Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/24apr1132

Code Companion: A Cross Repository Intelligent Code Assistant

Nivetha A.[1]; Sarmitha S.[2]; Vijayaadithyan V. G.[3]; Premkumar Murugiah[4]

[1, 2, 3, 4]Department of Artificial Intelligence and Data Science KPR Institute of Engineering and Technology Coimbatore, Tamil Nadu, India

Publication Date: 2025/07/11

Abstract: Navigating and comprehending varied code bases is a major difficulty in the quickly changing software development market. "Code Companion: A Cross Repository Intelligent Code Assistant" uses cutting edge artificial intelligence-driven chat bot technology to solve this problem. The goal of this system is to give developers a user-friendly interface via which they can query code functionality, structure, and other relevant data from various sources. The project dramatically improves the effectiveness and accessibility of coding skills by utilizing cutting-edge methods in natural language processing, transfer learning, and semantic search. "Code Companion" changes the game for intelligent code help by lowering the learning curve for new projects and encouraging teamwork among developers. This is a significant step toward more connected and understandable digital development environments.

Keywords: *Transfer Learning, Artificial Intelligence, Natural Language Processing, Semantic Search, Vector Database, Cross-Repository Analysis, Chat Bot, Large Language Model*.

How to Cite: Nivetha A.; Sarmitha S.; Vijayaadithyan V. G.; Premkumar Murugiah (2025). Code Companion: A Cross Repository Intelligent Code Assistant. *International Journal of Innovative Science and Research Technology*, 9(4), 3516-3520. https://doi.org/10.38124/ijisrt/24apr1132

I. INTRODUCTION

The potential of conversational interfaces to improve data interaction and user support has been highlighted by their recent evolution and integration into a variety of disciplines. Advances in Natural Language Processing (NLP) in particular have greatly enhanced conversational systems' capabilities, making them indispensable in domains like data analysis and software development. Among these interfaces, chatbots have become one of the most useful tools since they can mimic intelligent user conversations to speed up information retrieval and decision-making.

Assisting developers in comprehending and examining code from diverse repositories is made possible by chatbots' capacity to comprehend natural language inquiries and furnish succinct, pertinent responses. Though data is always growing and changing in the context of software development, existing research shows a gap in chatbot applications that make use of the whole range of data that is available. To overcome these difficulties, this paper presents "Code Companion," a chatbot- based solution that makes it easier for developers to gather and understand information by allowing them to "talk to the data" inside code repositories.

Using an advanced chatbot framework, "Code Companion" interprets user intents and makes codebase interactions easier, which speeds up development and increases efficiency. Through the integration of cutting-edge natural language processing (NLP) techniques and artificial intelligence (AI) analysis, this method facilitates not only the navigation of intricate code structures but also the identification and retrieval of certain code functionality and information dispersed over multiple repositories. This paper describes the creation, application, and assessment of "Code Companion," emphasizing its value to the software development community and its potential to completely transform the way developers work with codebases.

The Introduction is covered in Part I of the article, while the Related Work is covered in Part II. The suggested work is explained in Part III, acknowledged in Part IV, and concluded in Part V.

II. RELATED WORK

This part on related work examines the latest developments and difficulties in natural language processing, semantic search, and code aid tools. It establishes the foundation for "Code Companion" by pointing out the shortcomings of current solutions.

The authors of [1] suggested creating and assessing a chatbot with the goal of making the process of obtaining multidimensional data easier. The chatbot's purpose was to integrate data from several sources, converse with users, comprehend their requests, and extract pertinent information from the BIOD repository. 21 participants in an empirical

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/24apr1132

user study rated the usefulness, visibility, help, and simplicity of the chatbot by assigning them search tasks to complete. Results show that users were happy with the chatbot's usefulness and could communicate with it in an efficient manner, despite some difficulties with the intricacy of the interactions. The study emphasizes the need for ongoing development to improve user experience while highlighting the potential of chatbots to streamline data access.

The authors of [2] provide an extensive overview of AI-generated content (AIGC), with a special emphasis on ChatGPT and its consequences. As part of the methodology, pertinent literature from reliable sources like arXiv, IEEE Xplore, and the ACM Digital Library was systematically reviewed. The operating principles, security/privacy problems, and potential countermeasures of AIGC technologies were examined through an analysis of research articles, conference papers, and industry reports. The results show that AIGC has advanced significantly, especially in models like ChatGPT. But these developments also bring with them several security risks, such as deepfakes, biased and dangerous material, and data theft. To achieve responsible and ethical deployment, the study emphasizes how critical it is to address these issues. It also highlights the need for more investigation and creativity in creating safe, responsible, and functional AIGC frameworks to reduce possible risks and foster confidence among users and stakeholders.

With an emphasis on open data sources, the author [3] offers a unique method for multidimensional searching employing chatbot interfaces. Using natural language interaction, the approach entails creating a chatbot engine inside the Xatkit framework that allows users to formulate sophisticated queries. Through interactions with the BIOD (Blended Integrated Open Data) repository, Brazilian open data from multiple sources is integrated by the chatbot. To formulate multidimensional queries, natural language processing techniques are utilized to extract user intents and parameters. Intent recognition is accomplished by using regular expressions, which allows for dynamic query construction based on user input. The case study illustrates how well the chatbot works to give users easy access to multidimensional data. Natural language interaction allows users to create sophisticated queries without the need for specialized understanding of database architecture or query languages. Based on user input, the chatbot suggests pertinent metrics and dimensions by effectively extracting user intents and parameters. The integration with the BIOD repository showcases the versatility of the approach in accessing open data repositories, offering a user-friendly and engaging experience for querying multidimensional data.

An extensive assessment of language models (LMs) from an all-encompassing standpoint is presented in the [4] study. This investigation's technique takes a multimodal approach, fusing new assessment metrics with established benchmarks. Conventional benchmarks are evaluated on common datasets like Penn Treebank and GLUE and measure accuracy, perplexity, and syntactic parsing performance. The study also uses experimental validation and qualitative analysis to investigate emergent linguistic structure and in-context learning skills. Empirical research and theoretical analysis are used to examine the computational and statistical characteristics of LMs, such as scaling laws and computational efficiency. The findings demonstrate notable progress in LM capabilities, including gains in syntactic parsing, semantic compositionality, and generalization to various tasks. LMs perform better in syntactic parsing accuracy and semantic compositionality, which improves their ability to generalize to different tasks. Furthermore, the development of in-context learning skills shows encouraging promise for improving LM comprehension and reasoning. The efficient scaling characteristics and computational benefits of specific LM structures are revealed through computational analysis, which advances our knowledge of the underlying mechanisms. All things considered, the study offers a thorough summary of LM's capabilities and identifies areas that need more investigation and improvement.

This author [5] uses a multi-step strategy to increase the effectiveness of open educational resource (OER) recommender systems by utilizing natural language processing (NLP). Text data is first extracted from PDF documents and separated into smaller chunks to facilitate efficient processing. These pieces are then transformed into text embeddings using all-MiniLM-L6-v2 model, enabling accurate comparison and retrieval of relevant data. The system uses a vector space model for similarity search to determine the most pertinent text chunks in response to user queries. Furthermore, user inquiries are answered using a pre-trained language model known as GPT-3, which yields accurate and human-like responses. The results demonstrate how much more effective and efficient these strategies make OER recommendation systems.

By accurately answering user questions and retrieving relevant content from a knowledge base of previously processed texts, the system enhances the user's overall experience with instructional resources.

As part of the paper's [8] methodology, chatbots are categorized based on their use, and several design strategies and development platforms are examined. This includes natural language processing functions, pattern matching, stimulus- response methodology, and ontology representation as parsing methodologies. The research delves into the various domains in which chatbots can be employed, with a particular focus on customer service, education, insurance, and the Internet of Things. The study's conclusions show the wide range of applications for chatbots across several sectors. Notable outcomes include the development of chatbots for education, improved customer service in the insurance industry, enhanced workplace productivity through IoT integration, and multilingual communication. These findings demonstrate how versatile and promising chatbots are for satisfying a variety of user needs and improving user experience in general across a few domains.

This paper [9] used a narrative literature analysis methodology to analyze the transition from ChatGPT-3 to GPT-4 to evaluate advancements in AI-driven NLP tools. Secondary data was carefully collected from scholarly

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/24apr1132

publications, technical reports, and online sources, with a focus on performance indicators and application domains. A thematic examination of the gathered material revealed that GPT-4 greatly exceeded its predecessor, particularly in terms of translation accuracy, question-answering proficiency, and sentiment analysis capabilities. The methodical curation of data from places like Google Scholar and arXiv allowed for a detailed evaluation of GPT-4's advances. The test highlighted how well GPT-4 leverages deep learning to provide responses that are more realistic and appropriate for the given circumstances. Better comprehension of complex inputs and faster response times are two more indications of GPT-4's evolution into a more powerful and efficient NLP tool.

These findings demonstrate the critical role that GPT-4 plays in the evolution of AI-driven natural language processing and pave the way for further developments in NLP.

In the study [10], a sophisticated tool that analyzes source code modifications and branch differences automatically through machine learning techniques is shown. The method evaluates messages, comments, and source code to help analysts, code reviewers, or auditors finish tedious jobs on a regular basis. In order to free up user attention for automatically generated alerts, the system optimizes the quantity of source code reviewed per time unit in order to standardize reviewing or auditing procedures. The research offers justifications for using the tool, which is made available as open-source software. The results show how effectively the automated analysis found potential mistakes or flaws in the source code, enhancing the product's security and quality. When all is said and done, the study improves development processes by offering a useful method for examining and analyzing source code.

III. PROPOSED TECHNIQUE

This section introduces the proposed work aimed at enhancing code retrieval and comprehension through a novel semantic search system integrated with a chatbot interface. The architecture of the suggested model is shown in Fig. 1. The work revolves around several key components where each component plays a crucial role in creating a robust system capable of interpreting user queries, retrieving relevant code segments, and providing insightful responses.

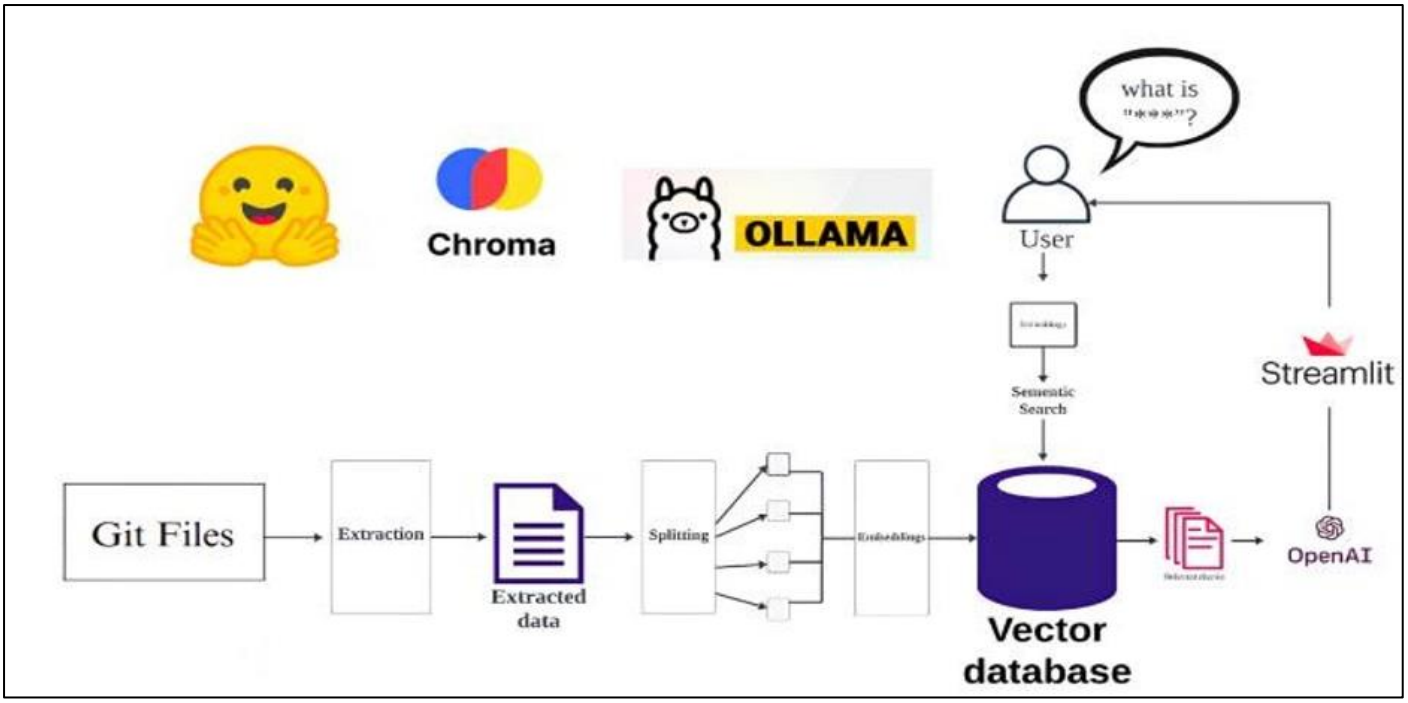

Fig 1 Proposed Architecture

The proposed model comprises three steps: Data Collection and Preprocessing, Semantic Search Implementation, Chatbot Integration.

➢ *Data Collection and Preprocessing*

Our proposed method starts with code extraction from several Git repositories. This technique includes cloning the repositories and retrieving the code files one at a time. After it has been extracted, the raw data is pre-processed to remove any redundant or unnecessary information that could complicate analysis. To ensure that the data is easy to understand and process further, white spaces, comments, and non-executable code should be removed. Following the preparation phase, the code data is divided into smaller bits. These pieces of code could be functions, classes, or other logical divisions that indicate different functionality. The next stage is to convert each segment into a high-dimensional vector by using embedding techniques. Embedding makes it possible to express code by capturing semantic information in a way that machine learning algorithms can comprehend. Specifically, we employ transfer learning techniques to further enhance pre-trained models that understand the broad context of programming languages to our specific datasets.

A concise illustration of the data preparation procedure is shown in Fig 2.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/24apr1132

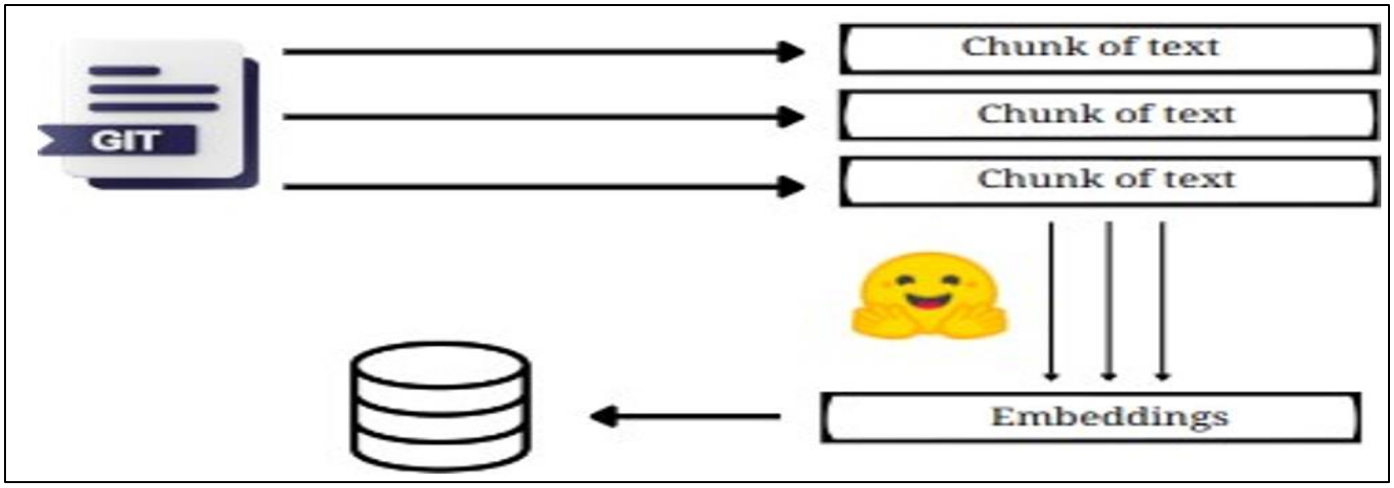

Fig 2 Data Preparation

➢ *Semantic Search Implementation*

By using embedded code, a semantic search system is developed that can decipher the meaning behind user requests. This involves employing natural language processing to assess the context of the user's question and determine which code segments are most relevant to answering their query. Semantic search yields more accurate and contextually relevant results than standard keyword search because it searches for meaning in the query rather than just words that match.

A concise illustration of Semantic search procedure is shown in Fig. 3.

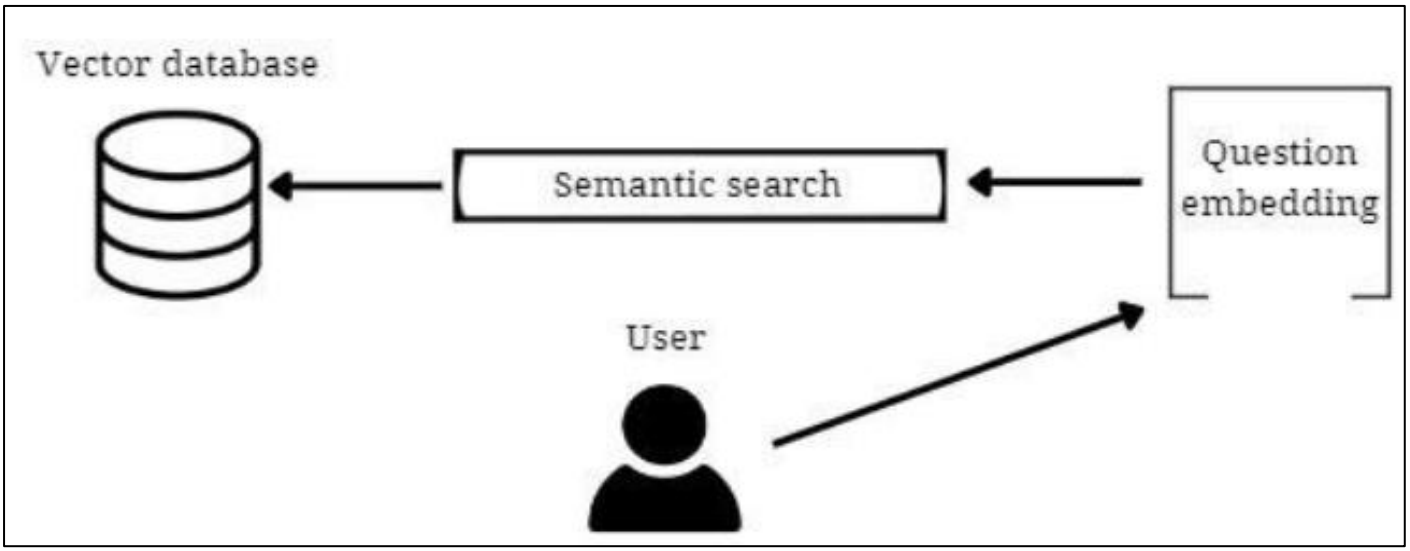

Fig 3 Semantic Search

➢ *Chatbot Integration*

The chatbot user interface of this system serves as a mediator between the user and the underlying technology infrastructure. The chatbot's job is to understand natural language queries to communicate with the semantic search system. Its development makes use of the Flask framework to manage user interactions and provide a fast and user-friendly interface. Because of the way the chatbot is built, it can respond to a wide range of inquiries, from simple ones about code to complex ones.

Using Chroma DB, the high-dimensional vectors representing the code segments are stored. The vector operations required for the effective similarity searches produced by our semantic search algorithm were taken into consideration when designing this specialized database. Quick results are guaranteed by the database management system, which manages the enormous volume of data generated by embedding the code from several repositories.

The chatbot uses CodeLlama's state-of-the-art AI models to enhance its understanding and ability to respond. Because these models were trained on a variety of data sets, they can understand complex linguistic patterns as well as technical features of the code. Because it uses CodeLlama, the chatbot can produce code snippets, provide detailed explanations, and deliver intelligent insights that are superior to those of rule- based systems.

A concise illustration of the Chatbot Integration procedure is shown in Fig 4

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/24apr1132

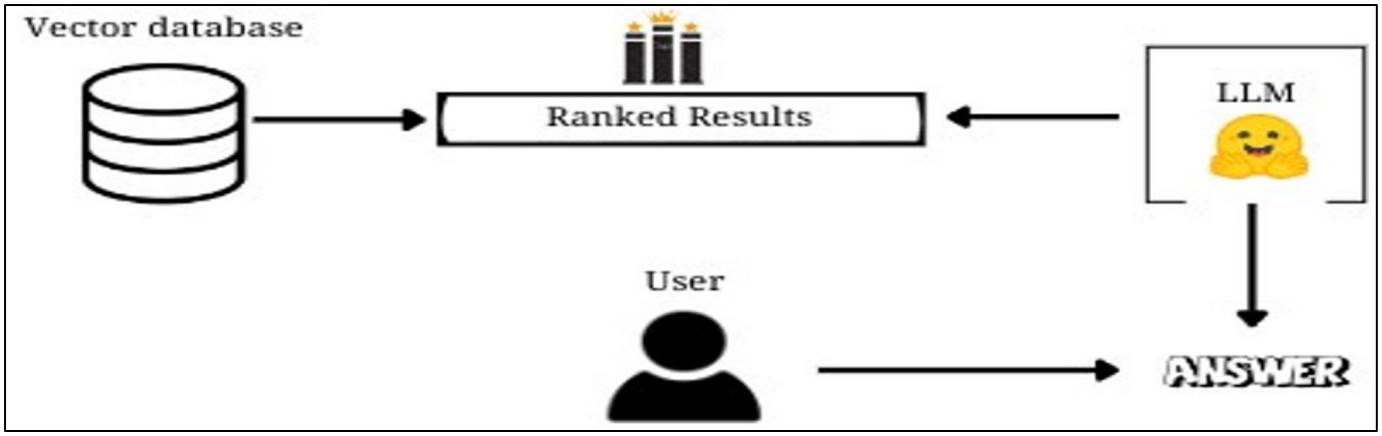

Fig 4 Chatbot Integration

IV. CONCLUSION

In software engineering and security, the automated inspection of source code patches via machine learning algorithms is a breakthrough. This paper presents a tool that simplifies the process of discovering probable defects and vulnerabilities in software systems by utilizing machine learning algorithms on source code, comments, and messages. The updated tool's open-source distribution strategy encourages cooperation between analysts and developers in addition to improving the effectiveness of code reviews and audits. Furthermore, the outcomes show how successful the methodology was, with improved speed and accuracy in assessing branch differences and source code modifications. This kind of automated analysis tool becomes more and more crucial as software systems continue to grow in complexity and size to ensure security and quality of software. This study offers the groundwork for future research and innovation in the field of software development, in addition to addressing present difficulties in the field.

All things considered, the results highlight how crucial automated source code analysis is to preserve the dependability and integrity of software systems in the modern digital environment.

REFERENCE

[1]. Adith Sreeram A S, Pappuri Jithendra Sai. "An Effective Query System Using LLMs and LangChain." International Journal of Engineering Research, July 2023.
[2]. Y. Wang et al., "A Survey on ChatGPT: AI-Generated Contents, Challenges, and Solutions," Xi'an Jiaotong University, Xi'an, China, 2023.
[3]. Maria Helena Franciscatto et al., "Talk to Your Data: a Chatbot System for Multidimensional Datasets," Federal University of Paraná, Curitiba, Brazil, 2022.
[4]. "Large Language Models: A Survey of Artificial Intelligence Advancements through Transformer Architectures by Douglas, M.R. july (2023)"
[5]. An Interactive Framework for Querying Data from Large PDF Files Author: Vishnu B V, Sharath S Rao, Netravathi B Year:2023
[6]. An Optimal Data Entry Method, Using Web Scraping and Text Recognition Author: Roopesh N, Akarsh M S, C. Narendra Babu Year: 2021
[7]. A Review on Chatbot Design and Implementation Techniques" by Ramakrishna Kumar and Maha Mahmoud Ali, February 2020.
[8]. Design and Development of CHATBOT: A Review Authors: Rohit Tamrakar and Niraj Wani Year: 2021 April
[9]. From ChatGPT-3 to GPT-4: A Significant Advancement in AI-Driven NLP Tools Author: M. S. Rahaman, M. M. T. Ahsan, N. Anjum, H. J. R. Terano, M. M. Rahman Year: 2023
[10]. Automated Analysis of Source Code Patches using Machine Learning Algorithms" by Antonio Castro Lechtaler et al., July 2023
[11]. "Unveiling Covert Conversational Agents: Enhancing Insight Archives and Dialog Acts with ChatGPT." In Proceedings of the 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I- SMAC 2023). IEEE Xplore, 2023.
[12]. OpenAI, "Language models can explain neurons in language models," 2023. Accessed: May 17, 2023. [Online]. Available:https://openaipublic.blob.core.windows.net/ neuron- explainer/paper/index.html
[13]. T. Wu et al., "A brief overview of ChatGPT: The history, status quo and potential future development," IEEE/CAA J. Automatica Sinica, vol. 10, no. 5, pp. 1122–1136, May 2023.
[14]. Chicaiza, J., Piedra, N., Lopez-Vargas, J., Tovar-Caro, E. (2017, April). Recommendation of open educational resources. An approach based on linked open data. In 2017 IEEE Global Engineering Education Conference (EDUCON) (pp. 1316-1321). IEEE.
[15]. Bengio, Y., Courville, A., Vincent, P. (2013). Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), 1798-1828.
[16]. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I (2018). Improving language understanding by generative pre-training