USING SMALL LANGUAGE MODELS AS TOOLS FOR TEACHING IN THE ELSEI MASTER'S PROGRAM

USO DE PEQUENOS MODELOS DE LINGUAGEM COMO FERRAMENTAS DE ENSINO NO PROGRAMA DE MESTRADO ELSEI

Zakaria Tagdimi
ORCID 0000-0003-1690-9182
Abdelmalek Essaadi University
Tetouan, Morocco
Zakaria.tagdimi@etu.uae.ac.ma

Tarik Touis
ORCID 0009-0008-4686-4273
Abdelmalek Essaadi University
Tetouan, Morocco
Tarik.touis@etu.uae.ac.ma

Souhaib Aammou
ORCID 0000-0002-6224-0929
Abdelmalek Essaadi University
Tetouan, Morocco
S.aammou@uae.ac.ma

Abstract. The Question Answering Introduction to Python (QAIP) system aims to enhance the learning experience in introductory Python courses by providing accurate and efficient answers to Python-related queries. The rise of Large Language Models (LLMs) has significantly impacted education, particularly within the framework of Education 4.0, which seeks to prepare students for a technologically advanced world. LLMs such as OpenAI's ChatGPT and GitHub's Copilot have revolutionized content creation, personalized tutoring, and student engagement, aligning with the goals of Education 4.0. However, the challenge of developing appropriate programming exercises and maintaining active learning in introductory programming courses persists, especially given the rapid online sharing of solutions. In this context, Small Language Models (SLMs) offer a lightweight, efficient alternative for educational integration. This article explores the integration of SLMs into the QAIP system within the E-learning and Intelligent Educational Systems (ELSEI) program, aiming to empower students with the skills to develop innovative educational tools. By narrowing the existing AI content gap, this work aspires to contribute to the broader discourse on AI accessibility and diversity. The development process involves thorough data collection, strategic model training, and careful deployment to ensure that the AI-driven system effectively meets student needs and enhances learning outcomes. Through this interdisciplinary effort, we aim to foster a culture of innovation and contribute meaningfully to the evolution of AI in education.

Keywords: Small language models; ELSEI; Question Answering system; Personalized Tutoring

Resumo. O sistema QAIP (Question Answering Introduction to Python) tem como objetivo aprimorar a experiência de aprendizado em cursos introdutórios de Python, fornecendo respostas precisas e eficientes para consultas relacionadas a Python. O surgimento de modelos de linguagem ampla (LLMs) teve um impacto significativo na educação, especialmente dentro da estrutura da Educação 4.0, que busca preparar os alunos para um mundo tecnologicamente avançado. Os LLMs, como o ChatGPT da OpenAI e o Copilot do GitHub, revolucionaram a criação de conteúdo, a tutoria personalizada e o envolvimento dos alunos, alinhando-se às metas da Educação 4.0. No entanto, o desafio de desenvolver exercícios de programação adequados e manter o aprendizado ativo em cursos introdutórios de programação persiste, especialmente devido ao rápido compartilhamento on-line de soluções. Nesse contexto, os Small Language Models (SLMs) oferecem uma alternativa leve e eficiente para a integração educacional. Este artigo explora a integração de SLMs no sistema QAIP dentro do programa E-learning and Intelligent Educational Systems (ELSEI), com o objetivo de capacitar os alunos com as habilidades para desenvolver ferramentas educacionais inovadoras. Ao reduzir a lacuna existente no conteúdo de IA, este trabalho pretende contribuir para o discurso mais amplo sobre acessibilidade e diversidade de IA. O processo de desenvolvimento envolve coleta minuciosa de dados, treinamento de modelos estratégicos e implementação cuidadosa para garantir que o sistema orientado por IA atenda efetivamente às necessidades dos alunos e aprimore os resultados da aprendizagem. Por meio desse esforço interdisciplinar, pretendemos fomentar uma cultura de inovação e contribuir de forma significativa para a evolução da IA na educação.

Palavras-chave: Modelos linguísticos pequenos; ELSEI; Sistema de resposta a perguntas; Tutoria personalizada

1. INTRODUCTION

In recent years, the rise of Large Language Models (LLMs) has significantly influenced the field of education. These models, such as Codex, OpenAI's ChatGPT and GitHub's Copilot, have revolutionized natural language processing, computer vision, and creative arts. Their ability to generate diverse and contextually relevant content makes them valuable tools for educational purpose. Within the framework of Education 4.0(Peláez-Sánchez et al., 2024), which aims to prepare new generations for a technologically fluid world. This educational landscape integrates emerging technologies and innovative strategies to create more autonomous, collaborative, and interactive learning experiences. LLMs align with this vision by enhancing content creation, personalized tutoring, and student engagement (Englhardt et al., 2024).

The exponential growth in the size of language models, now encompassing hundreds of billions of parameters, has fundamentally transformed their capabilities, enabling diverse applications in educational resource design, including the creation of materials, assignments, and course flow Kurdi et al. (2020). Leveraging advances in transformer-based architectures, such as attention mechanisms and contextual embeddings, these models exhibit emergent capabilities like zero-shot and few-shot learning, which significantly reduce the need for extensive human input. Pretrained on vast collection of textual data, LLMs can synthesize complex domain-specific knowledge, while fine-tuning on educational datasets enhances their relevance and accuracy for specific academic contexts (Bressane et al., 2024).

Through prompt engineering and reinforcement learning from human feedback (RLHF), these models can generate dynamic and personalized content, such as assignments calibrated by difficulty using perplexity measures or token probability distributions. Furthermore, LLMs can adapt course flow by analyzing learner profiles—leveraging embeddings to infer knowledge gaps and preferences—and integrating with knowledge graphs to ensure pedagogical coherence in the sequence of concepts (Chen et al., 2024). Quality assurance is facilitated through intrinsic and extrinsic evaluation metrics, while controlled generation techniques enhance content alignment with educational objectives. Moreover, the integration of LLMs into learning management systems (LMS) via APIs and automated pipelines supports scalable deployment, while real-time interaction capabilities enable dynamic learner engagement through question-answering systems and dialog management.

These technical innovations, underscore the potential of LLMs to redefine educational design by optimizing personalization, efficiency, and inclusivity, making them indispensable tools in the evolving landscape of digital education. Particularly through accurate and efficient support for teaching Python-related queries. This approach has become a prominent teaching method in computer science. Developing appropriate programming exercises is challenging, especially with the rapid online sharing of solutions. Educators face difficulties in creating novel exercises and tests, leading to the exploration of collaborative resource generation.

Large Language can play a key role in generating and supporting introductory programming courses, by providing personalized learning experiences and facilitating stable communication between teachers and students. (Sarsa et al., 2022). However, the scale and complexity of LLMs require substantial computational resources, which can pose accessibility challenges. The extensive training necessary to develop these models often demands high-powered GPUs and significant electricity consumption, leading to greater operational costs.
A question remains whether such emergent abilities can be achieved at a smaller scale using strategic choices for training, data selection...

Small Language Models (SLMs) are scaled-down versions of their larger counterparts. They are designed to offer a more accessible and resource-efficient option

SLMs are advanced computational models designed to understand, generate, and manipulate human language in a structured manner. They are highly relevant to fields involving

language processing and machine learning, which are integral components of modern educational technologies.

The system leverages SLMs ensures that it is lightweight, efficient, and suitable for the integration. By integrating SLMs into systems like QAIP, the E-learning and Intelligent Educational Systems (ELSEI) program equips students with the skills to develop cutting-edge educational tools that improve teaching and learning outcomes through intelligent, adaptive, and efficient technologies. By undertaking this article, we aim to foster a culture of innovation and technological empowerment within our academic community. As we embark on this endeavor, we remain cognizant of the challenges ahead, yet emboldened by the prospect of effecting positive change in the landscape of AI research and development. We will prioritize the collection of necessary data, the training of a suitable model, and the deployment of the results. This approach will involve thorough data collection, careful model training, and smooth deployment to ensure that the final AI-driven question-answer system effectively and efficiently addresses the needs of student's preferences and learning outcomes.

2. LITERATURE REVIEW

2.1. Language Models in Education

As LLMs are gaining momentum for their potential to bolster learner engagement, academic achievement, and student success, particularly in natural language processing (NLP). These models are designed to understand, interpret, and generate human language with high sophistication, enabling a wide array of previously unattainable applications. From the perspective of evolution and capabilities, the development of LLMs has progressed significantly, evolving from rudimentary text processing systems to highly advanced models capable of intricate language understanding (Gattupalli et al.,2023.).

These models, characterized by deep learning architectures encompassing billions of parameters, capture intricate language patterns, deeply understand context, and generate coherent, contextually relevant text. LLMs operate through a process known as unsupervised learning, where they learn from extensive datasets comprising books, articles, websites, and other textual sources. The primary objective is for the model to predict the next word in a sentence based on preceding words (Denny et al., 2022). Repeated across billions of sentences, this task enables the model to learn the statistical properties of language, including grammar, syntax, and semantics. The core mechanism involves transforming input text into numerical vectors through tokenization, processed through multiple neural network layers, with attention mechanisms allowing the model to focus on relevant text parts, enhancing its contextual understanding.

GPT-3, released by OpenAI in May 2020, is a groundbreaking large language model that is trained to predict the next token in a text sequence (Brown et al., 2020). An enhanced iteration of ChatGPT, based on GPT-4, was introduced in March 2023, is the focus of our work and showcases even more advanced natural language generation, understanding, and learning capabilities (OpenAI et al., 2024). GPT-4's abilities are transforming not only learning but also higher education research. Its knack for answering queries, providing explanations, and assisting in problem-solving makes it an asset in creating interactive learning experiences.

Recent studies and reviews highlight the potential of LLMs to enhance learning and teaching experiences significantly (Gao et al., 2022; Rojas-Sanchez et al., 2023). However, research on the application of LLMs in education is still limited compared to their use in other sectors like finance, medicine, banking, and e-commerce (Salas-Pilco et al., 2022). A study by (Englhardt et al., 2024) findings suggest that while LLMs may play a role in fully automated code generation, perhaps more powerful is a new human-AI co-development approach to embedded systems development that combines human context and high-level design objectives

with LLM code generation. While LLMs are becoming increasingly indispensable in the digital age, they are not expected to replace teachers or instructors but rather serve as valuable assistants to support both educators and learners. These studies offer critical insights into human behavior, cognition, and motivation, providing a scientific foundation for enhancing educational practices and deepening our understanding of human development (Sarsa et al., 2022).

2.2 Small Language Models

Small Language Models (SLMs) are significantly less resource-intensive compared to Large Language LLMs. They require fewer computational resources, making them more accessible for institutions with limited budgets. Due to their smaller size, SLMs are cheaper to train and deploy. This can be particularly advantageous in educational settings where cost is a crucial factor. A study by Schick and Schütze (2021) demonstrated that smaller models like pattern-exploiting training (PET) can achieve competitive performance on specific tasks without the hefty computational cost associated with models like GPT-3.

SLMs enable faster iteration cycles during development. This means educators and researchers can more rapidly test and refine models to suit their specific needs. The smaller size of these models means they can be deployed more quickly, especially in environments with limited computational infrastructure. A case study on the deployment of smaller models in low-resource settings shows that SLMs like DistilBERT can be rapidly integrated into educational tools with minimal latency and infrastructure requirements (Sanh et al., 2019).

SLMs tend to have a simpler architecture, which can make them easier to interpret and understand. This is particularly beneficial in educational contexts where transparency is important for both instructors and students. Their simplicity also makes SLMs more user-friendly, allowing non-experts to engage with and utilize these models more effectively. The work of (Wang et al., 2019) highlighted how SLMs like BERT-mini are more interpretable for educational purposes, enabling educators to better understand how the model arrives at specific answers.

SLMs can be more easily fine-tuned for specific tasks or domains, providing better performance for niche applications without the need for extensive computational power. Their smaller size allows them to be adapted for specialized educational tools where LLMs might be overkill. In a study by (Grangier et al., 2024), an SLM was fine-tuned for use in a particular educational domain, outperforming a larger model that was not as easily adaptable.

NVIDIA's Mistral-NeMo-Minitron 8B represents a significant advancement in the development of small language models, achieving a balance between model efficiency and accuracy. By reducing the original Mistral NeMo model from 12 billion to 8 billion parameters through pruning and distillation techniques, the Minitron 8B maintains high performance across key benchmarks while requiring fewer computational resources. This makes it accessible for deployment on standard workstations and laptops, particularly in settings with limited resources. Additionally, its scalability and adaptability allow for further customization to meet specific application needs, such as use on mobile or embedded devices. These findings underscore the potential of small language models to provide state-of-the-art generative AI capabilities in a cost-effective and operationally efficient manner, broadening their applicability in both research and industry contexts (Briski, 2024).

SLMs represent a significant advance in the field of artificial intelligence. These are effective and versatile solutions that successfully compete with popular LLMs. Small language models are changing computing standards by cutting expenditures and simplifying architecture, confirming that size is not the main criterion determining productivity. Although limitations remain, e.g., difficulties in identifying context, active research in this industry will increase the effectiveness of tiny systems shortly.

3. METHODOLOGY

3.1. Data Collection and Preprocessing

For developing an effective Question Answering Introduction to Python (QAIP) system, a significant amount of text covering various topics is essential. The quality and diversity of this data directly impact the QA system's ability to provide accurate and relevant answers. Therefore, our data collection module is designed to gather a comprehensive and diverse dataset that ensures the model can learn and generalize well across different subjects. By incorporating text from a wide range of sources, we aim to create a robust foundation that supports the (QAIP) system in delivering precise and contextually appropriate responds.

3.1.1 Source of Training Data

To collect introductory programming course text for training our QA system, we decided to utilize *Intopython.wiki* website for several reasons. Firstly, *Intopython.wiki* contains a large amount of data, providing a rich resource for diverse training material. Thise website cover various topics, ensuring that our dataset encompasses a wide range of subjects, which is essential for creating a versatile and comprehensive QA system. Additionally, the language level of the content is medium, making it suitable for a broad audience while maintaining clarity and accessibility. The type of data available on *Intopython.wiki* is primarily informative, which aligns well with our goal of developing a QA system capable of delivering accurate and useful information. Finally, the website focus on topics ensures that the text is well structured and coherent, which is ideal for training language models. This combination of factors makes *Intopython.wiki* an excellent choice.

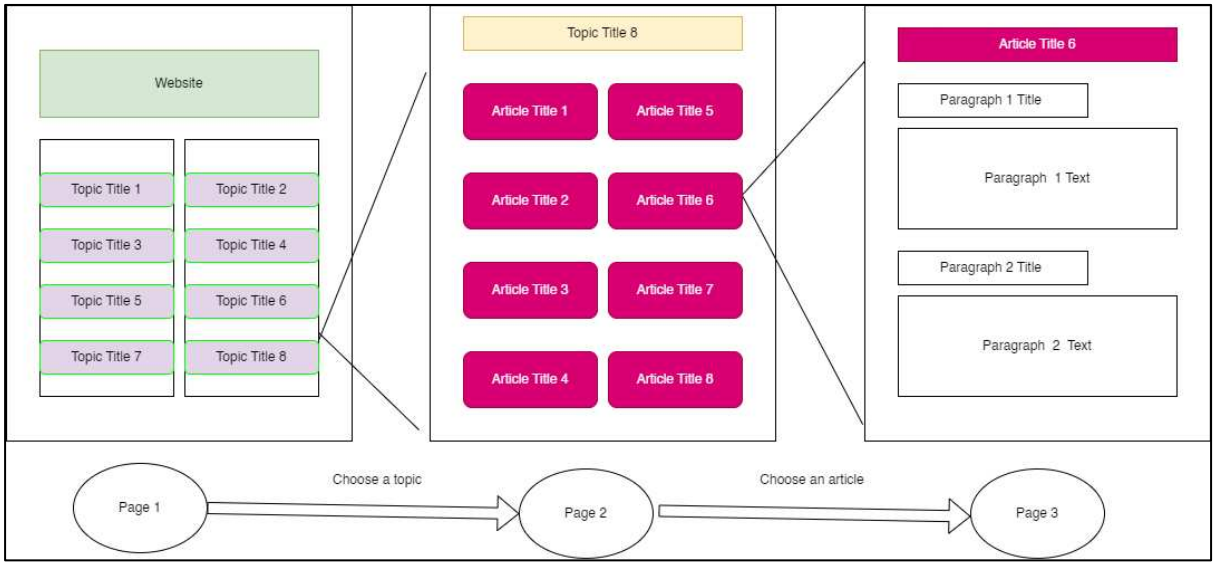

Figure 1. Website Structure

3.1.2. Implementation Strategy

The initial phase of our data collection involved extracting text from the *Intopython.wiki* website. Each resource addressed various subject areas. We accessed and read the content on the site, systematically extracting text from each resource page. This step was critical, as it established the groundwork for further analysis by identifying the main content in each resource. After successfully extracting the general text, we then focused on specific sections. For each identified section, we navigated to its corresponding pages. Each section typically provided detailed content on a particular subject area. Using a similar method, we read the

Br. J. Ed. Tech. Soc, v.17, n.se4, p.84-95, 2024
DOI http://dx.doi.org/10.14571/brajets.v17.nse4.84-95
ISSN 2316-9907

relevant pages and systematically extracted text from these sections, mapping out specific content for deeper analysis.

Our approach was methodical and comprehensive, starting broadly with general sections, then narrowing down to specific headings, and finally extracting detailed content within each section, ensuring a hierarchical and thorough data collection strategy.

3.2. Model Training

3.2.1. Environment

After completing data collection from the *Intopython.wiki* website, we moved to the modeling phase, starting with setting up the environment. We chose Google Colab and Google Drive due to their free accessibility and robust features. Google Colab provides powerful computational resources, including high-performance GPUs, which are essential for training and fine-tuning models, especially when dealing with kind of datasets extracted from website. This choice helps overcome the limitations of our local hardware, ensuring that the intensive computations required for model training are efficiently handled.

Integrating Google Colab with Google Drive enhances our workflow by enabling efficient data management. This setup allows for easy access, storage, and management of this type of datasets extracted from the website, facilitating smooth read and write operations. Mounting Google Drive in Colab replicates local filesystem operations, streamlining our workflow.

Google Colab's user-friendly interface and extensive support for Python libraries make it an ideal choice. Its ease of use and comprehensive documentation simplify the setup and maintenance of our development environment. This configuration maximizes our resources, ensuring effective training and fine-tuning of the model, leading to a robust and responsive system for processing and analyzing the content extracted from the website.

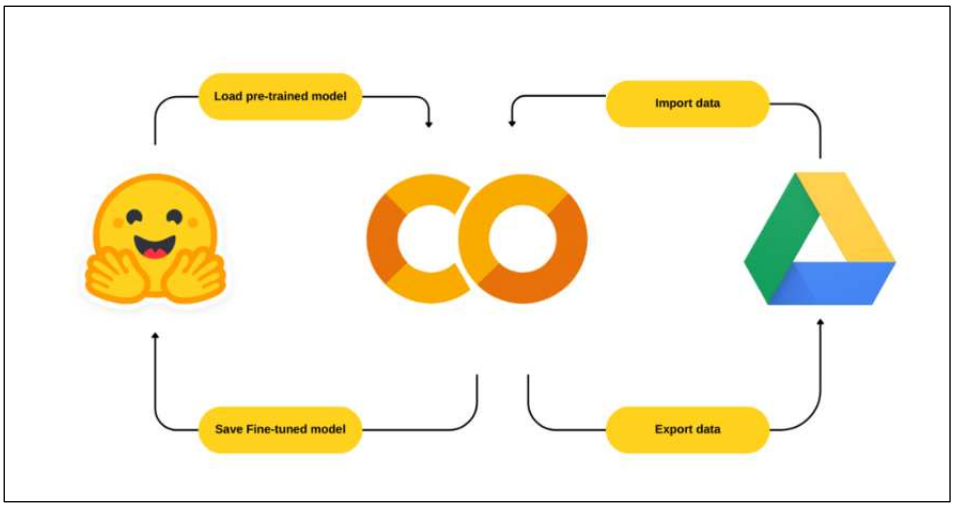

Figure 2. Environment Set Up

Additionally, we utilize Hugging Face, a premier platform for natural language processing (NLP) models and tools. Hugging Face provides a vast repository of pre-trained models and an intuitive interface for managing them. Leveraging Hugging Face allows us to efficiently load pre-trained models, fine-tune them on our dataset extracted from the *Intopython.wiki* website, and save the fine-tuned models back to the platform. This integration greatly enhances our workflow, enabling smooth transitions between model development, training, and deployment stages.

3.2.2. Fine-Turning of Small language Models

Fine-tuning is a vital step in developing and deploying Small Language Models, especially when using prebuilt models. It involves taking a pretrained model and making targeted adjustments to tailor its performance for a specific task or domain. This process is essential for harnessing the general capabilities of SLMs while customizing them to meet specific needs and enhancing their performance for particular applications.

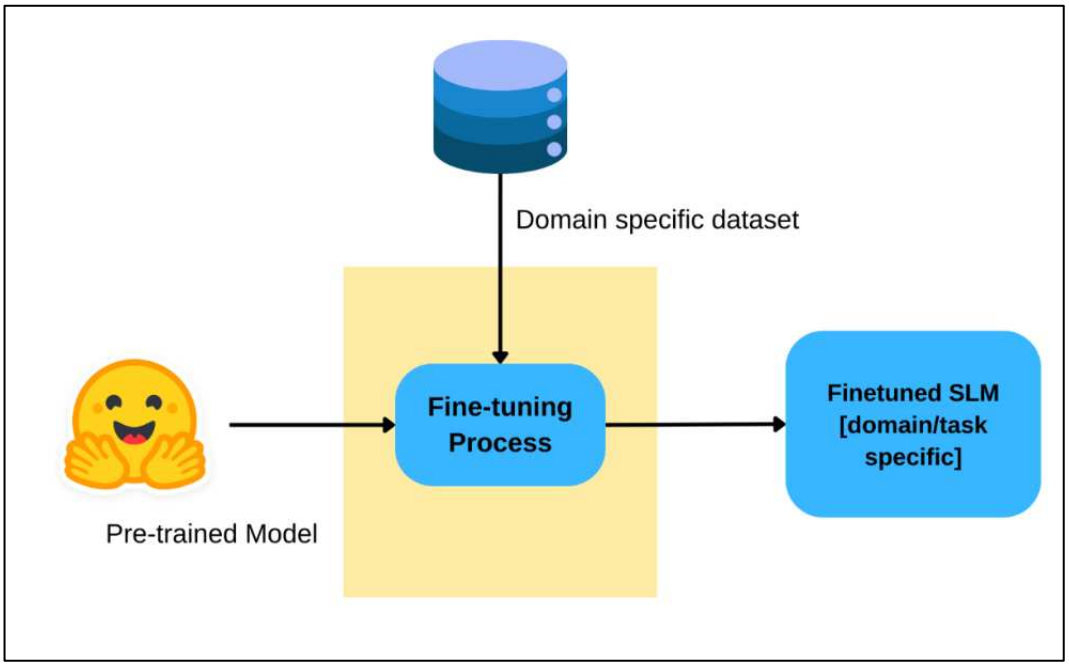

Figure 3. Fine-tuning

In the development of our Question Answering Introduction to Python (QAIP) system, careful consideration must be given to the selection of suitable language models. Given the constraints and requirements, we have established the following criteria to ensure optimal performance and feasibility:

- **Free to use:** The primary criterion for selecting a language model is its availability at no cost. As university students, our budget for the project is limited. Therefore, we need to leverage open-source models or those available under permissive licenses that allow for free usage in both development and deployment stages.
- **Reasonable Size:** Our computing resources are limited, with local machines that are not equipped with high-end processing power or extensive storage capabilities. Hence, it is imperative to select models that are of reasonable size, ideally within the range of 4-5 GB. This ensures that the models can be easily downloaded, stored, and processed on our local machines without significant performance degradation or storage issues.
- **Good Quality Speed Rate:** The chosen models must also exhibit a good quality speed rate. This involves balancing the trade-off between model size and processing speed to achieve an efficient performance. The models should be capable of providing quick responses to user inputs without compromising on the accuracy or relevance of the generated responses. Ensuring a good quality speed rate is critical for maintaining a smooth and responsive user experience with the (QAIP) system.

3.2.3. Deployment Architecture

As we reach the final piece of the deployment process, it is essential to outline the architecture that facilitates the interaction between the user, front-end interface, and backend systems. The following description details the flow of our deployment architecture, ensuring efficient and accurate response generation by the QA system. The architecture for deploying our (QAIP) system is designed to ensure efficient interaction between the user, front-end interface, and back-end systems. The flow of this architecture is detailed as follows:

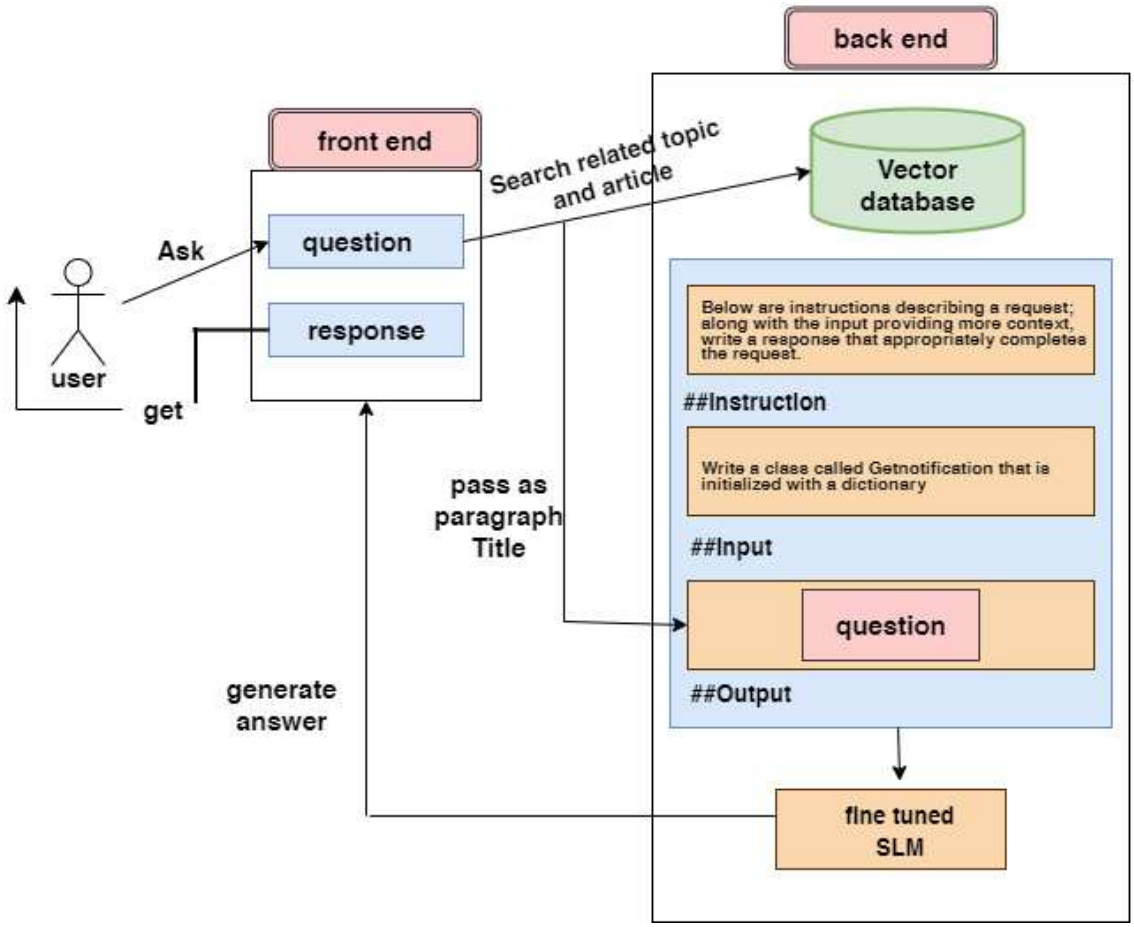

Figure 4. Application Architecture

The diagram illustrates the flow of a system designed to process user questions and generate answers using a vector database and a fine-tuned model. Here is a detailed description:

- User Interaction:
  - The user asks a question.
  - The front end receives the question and passes it for processing.
- Front End:
  - The question is sent to the backend.
  - After processing, the front end receives the response and displays it to the user.
- Back End:
  - The backend is responsible for handling the search and response generation.
  - The question is used to search for related topics and articles in a vector database.

- The backend contains a section with instructions, explaining the inputs needed:
  - Processing:
    - The relevant topic title, article title, and question are passed to the fine-tuned LLM.
    - The LLM processes these inputs to generate an answer.
  - Output:
    - The generated answer is passed back to the front end.
    - The front end displays the answer to the user.

4. RESULTS AND DISCUSSIONS

4.1. A Statistical Perspective

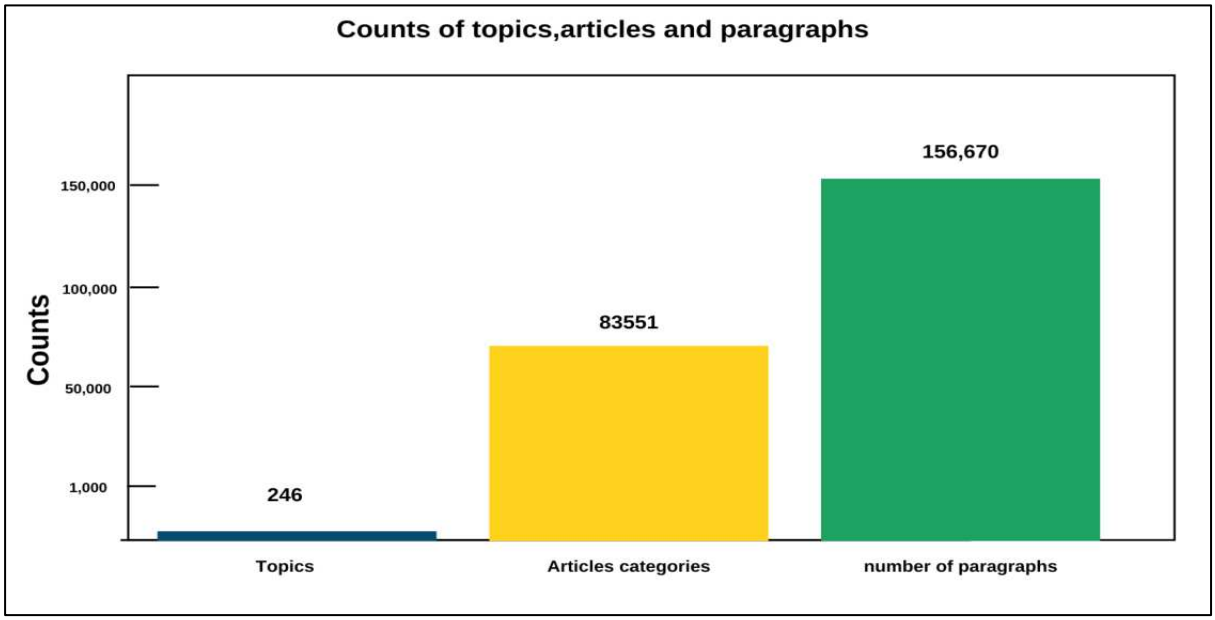

Figure 5. Word Count Distribution Across Topics, Articles, and Paragraphs

After scraping data from the website, we obtained the following distribution of data: The number of topics titles is 246, the number of articles titles is 83,551, and the number of paragraphs is 156,670.

This distribution reflects the diversity of our dataset, as the topics cover various subject. This ensures that our QA system, when fine-tuned, will be well-distributed across diverse subject areas. Additionally, the substantial number of articles indicates that our dataset is largely sufficient in terms of quantity. The quality of the dataset will be evaluated in the subsequent statistics.

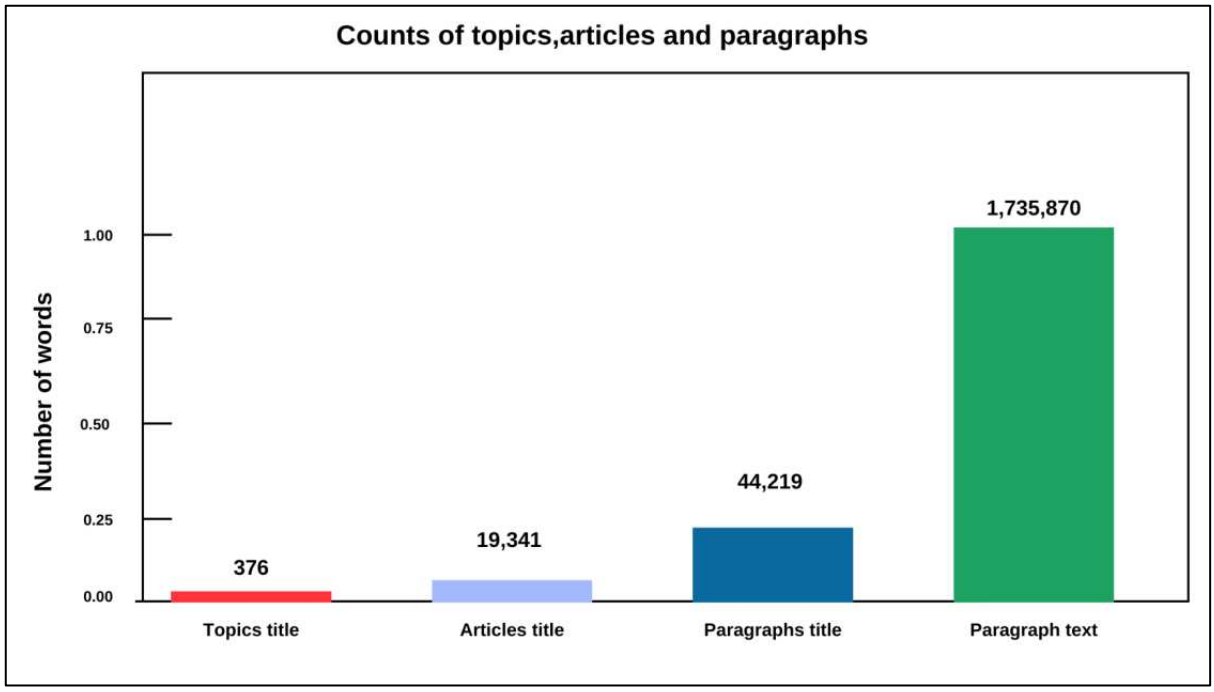

Figure 6. Word Count Distribution Across Topics, Articles, and Paragraphs

The scraped data consists of 376 different words used in topic title 19,341 words in article title, 44,219 words in paragraph title, and 1,735,870 words in paragraph text. This distribution highlights the extensive vocabulary captured across different sections of the dataset.

In the 300 topics, we have only 376 different words used, which reflects the oversimplicity and rearrangement of words to create new topics. A similar pattern is observed in the article titles, where 19,341 words are used. This indicates that while the data is quite compact and the vocabulary is relatively simple, which can help the model to learn basic patterns and combinations, there is a risk of oversimplification. We will explore this further in the modeling phase to understand its impact on the model's performance.

4.2. Deployment perspective

This SLM was created primarily to test out our model's performance in a real-world scenario. It serves as a functional prototype to evaluate how well the model handles user queries and provides responses. Overall, the design is clean and user-friendly, facilitating efficient interaction with the system while we assess its capabilities.

To evaluate the functionality, we will test several questions across various domains.

Overall, the system has the potential to deliver good answers, particularly when the topic is well-defined and pressing. The accuracy and relevance of the information presented indicate that the system can effectively retrieve and summarize pertinent data. While the content of the responses is generally accurate, there are areas where the structure and coherence of the sentences can be improved. The following points highlight the key observations:

- **Organization of Information:** The responses tend to provide a list of suggestion without clear transitions between points. A more organized approach, with clear headings and subheadings, could enhance readability and coherence.
- **Sentence Structure:** Some sentences lack proper flow and connectivity, which can make the information appear disjointed. Ensuring that each sentence logically follows from the previous one would improve the overall coherence.
- **Detailed Explanations:** le the list of suggestion is informative, adding brief explanations for why each suggestion is highly ranked would provide more context and depth. This would also help in maintaining the reader's engagement.

5. CONCLUSION

The development and integration of Small Language Models (SLMs) represent a significant advancement in enhancing accessibility and inclusivity in AI within the context of the E-learning and Intelligent Educational Systems (ELSEI) Master Program. This work addresses the existing gap in AI tools within the ELSEI program, particularly for chatbots, by systematically collecting data, fine-tuning advanced models, and implementing a user-friendly web interface.

Data collection from *Intopython.wiki* provided a diverse dataset, despite some limitations. Advanced scraping techniques and strategic data storage established a solid foundation for effective model training. Overcoming challenges related to resource constraints involved using Google Colab and optimizing model parameters and dataset size. The integration of a vector database further enhanced the system's functionality, offering efficient and accurate search results.

The final web application, while basic in design, demonstrated the practical utility of the model by delivering linguistically accurate responses. However, there is room for improvement in terms of factual accuracy and response coherence. Future enhancements could include integrating Retrieval-Augmented Generation (RAG) with a free API of pretrained SLMs, which may offer smoother performance and better computational efficiency. Additionally, diversifying the dataset by incorporating additional sources could enrich the model's learning experience and provide a broader range of content.

From an educational perspective, SLMs offer substantial benefits in teaching and learning by providing accurate and efficient support for Python-related queries. The QAIP system can be seamlessly integrated into teaching practices to deliver instant assistance, personalized learning experiences, and improved engagement in programming education. This integration not only enhances teaching efficacy but also fosters a more interactive and accessible learning environment, ultimately improving student outcomes and making complex topics more manageable.

REFERENCES

Bressane, A., Zwirn, D., Essiptchouk, A., Saraiva, A. C. V., Carvalho, F. L. de C., Formiga, J. K. S., Medeiros, L. C. de C., & Negri, R. G. (2024). Understanding the role of study strategies and learning disabilities on student academic performance to enhance educational approaches: A proposal using artificial intelligence. Computers and Education: Artificial Intelligence, 6(100196), 100196. https://doi.org/10.1016/j.caeai.2023.100196

Briski, K. (2024, August 21). Lightweight Champ: NVIDIA Releases Small Language Model With State-of-the-Art Accuracy. NVIDIA Blog. https://blogs.nvidia.com/blog/mistral-nemo-minitron-8b-small-language-model/

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei, D. (2020). *Language Models are Few-Shot Learners* (arXiv:2005.14165). arXiv. http://arxiv.org/abs/2005.14165

Chen, Y., Ding, N., Zheng, H.-T., Liu, Z., Sun, M., & Zhou, B. (2023). Empowering Private Tutoring by Chaining Large Language Models. ArXiv (Cornell University). https://doi.org/10.48550/arxiv.2309.08112

Denny, P., Kumar, V., & Giacaman, N. (2022). *Conversing with Copilot: Exploring Prompt Engineering for Solving CS1 Problems Using Natural Language* (arXiv:2210.15157). arXiv. https://doi.org/10.48550/arXiv.2210.15157

Englhardt, Z., Li, R., Nissanka, D., Zhang, Z., Narayanswamy, G., Breda, J., Liu, X., Patel, S., & Iyer, V. (2024). Exploring and Characterizing Large Language Models for Embedded System Development

and Debugging. *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems*, 1–9. https://doi.org/10.1145/3613905.3650764

Gattupalli, S., Lee, W., Allessio, D., Crabtree, D., Arroyo, I., & Woolf, B. (n.d.). *Exploring Pre-Service Teachers' Perceptions of Large Language Models-Generated Hints in Online Mathematics Learning*.

Grangier, D., Katharopoulos, A., Ablin, P., & Hannun, A. (2024). *Specialized Language Models with Cheap Inference from Limited Domain Data* (arXiv:2402.01093; Version 1). arXiv. https://doi.org/10.48550/arXiv.2402.01093

Kurdi, G., Leo, J., Parsia, B., Sattler, U., & Al-Emari, S. (2019). A Systematic Review of Automatic Question Generation for Educational Purposes. International Journal of Artificial Intelligence in Education, 30(1), 121–204. https://doi.org/10.1007/s40593-019-00186-y

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., … Zoph, B. (2024). *GPT-4 Technical Report* (arXiv:2303.08774). arXiv. http://arxiv.org/abs/2303.08774

Sarsa, S., Denny, P., Hellas, A., & Leinonen, J. (2022). Automatic Generation of Programming Exercises and Code Explanations using Large Language Models. *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 1*, 27–43. https://doi.org/10.1145/3501385.3543957

Wang, Z., Ng, P., Ma, X., Nallapati, R., & Xiang, B. (2019). Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 5878–5882). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1599