Chat With Documents Using Large Language Model (LLM)

*Pathan Simran, **Raykar Vaishnavi,***Pandarkar Vaibhav, ****Kadam Arjun,
***** Bhosale Swati S.

HSBPVT'S GOI FOE, Department of Computer Engineering, Kashti, Maharashtra, India

ABSTRACT

The integration of advanced language models, such as GPT3.5, into information retrieval and conversational AI has created new possibilities. This abstract explores the concept of "Chat with Documents using LLM" where users can interact with large language models to retrieve information from documents in a more natural and intuitive manner.

This innovative approach goes beyond traditional keyword-based searches and allows users to engage in chat-like conversations with LLM. Users can request specific information, summaries, or insights from documents, and the language model, through its deep understanding of the document's content, generates coherent and context-aware responses. This process bridges the gap between human-like conversation and document search, revolutionizing the way we interact with textual data.

One of the key advantages of this approach is its ability to handle multiple documents simultaneously. Users can ask complex questions, clarify doubts, and explore the content of various documents in a seamless manner. The language model interprets the context and learns from the user's interactions, further enhancing its understanding of the documents and improving its responses. Overall, "Chat with Documents using LLM" empowers users to retrieve information from documents in a more accessible and efficient way. It offers a more intuitive and natural conversational experience, making information retrieval a seamless part of human interaction.

Keyword: ***- Large Language Model (LLM), ANN, GPT, Machine Learning, Chat Bots***

1. Introduction

"Chat with Documents using LLM" is a ground breaking project at the intersection of conversational AI and information retrieval. In an age where digital information is abundant but often overwhelming, this innovative approach harnesses the power of Large Language Models (LLM), like GPT-3.5, to revolutionize the way we access and engage with textual data. Traditional document search methods often involve static queries and rigid keyword-based approaches, which can be cumbersome and inefficient. This project, however, envisions a more natural and human-like interaction with documents, where users engage in dynamic conversations with the LLM to extract valuable insights, summaries, or specific information from documents. By doing so, it enhances the accessibility and efficiency of information retrieval, empowering users to ask complex questions, seek clarifications, and explore document content in a more intuitive manner. Furthermore, the LLM's ability to understand context, adapt to user needs, and generate coherent responses transforms the way we interact with documents, offering an exciting glimpse into the future of information access. This introduction sets the stage for a deeper exploration of the project's implications, applications, and ethical considerations.

1.1 .Motivation

"Chat with Documents using LLM" is motivated by the need for a more user-centric, efficient, and intuitive way to access and interact with vast amounts of information. In an era of information overload, this project aims to bridge the gap between human conversation and document retrieval. It empowers users to explore, question, and understand content in a way that feels natural and responsive, unlocking new possibilities in education, research, and knowledge dissemination. This technology promises to make information retrieval more accessible and efficient, transforming the way we learn and work.

2. LITERATURE SURVEY

The convergence of conversational AI and information retrieval, as exemplified by "Chat with Documents using LLM," finds its roots in a rich body of literature that spans various domains, including natural language processing, information retrieval, and machine learning. One key foundational aspect is the development of Large Language Models (LLM), with models like GPT-3.5 gaining prominence. These models have demonstrated significant progress in understanding context and generating coherent text. Prior research has explored their applications in language understanding and generation, often in chatbot systems and text completion tasks.

Information retrieval has traditionally focused on keyword-based searches and retrieval systems. The limitations of these methods have been well-documented, with difficulties in understanding user intent, context, and the inability to handle complex queries. Conversational search, as proposed in this project, is an innovative solution to these limitations. Furthermore, this literature survey highlights the ethical considerations associated with using LLM for document retrieval and conversations. Privacy, biases in language models, and the need for responsible AI deployment have been emphasized in recent discussions.

Leveraging Large Language Models to Power Chabot's for Collecting User Self-Reported Data [1]:

This study examines the use of Chabot's in digital health that can communicate with users in natural language. Current Chatbot's often lack dynamic conversations and customization. The authors propose using Large Language Models (LLMs) like GPT-3 to build chatbots, as they can effectively engage in flexible conversations. However, there is limited understanding of how LLMs interpret prompts, and designing effective prompts can be challenging. Further research is needed on using LLMs for task-oriented chatbots. The authors aim to explore how LLMs can enable chatbots to collect user self-reports in natural conversations.

Towards a human-like open-domain Chabot [2]:

The development of language models has undergone several significant advancements. It started with statistical language models (SLMs), which predicted words based on a fixed context length. However, SLMs faced challenges due to the high dimensionality of words and lacked the ability to capture nuanced meanings. The next breakthrough came with neural language models (NLMs), which used neural networks to model word sequences. NLMs introduced distributed word representations, allowing them to capture more contextual information and generate more accurate predictions. This marked a significant improvement in language modeling performance. Building upon NLMs, pre-trained language models (PLMs) emerged. PLMs leveraged pre-training and fine-tuning processes to capture context-aware word representations. This approach revolutionized language modeling by enabling the transfer of knowledge from large-scale language corpora to specific tasks. PLMs introduced various architectures and strategies, leading to significant advancements in natural language processing tasks. With the introduction of large language models (LLMs), PLMs were further scaled to enhance their capacity for complex tasks. Models like GPT-3 and PaLM showcased emergent abilities with their enormous size and increased contextual understanding. These LLMs demonstrated remarkable performance and pushed the boundaries of what language models can achieve.

Baize: An Open-Source Chat Model with Parameter-Efficient Tuning on Self-Chat Data [3]:

In this paper, we have a clever plan. We use ChatGPT to have conversations with itself, playing both the user and the AI. This way, we create a big set of conversations that we can use to make chat models better. We can even make models that are really good at certain topics, like healthcare or finance. To make these models work well even with limited resources, we use a smart technique. We focus on a model called LLaMA, which can be a good alternative to other chat models. By training LLaMA with our new conversation data, we make a new model called Baize. And we also have a cool idea called Self-Distillation with Feedback to make Baize even better. The best part is that Baize can run on a regular computer, so more researchers can use it.

Limitations:

1. Order of topics not perfectly balanced.
2. Fatigue effects not fully mitigated for Food and Sleep topics.
3. Consistent path orders may lead to less confusion but were not perfectly followed.
4. Some participants may not strictly follow conversation paths.
5. Limited to specific data types for targeted information slots.
6. Incorporating multiple choice questions may impact Chabot performance.
7. Different composition of data types in each topic may influence results.
8. Conversation style of participants not controlled, affecting Chabot performance.
9. GPT-3 chosen for accessibility, but other LLMs may yield different results.

3. PROPOSED SYSTEMS

We proposed this model using a language model called LLM (Language Learning Model) for implementing the chat with documents functionality. LLM is a state-of-the-art language model that has been trained on a large corpus of text data and is capable of understanding and generating human-like responses.

Components:

1. Document Retrieval: To facilitate chat with documents, we need a document retrieval mechanism. This can be achieved by using a search engine or an indexing system that retrieves relevant documents given a user query. The retrieval algorithm can be designed based on various document relevance metrics, such as TF-IDF or BM25.

2. Document Representation: Once the relevant documents are retrieved, we need to convert them into a format that the Chabot can understand. This can be done by encoding the documents using techniques like word embedding’s or BERT (Bidirectional Encoder Representations from Transformers). These representations capture the semantic meaning of the documents and enable the Chabot to understand and respond to user queries.

3. Chabot Model: The core of the chat with documents functionality lies in the Chabot model. LLM can be used as the underlying Chabot model, which takes user queries as input and generates responses based on the retrieved documents. LLM can be fine-tuned using a combination of supervised and reinforcement learning, where the model is trained to generate relevant and coherent responses given the user query and the retrieved documents.

4. Document Summarization: To handle longer documents, a document summarization module can be integrated into the Chabot system. This module would extract the key information from the documents and present it in a concise format to the user. Techniques like extractive summarization using LSTM or transformer-based models can be used for this purpose.

5. User Feedback Loop: To improve the Chabot’s performance over time, a user feedback loop can be incorporated. The Chabot can ask the user for feedback on the relevance and quality of the provided information and use this feedback to improve its responses in future interactions.

Benefits:

1. Users can have meaningful conversations with the Chabot while referring to relevant documents.
2. Chabot can provide accurate and up-to-date information by leveraging the document retrieval and summarization capabilities.
3. Document retrieval and representation techniques can be tailored to specific domains or industries, making the Chabot more domain-specific and efficient.
4. User feedback loop helps in continuous learning and improving the Chabot’s functionality and performance.
5. Chatting with documents using LLM (Language Model for the Legal domain) enables seamless collaboration among legal professionals. They can discuss and make edits in real-time, allowing for quicker decision-making and efficient document drafting.

6. LLM can assist in verifying the accuracy of legal documents during chat conversations. It can provide suggestions on specific legal terms, references, or clauses, reducing the likelihood of errors or omissions.
7. Chatting with documents using LLM eliminates the need for back-and-forth communication via email or physical meetings. Legal professionals can ask questions, seek clarifications, or provide feedback directly within the document itself, saving time and increasing productivity

4. SYSTEM ARCHITECTURE

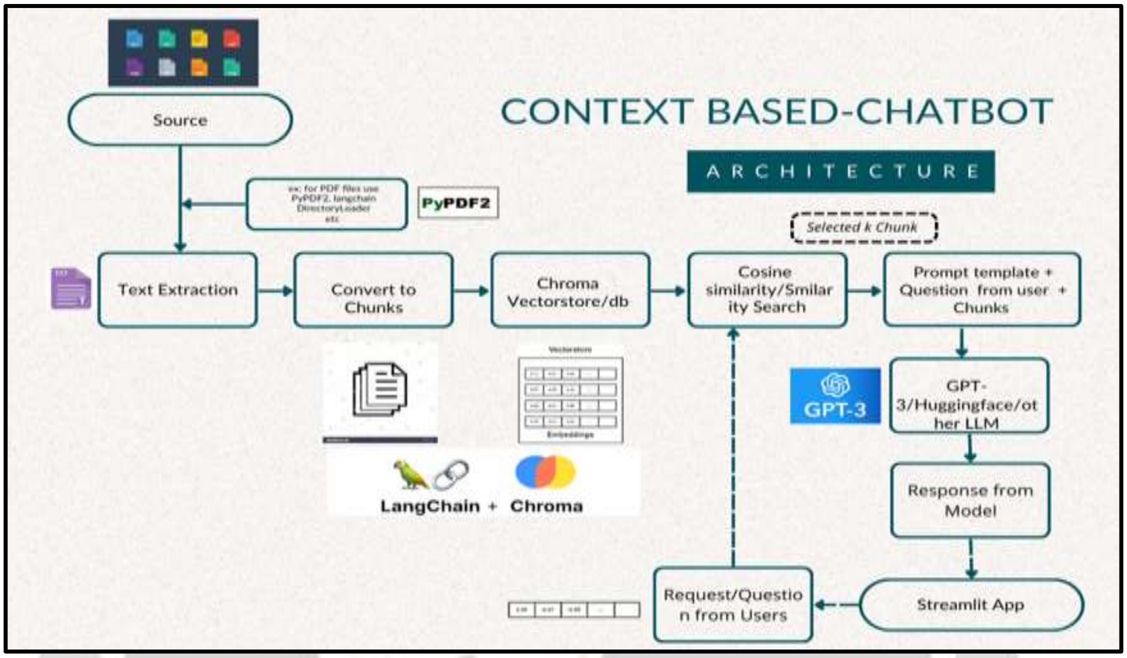
CONTEXT BASED-CHATBOT
ARCHITECTURE
Source
ex: for PDF files use PyPDF2, langchain DirectoryLoader etc
PyPDF2
Text Extraction
Convert to Chunks
Chroma Vectorstore/db
Cosine similarity/Smilarity Search
Selected k Chunk
Prompt template + Question from user + Chunks
GPT-3
GPT-3/Huggingface/other LLM
Response from Model
Streamlit App
Request/Question from Users
LangChain + Chroma

Figure 1: System Architecture

5. SYSTEM IMPLEMENTATION

1. Project Initiation: Define the project objectives, scope, and stakeholders. Assemble the project team, including developers, data scientists, UX designers, and AI experts. Identify key milestones and deliverables.
2. Requirements Analysis: Gather and document detailed software, database, and security requirements. Define user stories, use cases, and functional specifications. Determine technology stack and tools.
3. Design and Architecture: Create system architecture, outlining components and their interactions. Design the user interface for a seamless conversational experience. Plan the database schema and indexing structure. Implement security and privacy measures.
4. Development: Build the software components, incorporating natural language processing, document integration, and security features. Develop user interfaces for a user-friendly experience. Implement ethical AI guidelines and bias mitigation techniques. Ensure scalability and performance.
5. Database Implementation: Set up the database infrastructure, considering data storage, indexing, and redundancy. Implement data encryption and access controls. Develop data retention policies.
6. Testing: Conduct thorough testing, including unit testing, integration testing, and user acceptance testing. Verify system reliability, scalability, and security. Continuously monitor for potential biases and ethical issues in AI responses.
7. Deployment: Deploy the system on a secure server infrastructure. Configure system settings, access controls, and privacy features. Conduct a soft launch for initial user feedback.
8. User Training and Documentation: Provide user training and tutorials for system usage. Develop comprehensive documentation for users and administrators.
9. User Feedback and Iteration: Collect user feedback and implement improvements. Continuously update the system to enhance performance, user experience, and ethical considerations.

10. Compliance and Regulation: Ensure that the system complies with data protection regulations and privacy laws. Be prepared for audits and regulatory changes.
11. Ongoing Maintenance and Support: Establish a maintenance plan for regular updates, bug fixes, and system enhancements. Provide user support and address issues promptly.

6. ALGORITHM

6.1 LLM Algorithm:

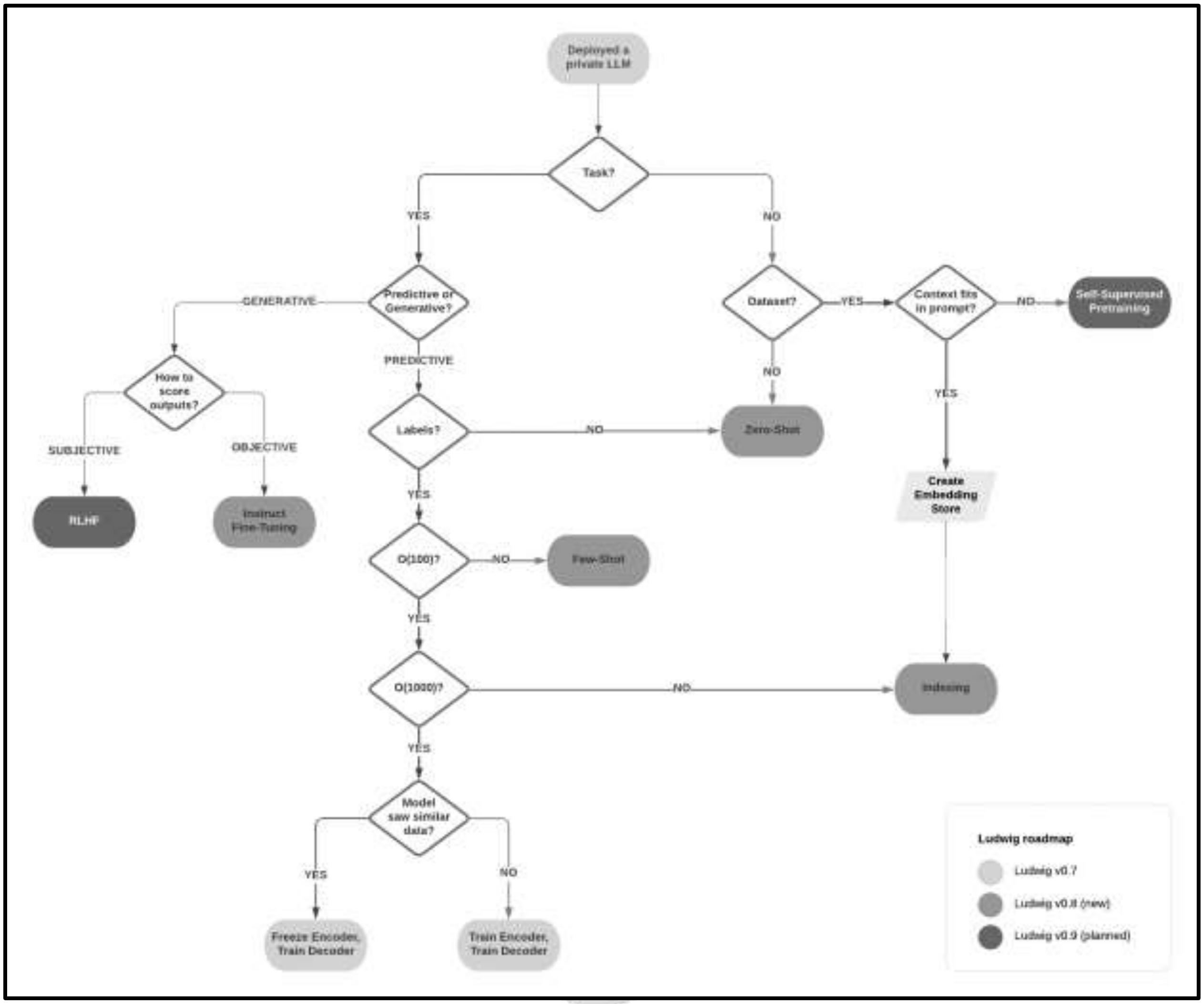

Figure 2: LLM Flowchart

Large Language Models (LLMs) are powerful retrained text generation models capable of performing a number of advanced generative and predictive tasks with little to no addition training, including following reasons:

- Natural language understanding and response (e.g., Chabot’s, Q&A systems)
- Text completion (e.g., coding assistants)
- Text summarization
- Information extraction (e.g., document to table)
- Text classification
- Basic reasoning (e.g., agent-based systems.

With all these capabilities and various techniques being proposed to "customize" LLMs for specific datasets and tasks, it can be daunting to decide where to start. That's where Ludwig comes in!

Ludwig's LLM toolkit provides an easy customization ramp that lets you go from simple prompting, to in-context learning, and finally to full-on training and fine-tuning. ANNs consist of layers of artificial neurons, called nodes, which are connected in a network. Each node receives inputs, processes them using mathematical functions, and produces an output. Through a process called training, ANNs learn from labelled data to make predictions or classify new inputs.

6.2. ANN Algorithm:

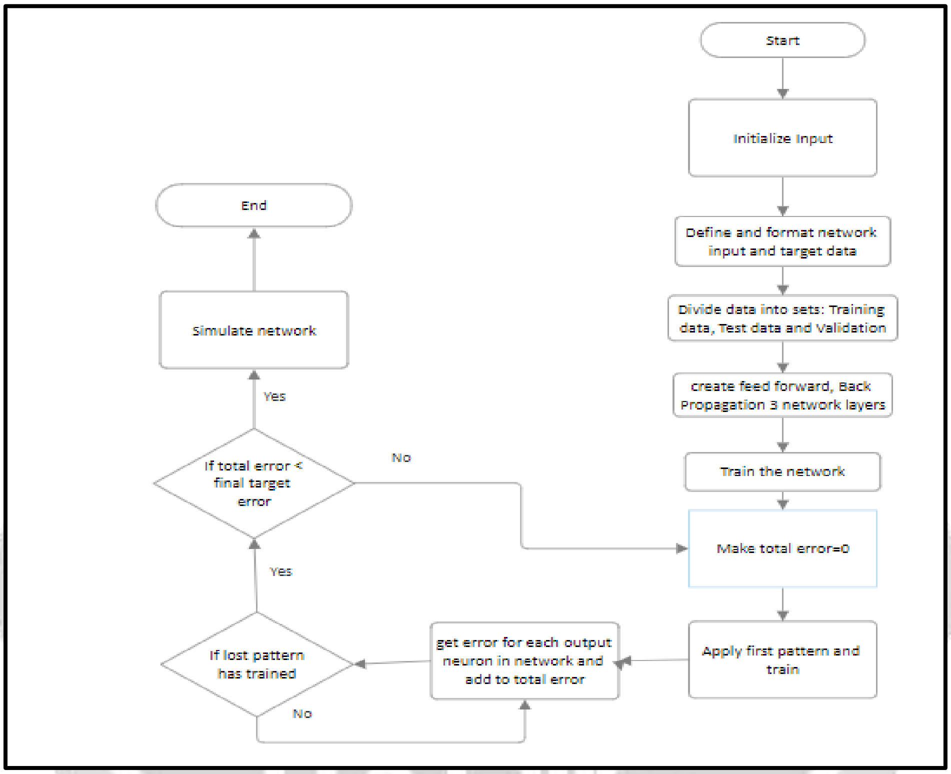
Start
Initialize Input
Define and format network input and target data
Divide data into sets: Training data, Test data and Validation
create feed forward, Back Propagation 3 network layers
Train the network
Make total error=0
Apply first pattern and train
get error for each output neuron in network and add to total error
If lost pattern has trained
No
Yes
If total error < final target error
No
Yes
Simulate network
End

Figure 3: ANN Flowchart

Artificial Neural Networks (ANN) are computational models inspired by the structure and function of biological neural networks. They consist of interconnected artificial neurons, or nodes, organized in layers. The basic unit of computation in an ANN is the artificial neuron, which takes weighted inputs, sums them, and applies an activation function to produce an output. This mimics the way neurons in the brain process information.

ANN algorithms learn from examples, adjusting the weights of connections between neurons to optimize the network's performance. The most commonly used learning algorithm for ANN is called back propagation. It involves propagating the error from the output layer back through the network, updating the weights accordingly.

ANNs can be organized into different architectures, such as feed forward, recurrent, or convolutional networks. Feed forward networks process data from input to output in a single pass, while recurrent networks can also process sequential data by using feedback connections. Convolutional networks are specialized for image and video processing tasks.

ANNs have been successfully applied to a wide range of tasks, including pattern recognition, classification, regression, and time series forecasting. They have been used in various domains such as image recognition, natural language processing, speech recognition, and finance.

While ANN algorithms are powerful and flexible, they also have limitations. They require large amounts of labelled training data to learn effectively, and they can be computationally expensive to train. Additionally, ANN models are often considered black boxes, meaning it can be difficult to interpret why they make certain decisions.

7. ADVANTAGES

1. Improved collaboration: Chatting with documents using LLM (Language Model) allows users to collaborate in real-time on a document. This facilitates efficient communication and collaboration among team members, enabling them to work together effectively on a shared document.

2. Increased productivity: With the ability to chat with documents, users can quickly discuss specific parts or sections of a document without the need for lengthy email threads or separate messaging platforms. This saves time and enhances productivity by streamlining communication and keeping all relevant discussions in one place.

3. Seamless integration: LLM can be seamlessly integrated into existing document editing and sharing platforms, making it easy to use and adopt. Users can chat directly within the document interface, eliminating the need to switch between different applications or platforms.

4. Clear communication: Chatting with documents using LLM allows for clear and concise communication. Users can provide instant feedback, ask questions, or make suggestions within the document context, making the conversation and its intent more apparent. This reduces miscommunication and ensures that everyone is on the same page.

5. Enhanced context and understanding: Document chat with LLM enables users to have a more comprehensive understanding of specific document parts. Users can discuss specific sections in detail, provide additional explanations, or seek clarification directly in the document. This fosters better understanding and collaboration among users, resulting in improved document quality.

8. CONCLUSIONS

The **"Chat with Documents using LLM"** project presents an innovative approach to natural language interactions with documents. This technology offers significant advantages, including efficient information retrieval, natural language conversation, context-aware responses, and privacy controls. It addresses critical aspects such as ethical AI usage and scalability. While it has several potential applications in education, research, customer support, content summarization, and more, it also faces limitations such as bias in responses, document format compatibility, and concerns regarding privacy and resource intensiveness. Maintaining regulatory compliance is an ongoing challenge. Despite these limitations, the project holds substantial promise and utility in various domains.

9. REFERENCES

1. Jing Wei, Sungdong Kim, Hyunhoon Jung,Young-Ho Kim "Leveraging Large Language Models to Power Chatbots for Collecting User Self-Reported Data" arXiv:2301.05843v1 [cs.HC] 14 Jan 2023
2. Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. arXiv preprint arXiv:2001.09977 (2020).
3. Canwen Xu1 , Daya Guo , Nan Duan , Julian McAuley An Open-Source Chat Model with Parameter-Efficient Tuning on Self-Chat Data arXiv:2304.01196v3 [cs.CL]
4. Jacob Austin. 2022. We found that code models get better when your prompt them with "I'm an expert python programmer". the new anthropic paper did something similar, prefixing the model's response with "I've tested this function myself so I know that it

5. BeichenZhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie and Ji-Rong WenA Survey of Large Language Models arXiv:2303.18223v1
6. Ondrej Plátek, Vojt ˇ ech Hude ˇ cek, Patricia Schmidtová, Mateusz Lango ˇ and Ondrej Dušek ˇCharles University, Faculty of Mathematics and PhysicsInstitute of Formal and Applied LinguisticsPrague, Three Ways of Using Large Language Models to Evaluate Chat, arXiv:2308.06502v1 [cs.CL]
7. Lund, B. D., & Wang, T. (2023). Chatting about ChatGPT: How may AI and GPT impactacademia and libraries? Library Hi Tech News, 40(3), 26-29. Chatting about ChatGPT: How may AI and GPT impact academia and libraries?
8. Nitin Malik, Artificial Neural Networks And Their Applications
9. Sakshi Kohli1,Surbhi Miglani2, Rahul Rapariya3, Basics Of Artificial Neural Network, IJCSMC, Vol. 3, Issue. 9, September 2014, pg.745 – 751
10. Humza Naveed1, Asad Ullah Khan1,∗Shi Qiu2,∗Muhammad Saqib3,4,∗Saeed Anwar5,, Muhammad Usman5,6 Naveed Akhtar, Nick Barnes2 Ajmal Mian8, A Comprehensive Overview of Large Language Models, arXiv:2307.06435v5