Volume 9, Issue 5, May – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

Leveraging LLM: Implementing an Advanced AI Chatbot for Healthcare

Ajinkya Mhatre[1]
[1]Dept of Information Technology, Pune Institute of Computer Technology, India

Sandeep R. Warhade[2]
[2]Dept of Information Technology, Pune Institute of Computer Technology, India

Omkar Pawar[3]
[3]Dept of Information Technology, Pune Institute of Computer Technology, India

Sayali Kokate[4]
[4]Dept of Information Technology, Pune Institute of Computer Technology, India

Samyak Jain[5]
[5]Dept of Information Technology, Pune Institute of Computer Technology, India

Dr. Emmanuel M[6]
[6]Dept of Information Technology, Pune Institute of Computer Technology, India

Abstract:- Using the application of Large Language Models (LLMs) in healthcare settings, mainly focusing on addressing general illness inquiries through chatbot interfaces. Leveraging the capabilities of LLMs, explore their potential to provide accurate and contextually relevant responses to users seeking information about common health concerns. LLM have the capacity continuously learn and improve from user interaction. Through benchmarking experiments, this paper evaluates the accuracy (61%) of LLM-based chatbots in understanding and responding to user queries related to general illnesses. The findings demonstrate the performance of LLMs against established benchmarks, shedding light on their efficacy in healthcare applications. By examining the intersection of LLM technology and healthcare, this research contributes to advancing the development of intelligent chatbot systems capable of providing reliable and informative support to individuals seeking medical guidance for general health issues.

Keywords:- *LLM, Healthcare, Chatbot, Benchmark, Accuracy.*

I. INTRODUCTION

In recent years, the intersection of artificial intelligence (AI) and healthcare has seen remarkable advancements, revolutionizing the way we approach medical assistance and patient care. [1] Among the various applications of AI in healthcare, chatbots have emerged as a promising tool for delivering personalized medical guidance and support. Leveraging the capabilities of LLMs, such as OpenAI's GPT series, these chatbots have the potential to address a wide range of health-related queries, providing timely and accurate information to users.

This research paper aims to make the integration of LLMs into medical chatbots and evaluate their effectiveness in improving patient engagement, healthcare accessibility, and overall user satisfaction. [2] By harnessing the vast knowledge encoded within these language models, medical chatbots can offer valuable insights, assist in symptom assessment, provide medication advice, offer lifestyle recommendations, and even facilitate mental health support.

The introduction of LLMs into the realm of medical chatbots brings several key benefits. Firstly, these models can comprehend and generate human-like responses, enhancing the conversational experience for users. [3] Additionally, LLMs have the capacity to process vast amounts of medical literature, clinical guidelines, and patient data, enabling chatbots to deliver evidence-based information and personalized recommendations. Moreover, by continuously learning from user interactions, LLM-powered chatbots can adapt and improve over time, refining their responses to better meet the needs of individual users.

The emergence of ChatGPT and other LLMs has sparked debate in academia. These AI-powered chatbots can generate human-quality text and code, potentially boosting research efficiency and educational experiences. However, concerns linger. LLMs might produce inaccurate or biased content, and their use could lead to plagiarism if not implemented responsibly. [4] The key lies in finding a balanced approach that leverages the benefits of LLMs while mitigating risks. This includes developing methods to identify and address bias, ensuring proper citation of LLM-generated content, and fostering critical thinking skills to evaluate the information these models produce.

However, while the potential of LLMs in medical chatbots is promising, several challenges and considerations must be addressed. [5] These include ensuring the accuracy and reliability of information provided, maintaining user privacy and data security, addressing ethical concerns surrounding AI-driven healthcare, and overcoming potential biases inherent in the training data of language models.

Through an in-depth analysis of existing literature, case studies, and empirical studies, this research paper will delve into the current state of LLM-powered medical chatbots, their strengths, limitations, and future directions. [6] By critically examining the opportunities and challenges associated with these innovative technologies, this paper seeks to contribute to the ongoing discourse on the role of AI in transforming healthcare delivery and patient empowerment.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

II. ARCHITECTURE

We use a medical chatbot, in which the proposed system aims to diagnose diseases and provide basic details about them before a patient consults a doctor and this will not only reduce healthcare cost but also improve accessibility to medical knowledge. It is more accessible and efficient healthcare services. It also provides patients and healthcare professionals with invaluable support, offering information on symptoms, diagnoses, treatment options, and medication details.

The source of the dataset are the approved books from Indian Council of Medical Research, All India Institute of Medical Sciences. In the figure 1, LangChain Directory Loader is a component of the LangChain platform that allows users to load documents from a directory into a LangChain project. This allows users to easily process and analyze large collections of documents, such as a corpus of news articles or a set of research papers. LangChain Directory Loader supports a variety of document formats, including plain text, HTML, and PDF. It also allows users to specify a custom loader for any document format that is not supported by default. LangChain Directory Loader is a powerful tool for users who need to process and analyze large collections of documents. It is easy to use and supports a variety of document formats.

The process of automatically locating and removing pertinent data from unstructured text documents is known as text extraction. This can include extracting entities, such as names, dates, and locations, as well as extracting relationships between entities. Text extraction is typically performed using a combination of NLP techniques, such as tokenization, part-of-speech tagging, and named entity recognition. These techniques are used to identify and extract the relevant information from the text.

Converting text to chunks is the process of dividing a text into smaller, more manageable pieces. This can be done for a variety of reasons, such as to improve the performance of a text processing algorithm, to make the text easier to read and understand, or to break the text up into smaller units for storage or transmission. There are several different ways to convert text to chunks. One common approach is to use a regular expression to split the text into smaller pieces based on a specific pattern. This will improve the performance of the text processing and it makes easier to read and understand. This chunk is saved in the server.

In the figure 1, Similarity search is the process of finding similar items in a database. This can be done for a variety of data types, including text, images, and audio. There is various approach to find the search like vector space model, hashing algorithm, etc. Cosine similarity is the approach for the similarity search in this architecture. A metric for comparing the similarity of two vectors is cosine similarity. It is calculated as the cosine of the angle between the two vectors. The cosine of an angle is equal to the dot product of the two vectors divided by the product of their magnitudes. Cosine similarity can be used to measure the similarity between any two vectors, regardless of their length or direction. Then selected chunks with the prompt templates and question from the user is given to the LLM.

LLM is a type of AI model that can process and generate text. Large text and code datasets are used to train LLMs, enabling them to discover statistical correlations between words and phrases. This knowledge can then be used to generate text that is like the text in the training dataset. Once the model has processed the prompt, it generates a response by predicting the most likely continuation or answer based on the information it has learned. The response is produced instantly by the model; it is not pre-written or taken verbatim from any source. However, it is important to note that the model may sometimes produce responses that resemble or paraphrase existing information due to the nature of its training on a vast corpus of text. Language models read prompts by analyzing the input text, understanding the context, and generating responses based on their learned knowledge of language. Although the responses are produced instantly and are meant to be helpful, it is crucial to double-check the information and not rely only on them for crucial details. This response is given to the user.

In the figure 1, whenever the question from the user and answer of the proposed model is saved in the database. The user database is stored in the database.

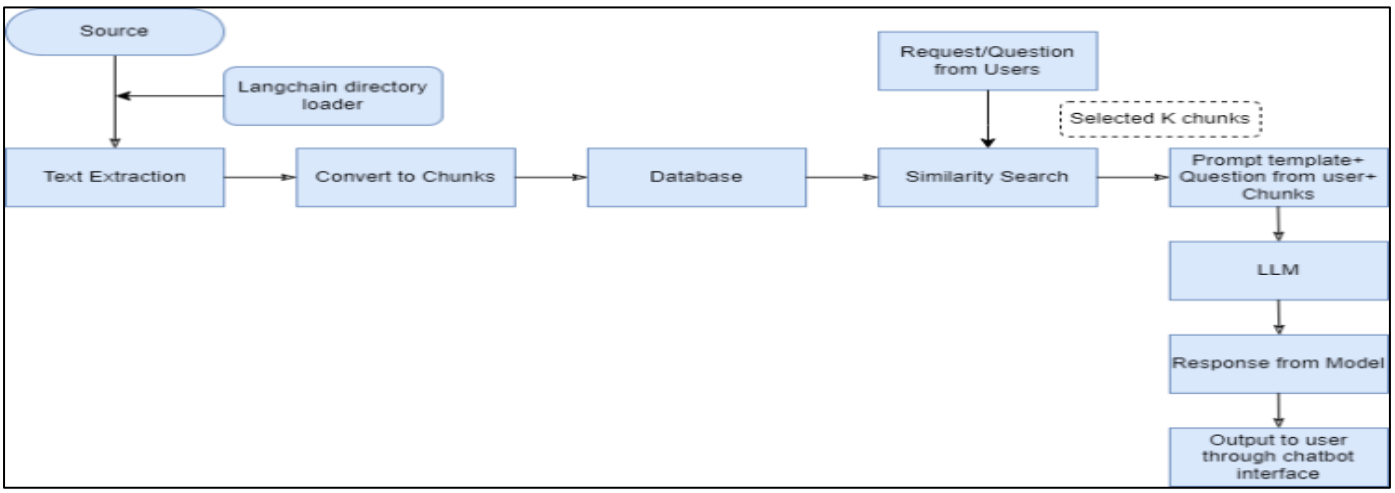

Fig 1 Architecture Diagram

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

III. EXPERIMENTAL WORK

The LLM proposed model demonstrates its versatility by leveraging two distinct hardware configurations for optimal performance. On CPU, the model is deployed on an Intel i5-1135G7 processor boasting 8 cores running at 4.200GHz, ensuring efficient execution of natural language processing tasks. For accelerated inference and enhanced parallel processing capabilities, the model also harnesses the power of GPU acceleration, supported by both NVIDIA GeForce RTX 3050 and AMD Radeon Graphics. Moreover, on CPU, the model benefits from an AMD Ryzen 7 5800H processor, further diversifying the hardware landscape and showcasing the model's adaptability across different computational environments. This multi-hardware approach enables the LLM proposed model to achieve efficient and scalable performance across various hardware configurations, catering to diverse computational requirements and user preferences.

We present a novel approach to transforming textual data from medical journal articles or conference papers into a vectorized database, enabling efficient storage and analysis. [7] Leveraging state-of-the-art natural language processing techniques, we pre-process the raw text, extract key features, and encode them into high-dimensional vectors, preserving semantic relationships and contextual information. This vectorized representation facilitates various machine learning tasks, such as classification, clustering, and information retrieval, while maintaining the integrity and richness of the original medical content. Our methodology not only streamlines data management and accessibility but also empowers researchers and practitioners in the medical field to derive valuable insights and make informed decisions from vast amounts of scholarly literature.

MongoDB, a popular NoSQL database, can be utilized in a wide range of applications, including those involving LLMs like GPT (Generative Pre-trained Transformer) models. [8] Here are some ways MongoDB could be used in conjunction with LLMs:

A. Algorithm:

➢ *Input:*

- Username/email and password entered by the user.

➢ *Connect to MongoDB Database:*

- Establish a connection to the MongoDB database where user credentials are stored.

➢ *Query user Credentials:*

- Search the database for the user with the provided username/email.
- Retrieve the corresponding user document.

➢ *Validate user Credentials:*

- Check if a user document was retrieved.
- If the document exists, compare the password provided by the user with the hashed password stored in the database.
- If the passwords match, proceed to the next step. Otherwise, return an error indicating invalid credentials.

➢ *Generate Authentication Token (Optional):*

- Upon successful validation, optionally generate an authentication token (e.g., JWT) for the user to maintain their authenticated state in subsequent requests.

➢ *Output:*

- If authentication is successful, return a success message or the generated authentication token.
- If authentication fails, return an error message indicating invalid credentials.

Using of the MongoDB, we storage user dataset. In the user dataset, data stored are username, email and password of the user. It also stored the user history, in which user interact with model. This stores the both question and answer from the model. MongoDB is used in the login page and signup page. History of the user is used while interact with the model.

LangChain is a framework designed to simplify the creation of applications powered by LLMs. [9] It provides tools and functionalities to make LLMs more usable and accessible. LangChain plays a crucial role in developing chatbots powered by Large Language Models (LLMs).

B. Algorithm:

➢ *Input: PDF File Path.*
➢ *Initialize a PDF Reader:*

- Use a library like PyPDF2 or pdfminer.six in Python to extract text from the PDF.

➢ *Extract Text Chunks:*

- Iterate through each page of the PDF.
- Extract text from each page.
- Split the text into smaller chunks based on certain criteria (e.g., paragraphs, sentences, or custom-defined boundaries).

➢ *Text Pre-Processing:*

- Normalize the text (e.g., lowercase, remove punctuation).
- Remove stop words (if necessary).
- Apply stemming or lemmatization (optional).

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

➢ *Convert Text Chunks to Vectors:*

- Use a technique like TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings to convert each text chunk into a numerical vector representation.

➢ *Initialize a Vector Database:*

- Choose a suitable database system to store the vectors.
- Set up a schema to accommodate vector storage.

➢ *Store Vectors in the Database:*

- For each text chunk, store its corresponding vector in the database along with any metadata needed for future reference.

➢ *Repeat Steps 3 to 7 for all Pages:*

- Iterate through each page of the PDF, extract text chunks, convert them to vectors, and store them in the database.

➢ *Output:*

Vector database containing vector representations of text chunks extracted from the PDF.

➢ *Optional:*

Index the vectors for efficient retrieval and search operations.

This algorithm outlines the basic steps involved in scanning a PDF file, extracting text chunks, converting them into vector representations, and storing them in a database. Depending on your specific requirements and constraints, you may need to customize certain steps or incorporate additional preprocessing or optimization techniques.

Cosine similarity is commonly used in [10] similarity search tasks, especially in natural language processing and information retrieval. [11]Using of the IDF-PIF, the proposed model gets more accuracy.

C. Algorithm

➢ *Input:*

- Word to find similarity with (query word).
- Vector database containing vector representations of words.

➢ *Retrieve Vector for the Query Word:*

- Search the database for the vector representation of the query word.

➢ *Calculate Cosine Similarity:*

- Iterate through each vector in the database.
- For each vector, calculate the cosine similarity with the vector of the query word.
- Keep track of the highest cosine similarity value and the corresponding word vector.

➢ *Output:*

- The word from the database with the highest cosine similarity to the query word.

When a user sends a query to the model, then the cosine similarity measures the cosine of the angle between two vectors, typically representing text embeddings in a high-dimensional space. Within LLM, Cosine Similarity is employed to compare the representations of different text segments or documents, assessing their similarity based on the direction of their vectors rather than their magnitudes. This implementation enables LLM to effectively identify semantically related documents or passages, aiding tasks such as document ranking, clustering, and question answering. This gives the model to give precise answer to the user.

For boosting Large Language Models with external knowledge, [13] Retrieval-Augmented Generation (RAG) is a technique that enhances the accuracy and reliability of LLMs like LLaMDA2 by incorporating external knowledge sources.

D. Algorithm:

➢ *Input:*

- Prompt: Initial context or information provided to the system.
- Query: User's query or request for additional information.

➢ *Retriever Module:*

- Use the query to search for relevant information from a knowledge base or corpus.
- Retrieve relevant passages or documents based on the query using techniques like TF-IDF, BM25, or neural retrievers such as DPR (Dense Passage Retrieval).
- Filter and rank retrieved passages/documents to identify the most relevant ones.

➢ *Answerer Module:*

- Extract relevant information from the retrieved passages/documents.
- Utilize natural language understanding techniques to comprehend the user's query and the retrieved context.
- Generate a structured representation of the retrieved information to facilitate subsequent processing.

➢ *Generator Module:*

- Input the prompt, query, and the enhanced context (from the Answerer module) into a Large Language Model (LLM).
- Fine-tune the LLM on the given prompt, query, and enhanced context to generate the text response.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

- Optionally, employ techniques like conditional text generation or prompting strategies to guide the generation process and ensure coherence with the context.
- Generate the text response based on the input prompt, query, and enhanced context.

➢ *Output:*

- *Generated Text Response:*

The output of the Generator module, providing the response to the user's query while incorporating the relevant information retrieved by the Retriever module.

By referencing external knowledge sources, RAG ensures the LLM's responses are grounded in reality and up-to-date. The retrieved documents provide context and factual grounding, minimizing the risk of the LLM generating inaccurate information. RAG models can be adapted to different domains by using domain-specific knowledge bases for retrieval. Since RAG retrieves information on-the-fly, the LLM itself doesn't need to be constantly retrained with new information.

MedMCQA is a game-changer for developing intelligent systems that answer medical questions. This massive dataset boasts over 194,000 high-quality, real-world medical school entrance exam questions, covering a whopping 2,400 healthcare topics across 21 medical subjects. With an average question length of just 12.77 tokens, it ensures variety and avoids overwhelming complexity. [14] But MedMCQA goes beyond just facts. Each question comes with the correct answer(s), other options, and detailed explanations, pushing AI models to truly understand and reason their way to the right answer. This diverse and in-depth approach tests 10+ crucial reasoning skills needed in medicine, making it a valuable tool for researchers and developers building the next generation of medical question-answering systems. Ultimately, MedMCQA paves the way for AI that can understand and respond to medical inquiries with accuracy and depth, potentially transforming healthcare communication and decision-making.

Using the MedMCQA, we evaluate the model for the accuracy, reliability, etc. Here we split the accuracy for the two queries, one for the easy questions and other for the hard question.

For example, "*Which vitamin is supplied from only animal source: Vitamin C or Vitamin B7 or Vitamin B12 or Vitamin D?*". This query is divided into the chunks. These chunks are converted into vector format. [15] To find the similar words, cosine similarity is use with vector database. The vector database is in the vector from the journal or the conference papers in the medical field. Using of the cosine similarity, it takes only the top 20 most similar vector are extracted. These vectors as the context are joined with the query to the proposed model. The proposed model computes the data it gets. Then it gives the answer.

For the easy question, whenever these types of the questions are given to the model. It gives the correct answer. It is also giving the correct answer of the relationship between the two topics.

But the hard question, the accuracy is coming in the low. Whenever the user the ask specific topic to the model, the model gives the proper answer for it, but whenever ask for the relationship between it the or like given in the above example. It gives a wrong answer.

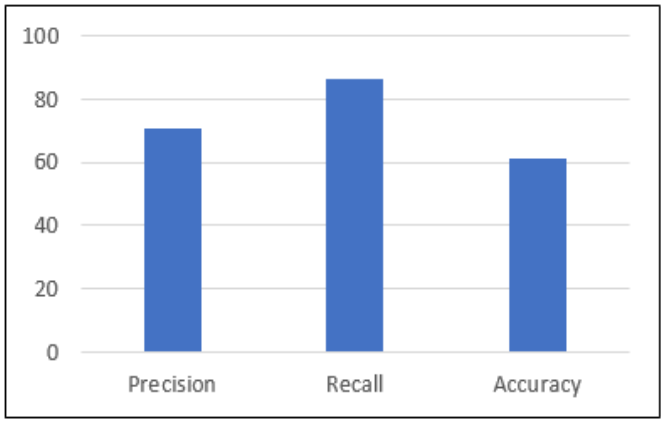

Fig 2 Precision, Recall and Accuracy of Proposed model

There benchmark is taken in the two types one using of the CPU and other the GPU.

When utilizing the chatbot model LLM on a CPU, users may experience lower accuracy compared to when it runs on a GPU or specialized hardware. This discrepancy arises due to the computational demands of LLMs, particularly as their size and complexity increase. CPUs lack the parallel processing power of GPUs and TPUs (Tensor Processing Units), leading to slower inference times and reduced efficiency in handling large-scale language models. Consequently, on CPU-based systems, the chatbot may struggle to process and generate responses with the same level of accuracy and speed achieved on more powerful hardware configurations. Users may notice delays in receiving responses or encounter inaccuracies in the generated text, impacting the overall quality of the interaction.

To address these challenges and enhance the performance of the chatbot on CPU-based systems, optimizations such as model compression, algorithmic improvements, and hardware acceleration techniques can be employed. Model compression techniques aim to reduce the size and computational complexity of LLMs, making them more suitable for CPU deployments without sacrificing accuracy significantly. Additionally, algorithmic optimizations tailored for CPU architectures can improve the efficiency of inference and mitigate performance bottlenecks. Furthermore, leveraging hardware acceleration solutions, such as multi-threading or distributed computing frameworks optimized for CPUs, can unlock additional computational power and enhance the responsiveness of the chatbot, ultimately improving the accuracy and user experience on CPU-based platforms.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

When leveraging a GPU for the execution of the chatbot model LLM, users can expect a notable increase in accuracy compared to CPU-based deployments. GPUs excel in parallel processing tasks, enabling faster computations and more efficient handling of the intricate operations involved in language modeling. With the abundant computational resources provided by GPUs, LLMs can be executed more swiftly, allowing for deeper exploration of contextual relationships and more precise generation of responses. As a result, users interacting with the chatbot on GPU-accelerated systems are likely to experience higher accuracy in the generated responses, leading to a more satisfying and seamless conversational experience.

Furthermore, GPU-accelerated deployments of LLMs offer scalability and flexibility, making them well-suited for handling larger models and accommodating increased workloads. [16] By harnessing the parallel processing capabilities of GPUs, the chatbot can efficiently process a larger volume of data and perform complex computations in real-time, enhancing its ability to understand user queries and generate contextually relevant responses. Consequently, users benefit from an improved accuracy level, as the chatbot can leverage the computational power of GPUs to deliver more nuanced and accurate responses, ultimately enriching the overall quality of the interaction.

Table 2 Benchmark of GPU

GPU	Token/Sec	Average latency	Total Time
1 L4 GPU	558.54 tokens/s	449.88 s	897.23 s
2 L4 GPU	1265.17 tokens/s	179.85 s	397.65 s
3 L4 GPU	1489.99 tokens/s	147.36 s	324.71 s
4 L4 GPU	1401.18 tokens/s	153.09 s	339.51 s

[17]LLMs demonstrated promising results, achieving an overall accuracy exceeding 70%. Bard slightly edged out ChatGPT in terms of accuracy and understandability, providing slightly more comprehensive and easily digestible answers. However, ChatGPT displayed a higher level of caution, generating fewer responses with the potential for misinformation. This trade-off highlights a key challenge in LLM development for the medical field – balancing accuracy with the need to avoid harmful information.

Forget just being "the ultimate AI chatbot," [18] ChatGPT success hinges on user trust and ethical considerations, according to a recent study. While information quality remains king, the research shows users also value feeling connected to the technology, even perceiving it as possessing superhuman abilities! But do not get carried away, ethics play a crucial role too. Users' personal values directly impact their loyalty to ChatGPT, highlighting the importance of responsible AI development.

Using the Vicuna-13 for the ChatGPT-4, the proposed model is compared with it. [19] The evaluation table is given below.

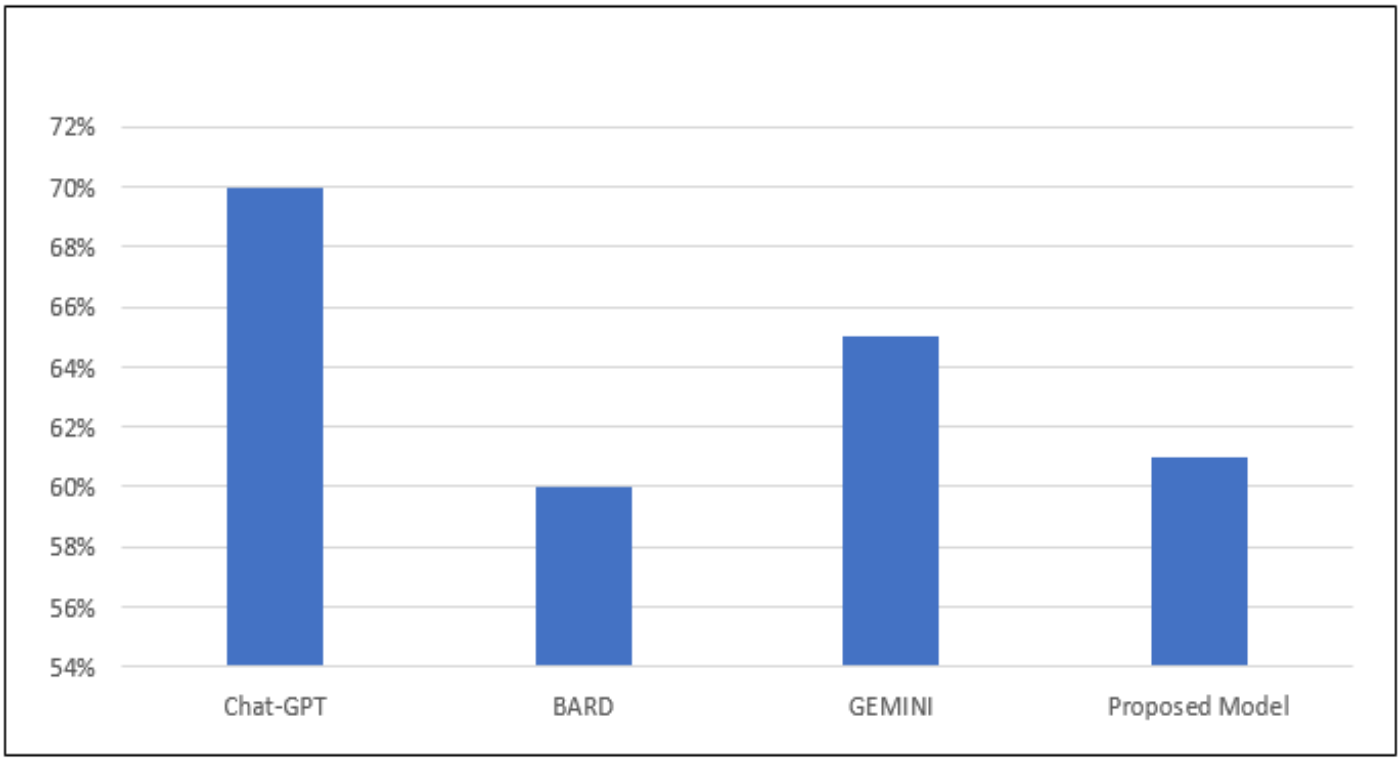

Fig 3 Comparison of LLM Model

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

These agents can handle complex tasks, but building trust in them is crucial. While traditional factors like reliability are important, LLM-based agents pose new challenges. [20] Researchers suggest focusing on making these agents more transparent and understandable to users. This includes aspects like how they reach decisions and the data they use. Additionally, incorporating features that make them relatable, like virtual assistants with voices or avatars, can also increase trust. Ultimately, the success of LLM-based automation hinges on addressing these new considerations and developing trustworthy AI agents that function ethically and in line with human values.

IV. CONCLUSION

In conclusion, the integration of large language models (LLMs) into medical chatbots represents a significant step forward in revolutionizing healthcare delivery and patient engagement. [21] Throughout this research paper, we have explored the potential of LLM-powered chatbots to provide personalized medical guidance, support, and information to users, enhancing accessibility and convenience in healthcare services.

The benefits of LLM-powered medical chatbots are manifold. These chatbots leverage advanced natural language processing capabilities to understand and respond to user queries in a conversational manner, mimicking human interactions. By tapping into vast repositories of medical knowledge encoded within LLMs, chatbots can deliver evidence-based information, assist in symptom assessment, offer medication advice, and promote healthy lifestyle choices.

REFERENCES

[1]. P. Rajpurkar, E. Chen, O. Banerjee, and E. J. Topol, “AI in health and medicine,” Nature Medicine, vol. 28, no. 1. Nature Research, pp. 31–38, Jan. 01, 2022. doi: 10.1038/s41591-021-01614-0.

[2]. B. Galitsky, “LLM-based Personalized Recommendations in Health,” 2024, doi: 10.20944/preprints202402.1709.v1.

[3]. R. Szilágyi and M. Tóth, “Use of LLM for SMEs, opportunities and challenges,” Journal of Agricultural Informatics, vol. 14, no. 2, Jan. 2024, doi: 10.17700/jai.2023.14.2.703.

[4]. J. G. Meyer et al., “ChatGPT and large language models in academia: opportunities and challenges,” BioData Mining, vol. 16, no. 1. BioMed Central Ltd, Dec. 01, 2023. doi: 10.1186/s13040-023-00339-9.

[5]. Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, “A survey on Large Language Model (LLM) security and privacy: The Good, The Bad, and The Ugly,” High-Confidence Computing, p. 100211, Mar. 2024, doi: 10.1016/j.hcc.2024.100211.

[6]. F. Jiang et al., “Artificial intelligence in healthcare: Past, present and future,” Stroke and Vascular Neurology, vol. 2, no. 4. BMJ Publishing Group, pp. 230–243, Dec. 01, 2017. doi: 10.1136/svn-2017-000101.

[7]. Ankit Bhakkad, S.C. Dharmadhikari, M. Emmanuel, and Parag Kulkarni, E-VSM: Novel Text Representation Model to Capture Contex-Based Closeness between Two Text Documents. 2013.

[8]. Anjali Chauhan, A Review on Various Aspects of MongoDb Databases. 2019.

[9]. O. Topsakal and T. C. Akinci, “Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast,” International Conference on Applied Engineering and Natural Sciences, vol. 1, no. 1, pp. 1050–1056, Jul. 2023, doi: 10.59287/icaens.1127.

[10]. C. Luo, J. Zhan, L. Wang, and Q. Yang, “Cosine Normalization: Using Cosine Similarity Instead of Dot Product in Neural Networks,” Feb. 2017, [Online]. Available: http://arxiv.org/abs/1702.05870

[11]. M. Emmanuel, B. D. R. Ramesh, and S. M. Khatri, “A novel scheme for term weighting in text categorization : Positive impact factor,” in Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013, 2013, pp. 2292–2297. doi: 10.1109/SMC.2013.392.

[12]. R. Hasan MBA and J. Ferdous, “Dominance of AI and Machine Learning Techniques in Hybrid Movie Recommendation System Applying Text-to-number Conversion and Cosine Similarity Approaches,” 2024, doi: 10.32996/jcsts.

[13]. P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” May 2020, [Online]. Available: http://arxiv.org/abs/2005.11401

[14]. A. Pal, L. K. Umapathi, and M. Sankarasubbu, “MedMCQA : A Large-scale Multi-Subject Multi-Choice Dataset for Medical domain Question Answering,” 2022.

[15]. M. R. Parvez, W. U. Ahmad, S. Chakraborty, B. Ray, and K.-W. Chang, “Retrieval Augmented Code Generation and Summarization,” Aug. 2021, [Online]. Available: http://arxiv.org/abs/2108.11601

[16]. W. Kwon et al., “Efficient Memory Management for Large Language Model Serving with PagedAttention,” in SOSP 2023 - Proceedings of the 29th ACM Symposium on Operating Systems Principles, Association for Computing Machinery, Inc, Oct. 2023, pp. 611–626. doi: 10.1145/3600006.3613165.

[17]. Y. Li, Z. Song, and W. Li, “Benchmarking Large Language Models in Adolescent Growth and Development: A Comparative Analysis of Claude2, ChatGPT-3.5, and Google Bard,” 2024, doi: 10.21203/rs.3.rs-3858549/v1.

[18]. B. Niu and G. F. N. Mvondo, “I Am ChatGPT, the ultimate AI Chatbot! Investigating the determinants of users’ loyalty and ethical usage concerns of ChatGPT,” Journal of Retailing and Consumer Services, vol. 76, Jan. 2024, doi: 10.1016/j.jretconser.2023.103562.

[19]. L. Zheng et al., “Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena,” Jun. 2023, [Online]. Available: http://arxiv.org/abs/2306.05685

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY1964

[20]. S. Schwartz, A. Yaeli, and S. Shlomov, "Enhancing Trust in LLM-Based AI Automation Agents: New Considerations and Future Challenges," Aug. 2023, [Online]. Available: http://arxiv.org/abs/2308.05391

[21]. B. D. Lund, D. Khan, and M. Yuvaraj, "ChatGPT in medical libraries, possibilities and future directions: An integrative review," Health Info Libr J, Mar. 2024, doi: 10.1111/hir.12518.