NEXUS: Neural Execution and Understanding System

Shawn Hagler
EEL-4914
Florida State University
Panama City, United States
sth20u@fsu.edu

Jaehyun Lee
EEL-4914
Florida State University
Panama City, United States
jl21bd@fsu.edu

Abstract

Leveraging the power of Large Language Models (LLMs), like ChatGPT, our enhanced virtual assistant framework transforms user interactions with its advanced comprehension and dynamic response generation. These LLMs not only unravel the complexity of human queries but also adeptly generate code, enabling the virtual assistant to seamlessly integrate and execute tasks via APIs without manual intervention. Our approach is paving the way for virtual assistants that autonomously interpret and execute a broad spectrum of complex user requests, bridging the divide between human conversation and automated action — a pivotal stride towards truly intelligent digital assistants.

I. Introduction

In the modern era, the proliferation of digital technologies has given rise to an unprecedented level of interaction between humans and machines. One of the most prevalent manifestations of this interaction is the use of virtual assistants (VAs), which have become deeply integrated into our daily lives. These assistants are called upon to perform an array of tasks, ranging from setting alarms and composing messages to conducting complex web searches.

However, despite their convenience and growing capabilities, existing VA frameworks are often limited by the degree of sophistication with which they can handle and interpret human language, as well as by their ability to autonomously execute actions based on these interactions. The crux of the challenges facing current virtual assistant technologies lies in the limitations of natural language processing (NLP) and the restricted flexibility of Application Programming Interface (API) integrations. While NLP enables machines to comprehend and process human language, it often falters when confronted with nuances, context, and the subtleties inherent in everyday communication. This shortfall is accompanied by the static nature of APIs within these systems, necessitating manual pre-coding and regular maintenance by developers to fulfill even straightforward user requests. These constraints lead to a noticeable disconnect, wherein user commands that deviate from predefined patterns or that require actions beyond the existing API integrations are either misunderstood or entirely unfulfilled. Addressing this problem, our project explores the integration of Large Language Models (LLMs), such as ChatGPT, within the VA framework, with the aim to revolutionize the way user interactions are perceived and acted upon.

The project delves into the potential of LLMs to dismantle the complexities inherent in user queries and to dynamically generate executable code. This advancement could empower VAs to directly interact with and execute tasks via APIs, reducing the reliance on manual coding and broadening the scope of autonomous functionality. The objective is to transcend the existing limitations by fostering a new generation of digital assistants that can seamlessly blend conversational comprehension with efficient task execution, thereby elevating the intelligence and utility of VAs to achieve a new benchmark in human-machine collaboration.

In our paper, we present the background information on the intersection of NLP and API interaction within the VA ecosystem, delineating the current challenges and the necessity for improvement. We then explore our approach to these challenges, offering an insightful look into how LLMs can redefine the capabilities of virtual assistants and, by extension, transform the user experience. This project aims to bridge the gap between human language and machine actions using advanced language models to understand user instructions, generate actions, and execute code, ultimately creating truly intelligent and efficient digital assistants.

II. Existing Works

In the development of our project, we reviewed recent advancements within the domain of large language models and their integration with application programming interfaces (APIs). The body of work pertinent to our research encompasses two notable projects: GorillaLLM, ToolLLM, and Rabbit R1 each with distinct objectives and applications in the field of machine learning.

GorillaLLM represents a step forward in language model interaction with API usage, focusing specifically on code synthesis. This system generates code sequences that enable the invocation of machine learning model APIs, with a discernible bias towards APIs facilitating other machine learning utilities such as image generation models. Despite its innovative approach to integrating API calls into language model outputs,

GorillaLLM has a critical limitation: the actual execution of the code is not part of the model's pipeline. Consequently, verification of the code's functionality in real-world scenarios remains unexplored within the system's capabilities.

Furthermore, its exclusive focus on machine learning model APIs restricts the breadth of applications that the model can address, leaving room for improvement and expansion to include a broader range of services. Building upon the concept of API interaction, ToolLLM emerges as a specialized fine-tuned language model designed to optimize API selection. Unlike its predecessor, ToolLLM's primary function is to discern the most appropriate API for a given user instruction. This functionality transforms ambiguous user needs into targeted API queries, showcasing an advance in language model cognition as it pertains to understanding and utilizing various APIs across different services and platforms. The continual progression of machine learning models, particularly those that interface with APIs, underscores the dynamic nature of the field.

In addition, our research delved into the exploration of Rabbit R1, a commercial device that implements an LLM designed to mimic browser and app interactions in cloud-based environments. Rabbit R1 has garnered substantial attention in the market, leveraging large language models to enable users to perform actions seamlessly and efficiently. With significant funding and a rapidly growing user base, Rabbit R1 exemplifies the practical application of advanced language models in real-world settings, showcasing the potential for enhanced human-machine interactions.

Our project aims to leverage and expand upon these foundational works to enhance the utility and precision of language models in code synthesis and API utilization, reflecting the ongoing evolution of machine learning-enhanced systems and their practical applicability in software development and beyond. As the landscape of machine learning and API integration evolves rapidly, our study is positioned to address the limitations observed in existing systems, thereby contributing to the progressive refinement of language model applications in technology.

III. Implementation

A. Large Language Model

In our implementation to enhance virtual assistant technologies, we have incorporated advanced Large Language Models (LLMs) using the transformer neural network architecture, renowned for its efficacy in natural language processing tasks. These LLMs operate through a dual-component system consisting of a parameter file and a run file. The parameter file is crucial as it holds the neural network's trained weights—these are the essential elements that enable the LLM to comprehend and produce language based on the patterns learned from extensive training datasets.

The core of the transformer architecture is its ability to process sequences of data (words) simultaneously unlike older sequential models. This is achieved through mechanisms like self-attention, which allows the model to weigh the importance of each word in a sentence, regardless of its position. This ability drastically enhances the model's context-understanding capabilities. Additionally, transformers utilize positional encoding to maintain the sequence of words, compensating for the architecture's naturally parallel nature, which does not inherently consider token order.

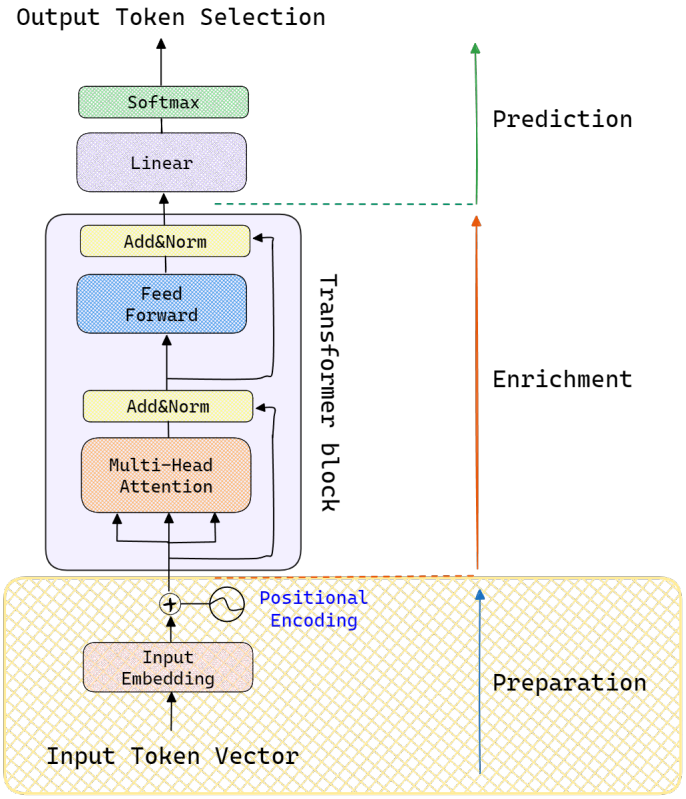

Fig. 1. Transformer Architecture

Beyond the initial training, LLMs must be fine-tuned to tailor their capabilities to specific contexts pertinent to the targeted application area. Fine-tuning allows the model to transition from general language comprehension to understanding nuanced contexts directly relevant to our project needs. This process involves adjustments to the transformer's parameters to enhance its performance on project-specific tasks by leveraging both the original pre-training and additional domain-specific data.

This fine-tuning process also demonstrated the model's enhanced capability to identify when external resources, like APIs, are necessary to fetch additional information crucial for resolving complex queries. This adaptability highlights the potential of LLMs across diverse applications, from powering AI-driven customer support systems to sophisticated, context-aware virtual assistants.

For our implementation within the NEXUS project, we have chosen LLaMa 3 due to its robust pre-training, which includes learning from a plethora of textual data up to April 2023. By employing high-level prompting techniques alongside this model, we can simulate the effects of fine-tuning, thereby

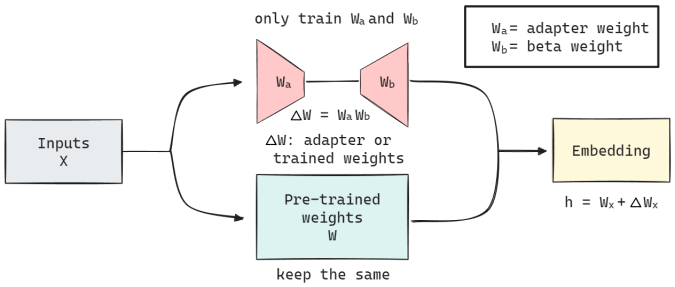

Fig. 2. Fine-Tuning Process

directing the LLM to undertake specific tasks tailored to our project needs without the extensive computational demands typically associated with training a model from scratch.

Through both pre-trained and fine-tuned models, we leverage the vast capabilities of advanced machine learning technologies to tackle the multifaceted challenges central to developing highly effective and intelligent virtual assistants. This strategic implementation underpins our visionary NEXUS project, which stands as a pioneering system in the realm of virtual assistance, showcasing how state-of-the-art technology can be utilized to elevate user interactions with digital systems.

B. NEXUS

Our project, NEXUS, stands at the forefront of virtual assistance technology, designed as a sophisticated end-to-end system that finely weaves together large language models and intricate auxiliary systems. NEXUS is crafted to significantly outperform existing virtual assistants by enhancing user-machine interactions with greater context sensitivity and adaptability.

The intricate fabric of NEXUS is woven from interdependent modules, each tailored to enhance user-machine interactions with precision and contextual acuity. A pivotal component of this system is the Scheduler module, which utilizes a bespoke version of the Llama 3 8B model, meticulously fine-tuned to meet the unique demands of our application. This model acts as the heart of our natural language understanding capabilities.

Fine-tuning a language model, particularly for a task as complex as scheduling and API interaction, involves extensive training on a domain-specific dataset. For the Scheduler, we constructed a specialized dataset comprising various API call scenarios, user requests, and other contextual interactions typically encountered in virtual assistant tasks. This dataset not only includes textual data but is also enriched with vectors representing API functionalities and potential user intents.

The fine-tuning process for our Llama 3 8B model followed a comprehensive approach. Initial training involved exposing the model to general language patterns from vast text corpora, following which we incrementally introduced our domain-specific data. This method ensures that while the model retains its extensive language understanding capabilities developed during pre-training, it also develops a sharp acuity for recognizing and processing commands and queries specific to our operational needs. In practice, fine-tuning involved adjusting the model's weights—a process completed through thousands of iterations of training.

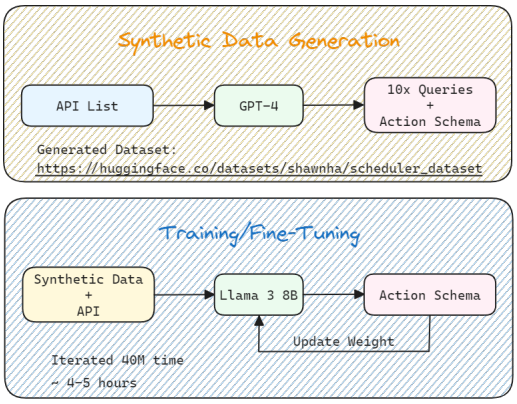

Fig. 3. Synthetic Data Generation and Fine-Tuning

Each iteration refines the model's ability to accurately translate user inputs into structured commands that directly link to our API select embedding. By training the model to predict and align with user expectations, we enhance its responsiveness and utility, making it more than just a query-processing unit.

Transitioning from the API selection phase, the NEXUS system moves to the Code Generator and Responder modules, which utilize the power of the Llama 3 70B model operated on Groq hardware platforms. Groq's architecture significantly bolsters the real-time processing capabilities of NEXUS, enabling rapid code generation and swift response formulation. The Code Generator dynamically constructs Python code in real-time, directly communicating with the requisite APIs based on the action schema defined by the Scheduler. This code is designed to extract the necessary data to fulfill user requests seamlessly.

Groq platforms are pivotal in enhancing the performance of the Responder module as well. By leveraging their powerful processing capabilities, the Responder module can quickly interpret the API's raw data outputs and transform them into structured, natural language responses. This not only ensures responses are swift but also that they maintain a high level of accuracy and context relevance, which is crucial for user satisfaction.

Through continuous updates and maintenance, driven by real-time data integration from the central API hub, NEXUS ensures perpetual system enhancement and relevance. This dynamic updating mechanism keeps the API database robust and extensive, empowering the Scheduler with up-to-date tools for API selection and interaction.

NEXUS is a trailblazer in the virtual assistant industry, heralding a new era of intelligent, context-aware, and highly responsive user experiences. The system's design underscores a substantial leap in artificial intelligence application, showcasing an exemplary model of how future interactions with

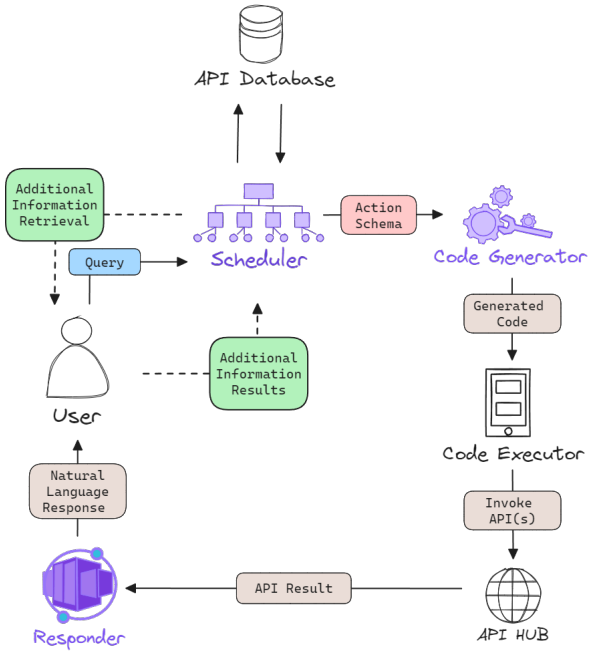

Fig. 4. NEXUS

digital assistants can evolve to become more intuitive and user-centric.

IV. Results

Throughout the testing phase of the NEXUS system, we conducted a series of sample scenarios designed to evaluate the effectiveness of the framework's interconnected modules. These scenarios were crafted to emulate typical user interactions with virtual assistants, showcasing the system's capacity to handle and respond to sophisticated queries. In one key experiment, we assessed NEXUS's ability to find the cheapest available plane ticket.

The process began with the Scheduler, where it demonstrated robust comprehension of the user's input. This was evidenced by its successful extraction and understanding of the various elements involved in the flight search query, such as destinations, dates, and preferences. The Scheduler then accurately constructed a structured query that encapsulated the user's intent for the API Selection, which proved pivotal in interfacing with our extensive API Database. Here, semantic search capabilities were employed to sift through and fetch multiple relevant API options tailored to the structured query. The APIs considered were sourced from RapidAPI, highlighting NEXUS's adaptability and applicability in real-world scenarios. The most suitable API was then meticulously selected to align with the user's specified criteria.

Subsequently, the Scheduler efficiently formulated an action schema, detailing the necessary parameters such as airport codes, potential travel dates, and other relevant query variables essential for conducting the flight search.

The Code Generator module was next tasked with synthesizing executable Python code designed specifically to engage with the selected API. This code was meticulously structured to include all necessary handling for requests, environment variables, and user-specified query parameters. Serving as the operational "muscle" of NEXUS, this code directly interacted with the external service to fetch the required data.

The execution of the generated code was managed by the Code Executor, which seamlessly communicated with the API hosted on RapidAPI. Upon receiving the API's response, the robustness of NEXUS was once again demonstrated. The raw data received was efficiently processed by the Responder module, transforming it into a user-friendly natural language format. The final output provided the user with detailed and actionable information, including the lowest available airfare, airline details, departure and return data, and other pertinent flight information.

Our results clearly indicate that NEXUS is not merely competent but exceptionally powerful in navigating the intricacies of API selection, code execution, and natural language response generation. The experiment confirmed that NEXUS can integrate seamlessly with existing technological infrastructures, interpret complex user requests with high accuracy, and provide automated responses that are both coherent and detailed.

This demonstration firmly establishes NEXUS's potential to revolutionize the virtual assistant space, asserting its ability to manage elaborate natural language queries from users and deliver clear, efficient, and actionable responses, thus reinforcing its place as a transformative force in the realm of digital assistance.

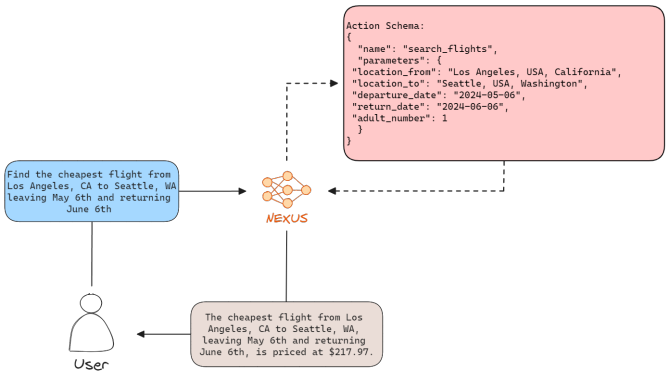

Fig. 5. NEXUS Sample Result

V. Conclusion

In this study, we unveiled and developed NEXUS, a groundbreaking virtual assistant framework, pivoting on the advanced functionalities of Large Language Models (LLMs) like Llama 3, to transcend conventional human-machine interaction paradigms. NEXUS integrates a robust Scheduler module (combining Natural Language Understanding and API Selection capabilities), a dynamic Code Generator, and a responsive

Responder module, creating a seamless conduit between user intent comprehension and automated task fulfillment.

Distinct from traditional virtual assistants which often grapple with rigid task execution pathways and limited comprehension abilities, NEXUS leverages fine-tuned language models and Groq-based hardware acceleration. This allows for precise API interaction and the real-time transformation of complex user instructions into actionable outcomes, all achieved with minimal need for manual intervention. Our extensive prototype testing demonstrates NEXUS's ability to adeptly manage intricate user queries, automate API selections effectively, generate tailored executable code, and deliver coherent, context-rich responses in natural language.

Deploying NEXUS in real-world scenarios has affirmed its potential to meet diverse user needs efficiently, signaling a significant advancement in the domain of virtual assistants. As we look to the future, our ambitions for NEXUS are to cultivate a proprietary LLM tailored specifically to our unique requirements. This advancement will be coupled with an expansion of the API database to encompass a broader spectrum of functionalities and an elevation in system autonomy and performance.

By investing in continuous innovation and adapting to the swiftly evolving fields of machine learning and API technology, we aim to enhance NEXUS's customization capabilities, scalability, and assurances of privacy. These efforts are geared towards ensuring that NEXUS not only keeps pace with technological advancements but also sets new standards in the virtual assistant industry.

Our commitment is deeply rooted in pushing the boundaries of what virtual assistants can accomplish, establishing NEXUS as a cornerstone for the next era where digital assistants are not merely tools, but essential, intelligent partners in our daily digital interactions. With the progression of NEXUS, we envision a paradigm where these systems provide seamless, efficient, and sophisticated support, fundamentally transforming human-machine cooperation. Thus, NEXUS is poised to exemplify the next-generation of intelligent virtual assistants, marking a pivotal leap forward in our ongoing journey towards enriched and effortless human-machine collaboration.

A. Future Work

The trajectory for future advancements of the NEXUS project is promising, building upon its current successes and robust foundations. A significant priority in our forthcoming development phases involves deepening our mastery over the technology that powers NEXUS, especially in transitioning from dependency on third-party models like OpenAI's ChatGPT to cultivating our own proprietary large language models (LLMs).

Securing additional funding—approximately $10,000—is crucial for the development and fine-tuning of a proprietary LLM specifically designed to meet the unique and growing demands of the NEXUS system. With sufficient financial resources, we can pioneer advancements in AI that are finely tuned to the specific functionalities of NEXUS, enhancing our system's autonomy. Ownership of our bespoke LLM will empower the team to rapidly iterate and evolve the model to best suit our needs without the constraints imposed by external models. Additionally, establishing our in-house hosting solutions for the LLM will not only boost performance and scalability but also enhance the privacy and security protocols for user data.

In conjunction with our efforts to develop a proprietary LLM, our team is keen on leveraging the University of Florida's HiPerGator supercomputer facilities. This will provide us with the computational power necessary to undertake the extensive training of our models and conduct high-level processing tasks that are computational-intensive, thereby accelerating our development cycle and enhancing the model's capabilities.

Further, refining our API database remains a critical area of focus. Our plan is to expand the diversity and scope of APIs available to NEXUS, ensuring a wide array of capabilities and responses to user queries. The meticulous selection process for these APIs will be geared towards maximizing utility and relevancy, ensuring that each new addition enhances our system's functionality without redundancy.

In the realm of technical enhancements, we aspire to incorporate advanced algorithms such as Directed Acyclic Graphs (DAG) for improving the scheduling of tasks within NEXUS. Exploring DAG algorithms could revolutionize the way NEXUS handles multi-tiered, complex queries by mapping out a sequence of actions that are optimized for efficiency and accuracy. Additionally, the use of DAGs might play a pivotal role in fine-tuning our language models across different modules—enabling NEXUS to handle multiple APIs simultaneously and more efficiently.

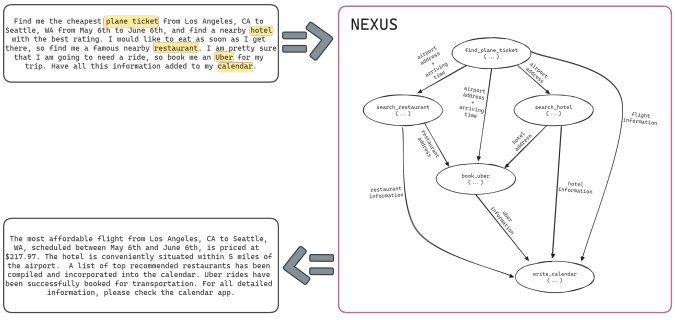

Fig. 6. NEXUS DAG Example

This comprehensive approach to the expansion and refinement of NEXUS underlines our commitment to pushing the boundaries of what virtual assistants can achieve. Through these efforts, we aim to usher in a new era of virtual assistance technology that is more robust, adaptable, and finely attuned to the nuanced demands of users in the digital age. Each step forward with these initiatives is a step towards a transformative future in AI-powered virtual assistance.

REFERENCES

[1] Patil, S. G., Zhang, T., Wang, X., and Gonzalez, J. E. (2023, May 24). Gorilla: Large Language Model Connected with Massive APIs. Retrieved from https://arxiv.org/abs/2305.15334

[2] Qin, Y. et al. (2023, October 3). ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. Retrieved from https://arxiv.org/abs/2307.16789

[3] Rabbit Research Team. (2023, December 3). Learning Human Actions on Computer Applications. Available: https://www.rabbit.tech/research

[4] Vaswani, A. et al. (2017, June 12). Attention Is All You Need. Retrieved from https://arxiv.org/abs/1706.03762

[5] Ouyang, L. et al. (2022, March). Training language models to follow instructions with human feedback. Retrieved from https://arxiv.org/abs/2203.02155

[6] Groq. (n.d.). Why Groq? Retrieved from https://wow.groq.com/why-groq/

[7] Meta. (2024, April 18). Introducing Meta Llama 3: The most capable openly available LLM to date. Retrieved from https://ai.meta.com/blog/meta-llama-3/