VTU-GPT: A Syllabus based Chatbot

Mrs.Vijaylaxmi Inamdar[1], Tejas SP[2], Sneha V[3], Shashank K Murthy[4], Varnith Aryan[5].
[1]*Assistant Professor, Department of Computer Science and Engineering*
[2,3,4,5]*UG students, Department of Computer Science and Engineering*
[1,2,3,4,5] *Dayananda Sagar Academy of Technology and Management, Bangalore, Karnataka India*

Abstract*— VTU-GPT (Visvesvaraya Technological University - Generative Pre-trained Transformer) is a meticulously crafted chat interface designed to address academic challenges at Visvesvaraya Technological University (VTU). Serving as a centralized hub, it provides VTU students with a seamless space for precise answers, doubt resolution, and access to syllabus-specific information. Utilizing Retrieval-Augmented Generation, VTU-GPT delivers concise explanations and furnishes pertinent links and accessible PDF resources. This paper explores the development, implementation, and impact of VTU-GPT, emphasizing its role in streamlining and enhancing the academic journey for VTU students.

Keywords*— Large Language Models (LLMs), Machine Learning, Natural Language Processing (NLPs), Retrieval Augmented Generation.

I. INTRODUCTION

Students in higher education frequently face a lot of obstacles along the way that call for creative solutions. The goal of this study is to address a pressing problem that Visvesvaraya Technological University (VTU) students confront, which is the lack of a specialized platform that makes it easier for students to access resources designed specifically for VTU. Students encounter additional challenges in locating trustworthy sources for effective preparation because necessary study materials are scattered across various platforms. The root cause of this problem is the lack of a specialised platform that can easily combine educational materials unique to VTU; this shortcoming is further highlighted by the fact that recommended readings are dispersed throughout multiple platforms.

The main objective of this research project is to present a game-changing approach to mitigate VTU students academic challenges [5]. Students' preparation becomes less effective as they struggle with a vast number of information sources and inaccurate sources dispersed across multiple platforms, and the process becomes a complex web of searching and cross-referencing. VTU-GPT is a proposed platform that carefully follows the VTU syllabus by creating a customized chat-like interface using Retrieval-Augmented generation technology. Specifically designed to match the VTU syllabus, this chat-like interface aims to offer relevant links and easily accessible PDF resources, personalized explanations, which stem from the recognition of the difficulties students encounter in obtaining reliable and comprehensive study materials. It seeks to bring about a fundamental shift in the way students interact with academic content, rather than merely a technical advancement. This single platform solution's goal is to bridge the gap between platforms that are specially designed to satisfy VTU's unique educational requirements. VTU-GPT can improve student's overall academic experience by addressing problems resulting from dispersed resources. Students can seamlessly navigate through the VTU syllabus, ensuring they receive accurate and personalized support in real-time.

With a specific focus on addressing the academic challenges faced by students at Visvesvaraya Technological University (VTU), the proposed VTU-GPT system represents a targeted evolution beyond current GPT models. Building on the foundation of natural language understanding, similar to models such as GPT-3, VTU-GPT aims to be an advanced tool that can understand and produce text relevant to the VTU syllabus in a [6] human-like manner. The proposed system aims to establish a centralized platform, enabling VTU students to effortlessly reach educational materials, find responses to academic questions, and address uncertainties. It builds upon the diverse capabilities of GPT models. Personalization remains crucial, allowing for the adjustment of VTU-GPT's behavior to seamlessly fit

the specific academic context of VTU. VTU-GPT emphasizes responsible usage above all else, acknowledging the limitations of current GPT models and the possibility of occasional incorrect information generation and input phrasing sensitivity. The system under consideration is designed to undergo ongoing improvement and development. It anticipates updates that will rectify any shortcomings and improve its functionality, making it a valuable resource for VTU students as they pursue their education.

Using Next.js and Express.js for frontend and backend development, respectively, and MongoDB for data persistence, this research aims to create a comprehensible and scalable web application with generative AI capabilities. To enhance the language generation capabilities of the system, we incorporate Django and the Django REST Framework to create API endpoints that enable communication with a generative AI model based on Python that is powered by HuggingFace and Langchain. This study investigates how to improve natural language generation and understanding in the application by using open-source large language models (LLMs), namely llama2 7B and Mistral 7B. The suggested architecture is implemented on AWS EC2, utilising AWS S3 for effective data storage and containerization enabled by Docker. This ensures flexibility and scalability in the implementation of this cutting-edge web application.

The successful implementation and integration of VTU-GPT is the expected result of this research, which will improve VTU students' academic experiences. The platform wants to be more than just an information store; it envisions becoming a flexible, dynamic tool tailored to the individual requirements of VTU students. With the help of this revolutionary platform, we hope to completely change the way that students interact with course material and create a more encouraging and stimulating learning environment for all members of the VTU community.

In conclusion, Visvesvaraya Technological University students confront a variety of challenges, and the VTU-GPT project offers a comprehensive response to these issues. This study aims to address the shortcomings in the current educational system and help build a platform that is efficient and compassionate, enabling students to pursue their academic goals within the special environment of VTU.

II. TECHNOLOGICAL FRAMEWORK

A. Mistral 7B

In the dynamic domain of Natural Language Processing (NLP), the pursuit of heightened model performance often necessitates larger models, escalating computational costs and inference latency [1]. Addressing this challenge, Mistral 7B is introduced as a meticulously designed language model demonstrating high performance coupled with efficient inference. Leveraging *Grouped-Query Attention* (GQA) and *Sliding Window Attention* (SWA), Mistral 7B achieves accelerated inference speed and reduced memory requirements. The model showcases enhanced efficiency, particularly in real-time applications, by effectively handling longer sequences at a reduced computational cost.

i. Sliding window attention

Sliding Window Attention (SWA) leverages the transformer's stacked layers to extend information access beyond a window size W. In this approach, the hidden state at position i in layer k, hi, attends to preceding hidden states within the range of i − W to i. This recursive process enables hi to [1] access tokens from the input layer up to a distance of W × k tokens. With a window size of W = 4096 at the final layer, there's a theoretical attention span of approximately 131K tokens. Practically, for a sequence length of 16K and W = 4096, modifications inspired by FlashAttention and xFormers yield a 2x speed improvement over a standard attention baseline.

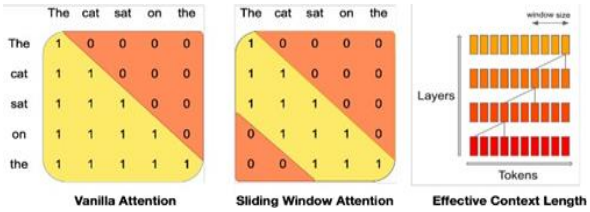

Figure 1: Stacked layers Sliding windown attention

ii. Rolling Buffer Cache

The Rolling Buffer Cache employs a fixed attention span, allowing for a controlled cache size. With a cache of size W, the keys and values at timestep i are stored at position i mod W in the cache. [1] As i

exceeds W, previous values get overwritten, capping the cache size. For instance, with W = 3, on a sequence of 32k tokens, this approach reduces cache memory usage by 8x without compromising model quality.

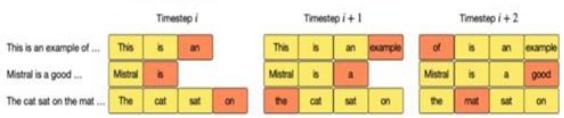

Figure 2: Cache Memory Usage

iii. Pre-fill and Chunking

When generating a sequence, we need to predict tokens one-by-one, as each token is conditioned on the previous ones. However, [1] the prompt is known in advance, and we can pre- fill the (k, v) cache with the prompt. If the prompt is very large, we can chunk it into smaller pieces, and pre-fill the cache with each chunk.

B. LangChain

LangChain is a framework to develop applications powered by LLMs with composability and reliability. LangChain has become the go-to tool for AI developers around the world to build generative AI applications. LangChain is used for connecting the external data sources and integration with many LLMs [2] available on the market. Apart from this,the main feature which is being used in this projectis that LLM-powered apps require a vector storage database to store the data they will retrieve later on.

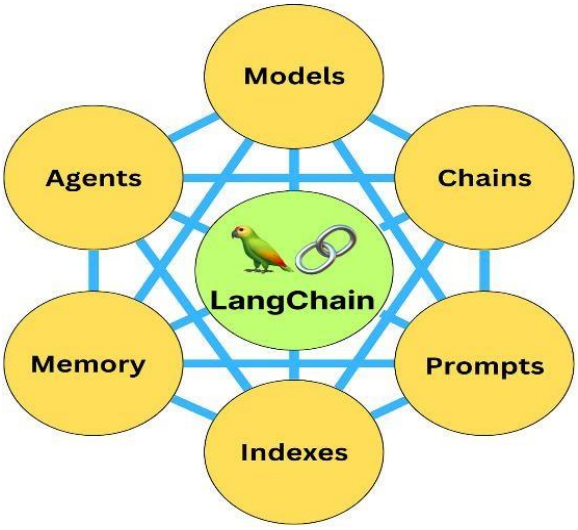

Figure 3: Components of LangChain

LangChain has six main components to build LLM applications: model I/O, Data connections, Chains, Memory, Agents, and Callbacks. The framework also allows integration with many tools to develop full-stack applications, such as OpenAI, Huggingface Transformers, and Vectors stores like Pinecone and chromadb, among others. The main value propositions of the LangChain are:

- *Components:* These are the abstractions needed to work with language models. Components are modular and easy to use for many LLM use cases.

C. Off-the-shelf chains: A structured assembly of various components and modules to accomplish a specific task, such as summarization, Q&A, etc.

HuggingFace

Hugging Face, a prominent machine learning and data science platform, supports users in building, deploying, and training ML models. It functions as a GitHub-like space for transparent collaboration and testing. Renowned for its Transformers Python library, Hugging Face simplifies model downloading and training, [3] offering developers an efficient way to integrate ML models into their workflows. The platform's open-source nature and deployment tools enable users to openly share resources, reducing model training time, resource usage, and the environmental impact of AI development.

- Fine-tune models: Users can fine-tune and train deep learning models using Hugging Face's application programming interface (API) tools.
- Host demos: Hugging Face lets users create interactive, in-browser demos of machine learning models. This lets users showcase and test models more easily.
- Research: Hugging Face has been involved in collaborative research projects, like BigScience research workshop, aiming to advance the field of NLP.

Major Features:

i. *Models.* Hugging Face hosts a large library of models that users can filter by type. There are more than 300,000 models on Hugging Face. Hugging Face also hosts the top open source ML models on the platform. The models on the leaderboard at the time of this writing include the following:
   - stabilityai/stable-diffusion-xl-base-1.0.
   - WizardLM/WizardCoder-Python-34B-V1.0.
   - Phind/Phind-CodeLlama-34B-v2.

ii. *Data sets*. Data sets help train models to understand patterns and relationships between data and creating a good data set can be

difficult. Hugging Face provides access to data sets uploaded by the community that users can access. Some example data sets in the Hugging Face library include the following:

- the_pile_books3, which contains all data from Bibliotik in plain text. Bibliotik is a repository of 197,000 books.
- wikipedia, which contains data from Wikipedia and more.

iii. *Spaces:* Machine learning models on their own typically require technical knowledge to implement and use. Spaces packages models in a user-friendly experience that lets users showcase their work. Hugging Face provides the computing resources necessary to host demos. Spaces doesn't require any technical knowledge to use. Some examples of Hugging Face Spaces include the following:

- LoRA the Explorer image generator. Users can generate images in different styles.
- MusicGen music generator. MusicGen lets users generate music.
- Image to Story. Users can upload an image, and a large language model uses text generation to write a story based on it.

D. Retrieval-augmented generation

Retrieval-augmented generation (RAG) is an AI framework for improving the quality of *LLM-generated* responses by grounding the model on external sources of knowledge to supplement the LLM's internal representation of information. Implementing RAG in an LLM-based question answering system has two main benefits: [4] It ensures that the model has access to the most current, reliable facts, and that users have access to the model's sources, ensuring that its claims can be checked for accuracy and ultimately trusted.

RAG has additional benefits. By grounding an LLM on external, verifiable facts, the model has fewer opportunities to pull information baked into its parameters. This reduces the chances that an LLM will leak sensitive data, or 'hallucinate' incorrect or misleading information.

RAG also reduces the need for users to continuously train the model on new data and update its parameters as circumstances evolve. RAG can lower the computational and financial costs of running LLM-powered chatbots in an enterprise setting.

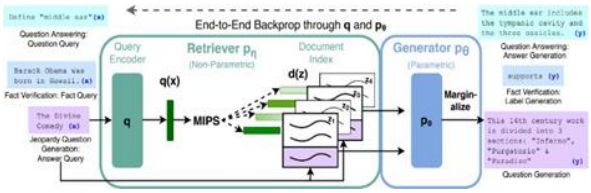

Figure 4: Working of Retrieval Augmented Generation

Examining RAG models, the utilization of input sequence (x) is directed towards retrieving text documents (z) to enhance context for generating the target sequence (y). Figure 1 illustrates two primary components within the models: (i) a retriever ($p\eta(z|x)$) providing distributions over text passages based on the query (x), and (ii) a generator ($p\theta(yi\ |x, z, y1{:}i{-}1)$). The generator generates each token using context from preceding tokens ($y1{:}i{-}1$), the original input (x), and a retrieved passage (z). To achieve end-to-end training for the retriever and generator, the retrieved document is treated as a latent variable. Two distinct models, RAG-Sequence and RAG-Token, introduce variations in incorporating latent documents to produce a text distribution. RAG-Sequence predicts each target token using the same document, whereas RAG-Token allows predictions based on different documents for each target token. Both models are formally presented, outlining details on the retriever ($p\eta$) and generator ($p\theta$), alongside training and decoding procedures.

III. PROPOSED WORK

It involves declaring and implementing a computer or programming process. Different executions or implementations may exist for a specification or standard.

1. Project Definition and Scope:

Objective: The primary objective of the "vtu-gpt" project is to create an intelligent conversational agent catering to the educational needs of VTU students. The system aims to provide personalized academic support, seamless access to educational resources, and an efficient user interface for enhanced interactions.

Scope: The conversational agent will be designed to address specific academic challenges faced by VTU students, utilizing state-of-the-art technologies for effective communication and resource provision.

2. Model Selection and Setup:
Choose GPT Model: Select a pre-trained GPT model based on project requirements. GPT-3, known for its advanced language understanding capabilities, is a prime candidate.
Install Dependencies: Establish the development environment with Next.js, Express.js, and MongoDB for web application development. Set up Django and Django REST Framework for API endpoint creation. Incorporate Python, Langchain, and HuggingFace for the generative AI model.

3. Data Collection and Preprocessing:
Define Training Data: Curate or generate a dataset specifically tailored to VTU's academic context. Ensure the dataset aligns with the input and output requirements of the chosen GPT model.
Preprocess Data: Clean and format the dataset to enhance its compatibility with the GPT model, optimizing the training process.

4. Model Fine-tuning:
Fine-tuning Consideration: Evaluate the need for fine-tuning the pre-trained GPT model on the custom dataset to improve its performance in addressing VTU-specific queries.

5. Conversation Loop Implementation:
User Input Handling: Develop a robust mechanism to receive user input, supporting interactions through a command line or a user-friendly web interface.
Tokenization: Utilize the GPT model's tokenizer to break down user input into [7] manageable components.
Model Inference: Leverage the pre-trained GPT model to generate responses based on tokenized input.

6. Web Interface:
Integration: Integrate Django to create an intuitive web interface, incorporating Next.js for dynamic and responsive web application development. Ensure seamless user interaction and navigation.

7. Ethical Considerations:
Bias Analysis: Implement systematic bias analysis on model responses to identify and mitigate potential biases and ethical concerns.
Implement Safeguards: Integrate mechanisms to filter and handle inappropriate or biased responses, emphasizing user safety and trust.

8. Security Measures:
Input Validation: Implement robust input validation to prevent malicious inputs and maintain the integrity of user interactions.
User Privacy: Enhance data privacy using the AES algorithm, ensuring secure handling of sensitive user information.

9. Continuous Monitoring and Updates:
Monitoring: Establish a continuous monitoring system to track the model's performance over time, identifying areas for improvement.
Update Mechanism: Plan for periodic model updates based on user feedback, new data, or evolving educational requirements.

10. Documentation:
Code Documentation: Document the codebase comprehensively, including functions, classes, and essential variables.
Project Documentation: Create comprehensive documentation covering project objectives, model details, and usage guidelines for developers and users.

11. Testing and Evaluation:
Unit Testing: Implement thorough unit tests to ensure individual components function as expected.
Evaluation Metrics: Define metrics for evaluating the model's performance, emphasizing response coherence, relevance, and overall user satisfaction.

12. Deployment:
Choose Deployment Platform: Opt for AWS EC2 instances for deployment, utilizing AWS S3 for efficient data storage. Implement Docker for containerization to ensure flexibility and ease of deployment.
Deployment Strategy: Plan the deployment process, considering scalability and resource requirements.

13. User Feedback:
Feedback Mechanism: Implement a user-friendly mechanism for users to provide feedback on the conversational agent's responses.
Iterative Improvement: Use user feedback to

iteratively enhance the model, addressing shortcomings and ensuring continuous improvement. This proposed system integrates cutting-edge technologies and methodologies to create an advanced conversational agent tailored to the unique educational needs of VTU students.

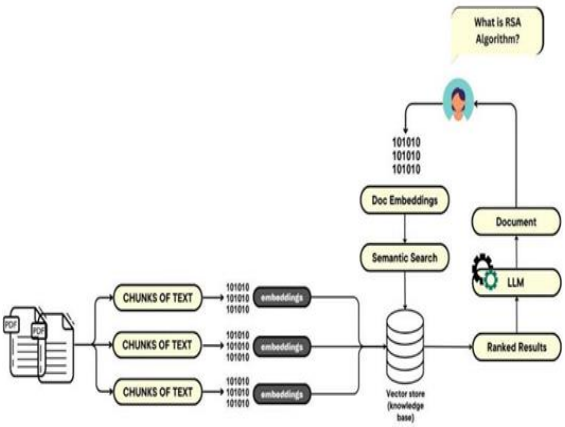

Figure 5: Approach to the Solution

IV. DISCUSSION OF SURVEYED WORKS

The survey aimed to assess the potential success and value of the "VTU-GPT" project in addressing the challenges faced by students and teachers in the VTU (Visvesvaraya Technological University) community. The objective was to understand the difficulties encountered while seeking information online and to gauge the demand for a conversational agent like "VTU-GPT"

Methodology:

i. Survey Distribution:

The survey was distributed using Google Forms. Participants included students and teachers from VTU across various disciplines.

ii. Survey Components:

The survey consisted of questions related to:

- Difficulty in Finding Information: Assessing the challenges faced by respondents in finding accurate and timely information online.
- Use of Educational Resources: Exploring the usage patterns of existing educational resources.
- Interest in a Conversational Agent: Gauging the interest and perceived value of having a conversational agent for academic queries.

Findings:

i. Difficulty in Finding Information:

- Result: A significant number of respondents reported difficulties in promptly finding accurate answers to academic queries.
- Insight: Many respondents expressed frustration with the time-consuming process of sifting through vast online resources.

ii. Use of Educational Resources:

- Result: The survey revealed that a considerable portion of participants relies on various online resources for academic support.
- Insight: Despite the availability of diverse resources, respondents faced challenges in discerning the most relevant and accurate information.

iii. Interest in a Conversational Agent:

- Result: A high percentage of respondents expressed interest in a conversational agent specifically tailored to address VTU-related queries.
- Insight: There is a clear demand for a solution that can provide immediate and accurate answers to academic questions.

Implications for "VTU-GPT":

i. User Need Validation:

The survey results validate the identified need for a tool like "VTU-GPT" to find accurate academic information.

ii. Time Efficiency:

Respondents highlighted the time-consuming nature of current search methods, indicating the potential for "vtu-gpt" to enhance efficiency.

iii. Interest and Adoption:

The high level of interest suggests a positive reception and potential adoption of "vtu-gpt" among the VTU student and teacher community.

The survey results underscore the importance of the "VTU- GPT" project in addressing the challenges faced by the VTU community in accessing timely and accurate academic information. The positive response indicates a strong potential for user adoption and emphasizes the project's relevance in enhancing the educational experience for students and teachers alike.

V. CONCLUSION

In conclusion, the VTU-GPT project has successfully demonstrated its capability to provide versatile task adaptation and a nuanced understanding of the model's functionalities. Ethical considerations were a primary focus, with implemented measures to mitigate biases, ensuring a responsible and inclusive user experience. The collaborative development approach fostered a holistic perspective, addressing specific academic challenges faced by Visvesvaraya Technological University (VTU) students.

While the project showcased the potential of GPT in Natural Language Processing (NLP), it also acknowledged challenges in interpretability and resource optimization. Looking ahead, future exploration will delve deeper into interpretability enhancements and staying attuned to advancements in transformer models. This project serves as a notable example of leveraging advanced AI technologies to enhance educational support while underscoring the importance of collective efforts in navigating the complexities of AI for the benefit of the academic community.

ACKNOWLEDGMENT

The authors extend sincere gratitude to all contributors to the successful completion of this research paper on VTU-GPT. Special thanks to Ms. Vijaylaxmi Inamdar, our Guide, for invaluable guidance and unwavering support. We appreciate the Computer Science and Engineering department for providing a conducive academic environment. The engagement and feedback from Visvesvaraya Technological University (VTU) students were pivotal in shaping the VTU- GPT platform.

Heartfelt thanks to friends and family for their encouragement during the demanding research phases. This endeavor wouldn't have been possible without the collaborative efforts of all mentioned above, and for that, we are truly grateful.

REFERENCE

[1] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven LeScao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, William El Sayed, *"Mistral 7B",* arXiv:2310.06825 [cs.CL].

[2] Ben Auffarth, *"Generative AI with Langchain",* chapter 2.

[3] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, Yueting Zhuang,*" HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face",* arXiv:2303.17580 [cs.CL].

[4] Patrick Lewis, Ethan Perez, Aleksandra Piktus, *"Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks",* arXiv: 2005.11401 [cs.CL].

[5] Bruno Silva, Leonardo Nunes, Roberto Estevão,*" GPT-4 as an Agronomist Assistant? Answering Agriculture Exams Using Large Language Models",* arXiv:2005.11401 [cs.CL].

[6] Yann Hicke, Anmol Agarwal, Qianou Ma, Paul Denny,*" ChaTA: Towards an Intelligent Question-Answer Teaching Assistant using Open-Source LLMs ",* arXiv:2311.02775 [cs.LG].

[7] Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, *"8192-Token General-Purpose Text Embeddings for Long Documents"*,arXiv:2310.19923 [cs.CL].

[8] Yijia Zhang, Sicheng Zhang, Shijie Cao*," AFPQ: Asymmetric Floating Point Quantization for LLMs",* arXiv:2311.01792 [cs.CL].

[9] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, RanganMajumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang, *"MS MARCO: A Human Generated Machine Reading Comprehension Dataset",* arXiv:1611.09268 [cs], November 2016. URL http://arxiv.org/abs/1611.09268. arXiv: 1611.09268.

[10] Petr Baudiš and Jan Šedivy, "Modeling of the question answering task in the yodaqa system", In International Conference of the Cross-Language Evaluation Forum for European Languages, pages 222–228. Springer, 2015. URLhttps://link.springer.com/chapter/10.1007 % 2F978-3-319-24027-5_20.

[11] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang, "Semantic Parsing on Freebase from Question-Answer Pairs", In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D13-1160.

[12] Bin Bi, Chenliang Li, Chen Wu, Ming Yan, and Wei Wang, "Palm: Pre-training an autoencoding & autoregressive language model for context-conditioned generation", ArXiv, abs/ 2004. 07159,2020. URL https://arxiv. org/abs/ 2004. 07159.

[13] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes,"Reading Wikipedia to Answer Open-Domain Questions", In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pages 1870–1879, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/ v1/P17-1171. URLhttps://www.aclweb.org/ anthology/P17- 1171.

[14] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang, *"Sparks of artificial general intelligence: Early experiments with gpt-4"*, 2023.

[15] Ranveer Chandra, Swami Manohar, Tusher Chakraborty, Jian Ding, Zerina Kapetanovic, Peeyush Kumar, and Deepak Vasisht, *"Democratizing data-driven agriculture using affordable hardware",* IEEE Micro, 42(1):69–77, 2022.URL https://www.microsoft.com/en-us/research/publication/ democratizing-data-driven-agriculture-using- affordable-hardware/.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, *"Bert: Pre-training of deep bidirectional transformers for language understanding",* 2019,arXiv:1810.04805.

[17] Semire Dikli. An overview of automated scoring of essays,*"The Journal of Technology, Learning and Assessment",* 5(1), 2006.

[18] S. J. Pan and Q. Yang, ''*A survey on transfer learning*,'' IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Jan. 2021.

[19] F. Li, Y. Jin, W. Liu, B. P. S. Rawat, P. Cai, and H. Yu, *''Fine-tuning bidirectional encoder representations from transformers (BERT)—Based models on large-scale electronic health record notes: An empirical study,''* JMIR Med. Informat., vol. 7, no. 3, Sep. 2019, Art. no. e14830.

[20] L. Ehrlinger and W. Wöß, *''Towards a definition of knowledge graphs,''* SEMANTICS (Posters, Demos, SuCCESS), vol. 48, nos. 1– 4, p. 2, 2016.

[21] Ofir Press, Noah A. Smith, and Mike Lewis,*"Train short, test long: Attention with linear biases enables input length extrapolation",* 2022.

[22] Michael Günther, Louis Milliken, Jonathan Geuter, Georgios Mastrapas, Bo Wang, and Han Xiao, *"Jina embeddings: A novel set of high-performance sentence embedding models",* arXiv preprint arXiv:2307.11224, 2023.

[23] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei,*"Text embeddings by weakly supervised contrastive pre-training"* ,arXiv preprint arXiv: 2212.03533, 2022.

[24] Yoshua Bengio, Nicholas Léonard, and Aaron Courville*,"Estimating or propagating gradients through stochastic neurons for conditional computation"* arXiv preprint arXiv:1308.3432.

[25] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman*," Evaluating large language models trained on code",* arXiv preprint arXiv:2107.03374.