - called Kwame [Boateng 2021]. Boateng noticed that the students are divided into English speakers and French speakers. Using Sentence-BERT as a base question-answer model, they trained a model on the course material and previous class versions' question-answer pairs and quiz questions.

Khanmigo. Recently, Khan Academy has introduced their own AI teaching assistant on the online learning platform called Khanmigo [kha [n. d.]]. It is another question-answering chatbot that can help students learn. It can not only understand reasoning in the form of math and code, but it can also find mistakes, understand intention and guide students to the right answer. For example, suppose a student is working on a math problem and makes a mistake. They can show the bot their work and the bot will understand what mistake was made and remind the student of a concept that they misunderstood or forgot.

3 METHODOLOGY

As mentioned before, we have built this application on top of the Campuswire platform. A fake user can be created that will subscribe to email alerts to all the posts made on the platform to access each question.

The email reading-scraping functionality is implemented using the Python Microsoft Graph API. A question will be inputted to our question-answering module which will generate replies and email them to the course staff. We have utilized the ChatGPT model via API to generate the responses to the questions as it is relatively fast, has language understanding, and has been instruction tuned. Language understanding gives the ability to answer improperly framed questions. Instruction Tuning is qualitatively proven to perform much better at Q&A tasks than non-tuned models. Overall our approach and email these generated replies to the course staff.

3.1 Question-Answering Module

As brought up earlier, the question-answering module uses ChatGPT as the backbone language model. To improve its ability to answer questions relating to a course, we also provide it the corresponding textbook. Because we developed this for our Information Retrieval class, we provided the book "Text Data Management and Analysis:

```
Welcome to QuickAns!

To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code EA3SGCCQF to authenticate.
```

Fig. 1. Microsoft Graph authentication.

```
Hello, User
Email: email@id.com

Anonymous posted a note in Advanced Information Retrieval
Anonymous posted a question in Advanced Information Retrieval
```

Fig. 2. Example emails scraped by the Microsoft Graph API.

A Practical Introduction to Information Retrieval and Text Mining" [Zhai and Massung 2016].

We employed a search-based method. We created chunks of the textbook (5-7 sentences; chunks can overlap) and created embeddings out of them using the 'text-embedding-ada-002' model. Then, we create embeddings out of a student's questions, and use a similarity function to determine the closest textbook paragraph to the question. From there, we feed in the question and the related paragraph into the GPT-based model and retrieve an answer.

Fine-tuning versus searching. A recent trend involves fine-tuning a language model on your text data to enable a language model to understand a specific type of material or domain. However, fine-tuning is not a great technique. A language should be fine-tuned on tasks, not data. Embedding search-based methods are better for focusing the context, which is the case in QuickAns. Furthermore, we chose to search chunks of text (as opposed to the entire textbook, or even chapter) because of the token limit. Even without the token limit, we expect the output to be somewhat the same.

4 USAGE

We make our code available on GitHub.

To start the service, we run `python3 quickAns.py`. As mentioned before, we use the Microsoft Graph API to read/scrape (only) Campuswire emails. So, the first thing a user will do after executing the script is giving them access to their emails (they need to enable all email notifications on Campuswire), as shown in Figure 1. After signing in, QuickAns will scrape emails and display the top k on the terminal, as shown in Figure 2. (Note: The user information is anonymized in the picture - in the system, it will display the name and email of the account that logged into Microsoft Graph.)

It will take the top post, send it to the QuickAns answering module, and print out the answer on the terminal. It will also email the exchange to an email account(s) linked to the course staff. Figure 3 displays a sample email that a TA would receive at the end of this pipeline.

Finally, this service needs to be hosted on a server so that the system can continuously receive/answer queries. The server will scrape emails every 5 minutes for new questions. In order to avoid inefficiencies, there is functionality included so that the same question is not answered multiple times.

| Average- | QuickAns | GPT3.5 | Difference |
|--------------------|----------|--------|------------|
| Jaccard Similarity | 8.26% | 7.76% | +0.50% |
| BLEU | 30.78% | 30.64% | +0.14% |
| ROUGE-1 | 26.08% | 24.67% | +1.41% |
| ROUGE-2 | 6.19% | 5.37% | +0.82% |
| ROUGE-L | 23.86% | 22.17% | +1.69% |

Table 1. GPT-3.5 and QuickAns evaluated over a variety of word-overlap and similarity metrics wrt the ground truth answers from Campuswire.

5 EVALUATION

5.1 Quantitative

We evaluate QuickAns on the current set of class questions we have (note: we were given access to previous offerings' Campuswire sections of this class). After removing the logistical questions asked on the class Campuswire, we were able to obtain 56 class content-related questions. We treated the answers of those questions as the ground truth answer. As a baseline, we use GPT-3.5, or InstructGPT [Ouyang et al. 2022]. We ask the same questions to GPT-3.5 and QuickAns's base model.

Table 1 highlights the performance of each model in various metrics such as Jaccard Similarity [Niwtanukul et al. 2013], BLEU [Papineni et al. 2002], ROUGE1, ROUGE-2 [Lin 2004] and ROUGE-L [Lin and Och 2004]. All of these metrics measure the similarity between two different texts. For QuickAns to work, the similarity between the ground truth and QuickAns' response should be higher than GPT-3.5's answer. As the table shows, QuickAns does better (if only slightly) than GPT-3.5 in all metrics.

5.2 Qualitative

In order to solidify the claim that using a search-based method is better than simply using GPT-3.5, we show a qualitative example. We ask both models certain questions.

5.2.1 Explaining concepts wrt course context: Multi-Bernoulli and multinomial. We first ask: "when modeling queries, how is multi-Bernoulli different from multinomial"? The answer should be related to word frequencies - i.e., in a multi-Bernoulli setting, the word

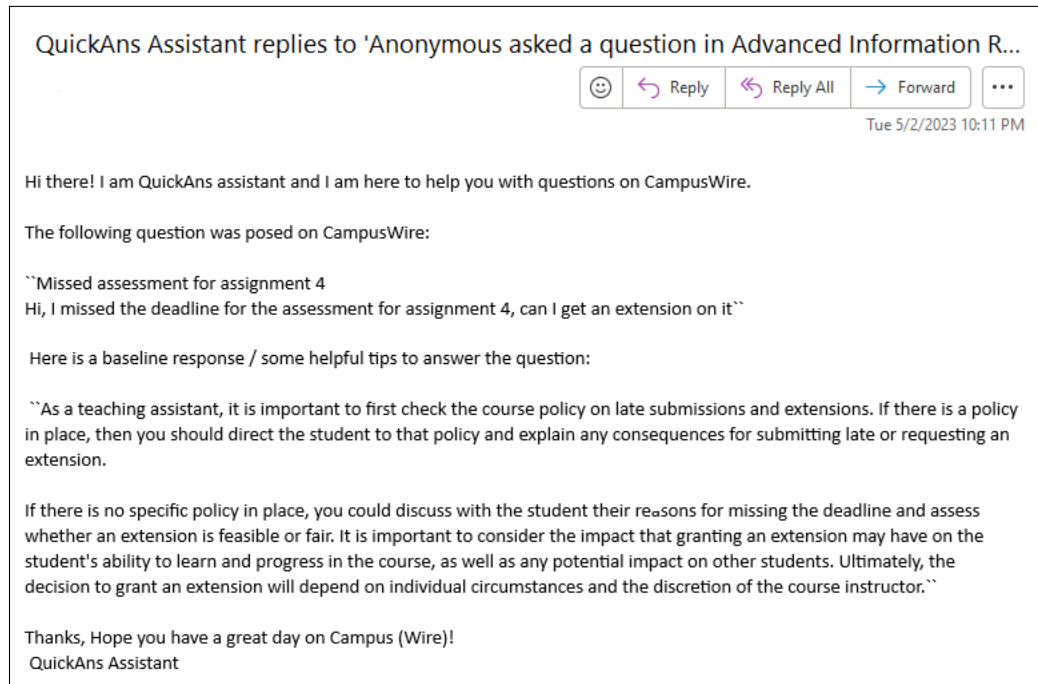


Fig. 3. A sample email that a TA would receive through the QuickAns pipeline.

frequencies in queries are modeled by absence or presence, while in a multinomial setting, they are modeled by term frequency.

Here is the answer given by GPT-3.5:

Multi-Bernoulli and Multinomial are two different models used in probability theory and statistics to model queries. Multi-Bernoulli is a model used to represent a sequence of binary events, where each event can have one of two outcomes (success or failure). It is used to model queries where the data is binary, such as in object tracking or sensor fusion. In Multi-Bernoulli, the probability of success or failure is assumed to be constant across all events. On the other hand, Multinomial is a model used to represent a sequence of events where each event can have one of several outcomes. It is used to model queries where the data is categorical, such as in natural language processing or image classification. In Multinomial, the probability of each outcome is assumed to be constant across all events. In summary, Multi-Bernoulli is used for binary data, while Multinomial is used for categorical data.

Here is the answer given by QuickAns:

Multi-Bernoulli and multinomial are both probability distributions used for modeling queries in information retrieval. The Bernoulli distribution models the probability of success for a single event, while the binomial distribution models the probability of success for multiple events. The multinomial distribution is similar to the binomial distribution, but it models the probability of observing each word in a vocabulary in a document, rather than just success or failure. Multi-bernoulli is a special case of the multinomial distribution, where the vocabulary size is limited to two. In other words, it models the probability of observing a binary outcome, such as the presence or absence of a word in a document. In terms of parameter estimation, both multi-bernoulli and multinomial can be estimated using maximum likelihood estimation (MLE) or Bayesian estimation. For Bayesian estimation, the Dirichlet distribution is often used as a prior for the multinomial distribution. The posterior distribution is also a Dirichlet distribution, due to the conjugacy between the multinomial and Dirichlet distributions. In summary, multi-bernoulli and multinomial are both probability distributions used for modeling queries in information retrieval, with multi-bernoulli being a special case of the multinomial distribution. Both can be estimated using MLE or Bayesian estimation with the Dirichlet distribution as a prior for the multinomial distribution.

Analysis. Both answers have explanations for what the Bernoulli and multinomial distributions are. However, that is where the GPT-3.5 answer stops. The QuickAns answer actually elaborates as to how these distributions relate to modeling queries and specifically mentions the notion of word frequencies.

5.2.2 Finding information from the textbook: word relations. The question is: "what type of word relations are there"? The expected answer should be paradigmatic and syntagmatic because the context is word association mining in information retrieval.

Here is the answer given by GPT-3.5:

There are several types of word relations, including:

1. Synonyms: words that have similar meanings, such as "happy" and "joyful."
2. Antonyms: words that have opposite meanings, such as "hot" and "cold."
3. Homonyms: words that sound the same but have different meanings, such as "flower" and "flour."
4. Homophones: words that sound the same but are spelled differently and have different meanings, such as "there" and "their."
5. Hyponyms: words that are more specific than a general term, such as "apple" being a hyponym of "fruit."
6. Hypernyms: words that are more general than a specific term, such as "vehicle" being a hypernym of "car."
7. Meronyms: words that are part of a larger whole, such as "wheel" being a meronym of "car."
8. Holonyms: words that are the larger whole of a part, such as "car" being a holonym of "wheel."

Here is the answer given by QuickAns:

There are two types of word relations: paradigmatic and syntagmatic. Words have a paradigmatic relation if they can be substituted for each other without affecting the understanding of the sentence. This means that the two words are in the same semantic or syntactic class. For example, the words "cat" and "dog" have a paradigmatic relation because they are both in the same word class, "animal." On the other hand, words have a syntagmatic relation if they are used together in a sentence. This means that the words are related in terms of their position in a sentence and the meaning they convey when used together. Chapter 14 of the course textbook provides additional ideas on how to group similar terms together.

Analysis. This example highlights the use of embeddings based search methods. While GPT-3.5's answer is correct, it was never given the context of information retrieval. In fact, even after some prompt engineering (asking GPT-3.5 "In the context of Information Retrieval, what type of word relations are there?"), GPT-3.5 still outputs the same answer. On the other hand, QuickAns was able to extract the context from the given textbook very well and answered the question perfectly, even with an example.

5.2.3 Asking something that GPT-3.5 would not know: CPLSA. Here is another interesting example that we noticed from the student questions. CPLSA stands for Contextual Probabilistic Latent Semantic Analysis. If one searches for CPLSA, one might find lecture videos/resources from UIUC's CS 510 class itself - meaning, CPLSA is an acronym that Professor Zhai (the author of the textbook we use) came up with. Therefore, if given the question: "In CPLSA, if each view has only one coverage, why do we have two steps (step1: choose a view, step2: choose a coverage)?", here are the different responses.

Given by GPT-3.5:

As an AI language model, I am not familiar with the specific context of CPLSA. However, in general, it is possible that a view may have multiple coverages, or that the choice of view may influence the available coverages. Additionally, the two steps may be necessary for organizational or procedural reasons, such as ensuring that all necessary information is collected and recorded in a systematic manner. It may be helpful to consult the specific guidelines or documentation for CPLSA to better understand the reasoning behind the two-step process.

Given by QuickAns:

Dear student, Thank you for your question. You are correct that each view only has one coverage in CPLSA. However, the reason we have two steps (step 1: choose a view; step 2: choose a coverage) is because each word can potentially be generated using a different view and a different topic coverage distribution depending on the contexts chosen to direct the generation process. In other words, the context that determines the choice of view of a topic can be different from the context chosen to decide the topic coverage. This allows for the discovery of different variations of the same topic in different contexts due to the dependency of a view of topics on context and different topic coverages in different contexts due to the dependency of topic coverage on context. I hope this helps clarify your question. Please let me know if you have any further questions or concerns. Best regards, [Your Name]

Analysis. First off, GPT-3.5 mentions that it does not know what CPLSA is. It gives an overly vague answer and concludes with a request for more information. On the other hand, QuickAns grasps the concept clearly and gives a perfectly correct answer. This is something that even some prompt engineering tactic ("In the context of Information Retrieval, ...") cannot solve. This also shows that QuickAns' dependency on GPT-3.5's knowledge base is quite low.

6 CONCLUSION AND FUTURE DIRECTIONS

In this report, we have outlined the pipeline and fine-grained details of our virtual teaching assistant, QuickAns. We evaluated it on some of the class questions we already have available to us and found that our method works better than the baseline.

We would also like to explore some future directions with this project. As mentioned in section 5.1, we took out the logistical questions because QuickAns was not aware of such data. However, in the future, we can make that possible by providing due dates and class policies to QuickAns to make it more powerful. Additionally, we would want to configure this project as a black-box into which course staff will provide their textbook and any class details they wish QuickAns to know. This way, it will become more and more accessible. Furthermore, we can enable this bot to other platforms such as Piazza, Canvas, Black Board, etc.

Next, language models are known to suffer from hallucination. In this case, we noticed that QuickAns was making references to unknown slides and textbook pages.

Finally, a sub-direction in the field of virtual teaching assistants is enabling such bots for personalized learning. An interesting problem to solve would be enabling this system for personalized learning. When generating responses, is there a way we can condition on the

complexity of the answer? One easy answer is prompt engineering, but we are interested in the idea of leaving the AI NLP-agent untouched and building modifications around it.

REFERENCES

- [n. d.]. Khanmigo Education Ai Guide. <https://www.khanacademy.org/khan-labs>
- Luca Benedetto, Paolo Cremonesi, and Manuel Parenti. 2019. A virtual teaching assistant for personalized learning. *arXiv preprint arXiv:1902.09289* (2019).
- George Boateng. 2021. Kwame: a bilingual ai teaching assistant for online suacode courses. In *Artificial Intelligence in Education: 22nd International Conference, AIED 2021, Utrecht, The Netherlands, June 14–18, 2021, Proceedings, Part II*. Springer, 93–97.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.
- Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. 605–612.
- Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. 2013. Using of Jaccard coefficient for keywords similarity. In *Proceedings of the international multiconference of engineers and computer scientists*, Vol. 1. 380–384.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 311–318. <https://doi.org/10.3115/1073083.1073135>
- ChengXiang Zhai and Sean Massung. 2016. *Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining*. Vol. 12. Association for Computing Machinery and Morgan amp; Claypool.