International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 04 | April – 2026 DOI: 10.55041/ISJEM06598
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Twinity: A Retrieval-Augmented AI Enabled Avatar-Based Co-Pilot Assistant with Multimodal Chat and Voice Interaction

1st **Firoz Ahmed Siddiqui**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0008-4139-8293

2nd **Md. Aadil Siddiqui**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0006-9931-4617

3rd **Md. Zaid**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0002-5694-5567

4th **Md. Kounen**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0003-0664-6128

5th **M. Amaan Ansari**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0003-0471-9303

6th **Md. Aaquib Sheikh**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0001-4642-1834

7th **Faizan Ahmad Khan**
Artificial Intelligence and Data Science
Anjuman College of Engineering and Technology
Nagpur, India 0009-0002-7503-9711

***Abstract*—** With AI rapidly advancing into many areas of our daily lives, most people are aware that traditional forms of chatbots/virtual assistants have generally helped increase access and automation; however, many of the currently available forms still struggle to provide context-based responses (e.g., be aware of the surrounding environment), support multiple types (i.e., voice, visual, etc.) of communication, or have the ability to create more personalized and engaged responses with a human- like interaction style. Due to the introduction of new technologies, such as Retrieval-Augmented Generation (RAG), vector databases, and neural text-to-speech (or voice synthesis), there is now an opportunity for intelligent assistance to respond more accurately, contextually relevantly, and in a way that feels more human-like than previous generations of chatbots/virtual assistants.

This paper describes the creation of a web application called Twinity, a Retrieval-Augmented AI (RAAI) avatar-based co- pilot assistant. This application is designed to allow multiple forms of communication, such as text and voice. The integrated system includes a chat module and a voice module that provide a consistent intelligent assistant that can answer questions, support users, and seamlessly interact. The backend of the application is created using Python; the frontend is built using React to provide a dynamic and responsive user interface. The Llama Parser is responsible for entering knowledge into the system and understanding documents, and Qdrant was chosen to be the vector database used as a source for semantic search and retrieval capabilities in the system. The Coqui TTS speech engine provides a means of generating realistic spoken responses using neural speech synthesis technology.

In contrast to traditional voice-enabled assistants that rely solely on a memory of previously-trained models, Twinity uniquely utilizes Retrieval-Augmented Generation (RAG). RAG uses additional contextual information from external data sources to minimize hallucinations in its answers and to increase the credibility of its answers, both with respect to factual accuracy and with respect to the type and context of information that it has been programmed to provide. The system was evaluated in an academic setting with a professor, and results show that the system produced highly usable outputs, relevant responses, and practical application for use in the education sector.

Currently, the focus is on Chat and Voice Intelligence. There is a proposed future Face Module, which may include lip sync and facial expressions/emotional intelligence and also capability for rendering personalized avatars. The Twinity framework is an example of a scalable framework for next-generation human- centered AIs that should include Intelligence, Accessibility, and immersive human interaction. Related keywords - Artificial Intelligence, R.A.G., Avatar Assistant, Co-Pilot, Qdrant, Llama Parser, Coqui TTS, Voice Assistant, Web Application, Human Computer Interaction.

Keywords – Artificial Intelligence (AI), Digital Twins, Generative AI, Computer Vision, Natural Language Processing (NLP), Human Resource Management, Education, Autonomous Avatars, Ethical AI, Transparency, Privacy, Human Oversight, Operational Efficiency, Accessibility, Fairness, Inclusivity, Digital Transformation.

I. INTRODUCTION

AI is one of the most ubiquitous technologies of the 21st century; it has changed how businesses are run in industries like healthcare, education, business, entertainment, and communication. AI technologies have provided a means for intelligent assistants to be used by users in completing tasks (e.g., answering questions, automating workflows) and improving productivity. Intelligent assistants like Siri, Alexa, and Google Assistant introduced voice-enabled interaction to millions of people all around the world; likewise, advanced conversational systems based upon large-scale language models (e.g. GPT-3) have emerged offering advanced text-based conversational systems.

Despite these advancements in intelligent assistants, there remain significant limitations within the current generation of assistants. The majority of current intelligent assistants rely heavily upon static training data (i.e. data collected pre-1988) to produce their answers to user queries, resulting in answers that are potentially outdated and/or inaccurate. Furthermore, most current intelligent assistants can only support one manner of interaction(e.g., textual or voice-based), thus limiting the user experience and overall engagement with the users. Finally, the majority of intelligent assistants today do not have a visual identity and/or avatar embodiment associated with them, resulting in interactions that are generally perceived as being less personal and/or less immersive.

To overcome these limitations, this article presents Twinity, a co-pilot assistant based upon artificial intelligence and avatar technology that delivers a more intelligent, engaging, and multimodal user experience. The Twinity co-pilot will utilize Retrieval Augmented Generation (RAG), Conversational artificial intelligence, and neural voice synthesis in providing users with unique and engaging experiences.

Right now, Twinity has Two Functional Component:

The Chat module that allows users to chat with Natural Language Text Queries, and a Voice Module that allows users to communicate via Speech with Synthesized Voice Responses from Coqui TTS.

The system architecture is designed for future expansion and will include a Face Module that will provide visual avatar capability such as Lip Sync, Emotion Display, and Customizable Digital Twin Representation.

The goals of this study are to:

Design and implement a Retrieval-Augmented AI Assistant
Integrate Text and Voice Communication in one system
Improve Relevance of Responses through Semantic Retrieval with Qdrant
Provide a more Natural Method of Voice Interaction for Improved Accessibility
Provide a Scalable Solution for Future Avatar Intelligence

The rest of the paper presents Related Work, System Design, Methodology, Implementation, Evaluation, Advantages, Limitations and Future Directions.

II. THE LITERATURE REVIEW

2.1 Intelligent Assistants

Before the second generation of intelligent assistants, command execution and predefined responses formed the main focus of their development. However, while early systems such as Siri and Alexa helped popularize voice interaction as a user interface for intelligent assistants, they had limited ability to understand context and reason adaptively.

Now, thanks largely to recent advances made in the development of large language models (LLMs), intelligent assistants have greatly improved their ability to understand natural language input and produce highly detailed and contextually-rich outputs. Nevertheless, despite these advances, LLMs may still hallucinate and produce inaccurate fact data when generating responses to fact-based queries that require domain-specific expertise.

2.2 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a newer paradigm that provides a solution for the limitations associated with the use of standalone LLMs. Rather than generating an answer to an input query solely based on internally-stored model parameters, RAG technologies allow an intelligent assistant to retrieve information from external knowledge bases and use that information as context when generating an answer response. This enhances the overall factual correctness of generated answers, reduces the likelihood of hallucinatory outputs, and makes it easier for intelligent assistants to be adapted to particular domains or fields of expertise.

Typically, there are four steps involved in RAG technology implementation:

1) The ingesting of documents into an external knowledge base;
2) Generating embeddings for each of the ingested documents;
3) Using a vector similarity search to retrieve the most relevant embedded documents to the input query; and
4) Generating an answer response using the relevant results from the vector similarity search.

2.3 Vector Databases

Both FAISS and Pinecone are examples of popular vector-based databases used by semantic search systems to store high-dimensional embeddings for purposes of retrieving similar objects using vector-based similarity search techniques. As a result of their ability to quickly retrieve embedded document information along with their filtering capabilities and seamless integration into Python-based applications, Qdrant is a great choice for Intelligent Assistant deployments.

2.4 AI Avatars and Digital Human Beings

The advancement of AI avatars - digital 3D actors that can imitate humans - marks the next evolution of HCI. By integrating voice, facial movements, and personality into an avatar's design, AI avatars enable much more engaging and realistic experiences for users of all ages and types. New developments in digital human representation and talking face generation support the continued development of new educational, customer-service, tutoring, and productivity solutions that utilize AI technology.

2.5 Research Gap

Currently, most research and development will focus on only a single or limited number of capabilities from among the following:

1) Conversational Intelligence
2) Voice Communication
3) Knowledge Retrieval
4) Embodiment of an Avatar

There are few examples of complete systems that have combined all of these capabilities and done so in a modular and scalable fashion. Twinity is a solution that combines RAG technology with Voice Intelligence and future Avatar Embodiment in a complete offering, providing a shared co-pilot assistant for individuals and businesses alike.

III. PROPOSED SYSTEM

3.1 Introduction

Twinity is an AI co-pilot assistant for the internet that allows users to interact via text, voice, and with context-aware intelligence. Twiinity will be developed and utilized in different areas including; academic assistance, productivity assistance, document assistance, and general user interfaces.

The Twinity AI co-pilot system utilizes a modular design so that new features can be added without having to reengineer the entire Twinity platform.

IV. SYSTEM ARCHITECTURE

4.1 The User Interface (UI) Layer Using React

The user interface (UI) layer is made using React, an open-source library. Through React, the user interface is both responsive and interactive.
The following are native components to the interface:
- Chat window component
- Voice Control Component
- Input Box Component
- Response Display Component
- User Navigation Component

React also provides real-time updates, fast rendering of page elements, and fluid interaction among all components of the user interface.

4.2 The Core Backend Layer Using Python

The core backend layer is responsible for all of the underlying AI functionality, including:
- Managing API requests to/from the ChatGPT server
- Preprocessing the user's query
- Executing the Retrieval Pipeline
- Generating a response
- Coordinating audio synthesis for generating the audio response

The programming language used for the core backend functionality is Python; this design was chosen due to its strong support for AI development and the many compatible libraries available for machine learning.

4.3 The Knowledge Processing Layer (KPL)

Llama Parser
Llama Parser is used for processing both structured (table-type) and unstructured (written) documents. When a PDF or other document is provided, Llama Parser will extract the text inside; after the text is extracted, Llama Parser will convert it into machine-readable files that can be embedded to the KPL.

Embeddings
The text extracted from documents will be converted from their original text and turned into numerical vector representations using embedding models. The embeddings are created based on the semantic meaning of each piece of input rather than ensuring that the output matches based on keyword matching.

Qdrant Vector Database
The generated embeddings can be stored in the Qdrant vector database and utilized to provide fast similarity searches and context-based retrieval of the most relevant chunks of data in response to the user's query.

4.4 AI Response Layer

The collected context will then be fed into an LLM resulting in a final output response that is built on relevant context leading to fewer hallucinations and improved answer quality.

4.5 Voice Layer (Coqui TTS)

The second layer of the AI agent consists of converting the text answer into human-likeness or naturalness via even more advanced TTS algorithms/technologies using Coqui TTS. Compared to conventional TTS systems that generate robotic-sounding speech, Coqui's advanced neural-based synthesis technology produces crystal-clear, highly accurate audio/SND with near-realistic human qualities.

V. WORKFLOW OF TWINITY

The Process of Twinity
1. User submits their text or voice input to Twinity.
2. When utilizing voice input, Twinity converts the spoken language to text.
3. Twinity processes the user's request with its backend system.
4. Twinity generates embeddings for the given request.
5. Qdrant retrieves the most pertinent data chunks related to the user's request.
6. Twinity sends the user-defined context (i.e., retrieved data chunks) to its language model.
7. The language model produces a final response for the user’s request.
8. Twinity displays the user’s final response in a chatbox.
9. If the user is in voice mode, Coqui TTS translates the final response into speech.

VI. Implemented Modules

6.1 Chat Module

The chat module allows for natural language use by means of text; therefore, users can ask any question or request to have something explained or get academic assistance, as well as chat with others.

Functionalities

- Messages in real-time
- Answers based on conversation context
- Knowledge-based responses
- Simple user interface
- Quickly handle queries
- Use Cases
- Student assistance
- FAQs
- Productivity guidance
- General AI chat

6.2 Voice Module

The voice module is the next enhancement to Twinity that will help to improve accessibility and convenience by providing users with a spoken response. The Voice Module will allow hands-free use and provide a more engaging experience than a text-only experience.

Functionalities

- Natural speech output using Coqui TTS
- Rapid spoken responses
- Improved accessibility
- Improved engagement
- Human-like audio experience
- Advantages
- Ideal for users that multi-task
- Suitable for visually impaired individuals
- More engaging than text-only systems

6.3 Future Face Module

The Face Module is the next major enhancement coming to Twinity; it will provide a visual representation of the assistant by creating a visually expressive avatar for the assistant.

1) Features to be Developed
2) Real-time lip-syncing
3) Expressions to convey emotional states
4) Eye movement and gestures
5) Custom appearance (digital twin)
6) Realistic talking avatars
7) Multi-lingual chatting avatars

The Face Module has the potential to create a strong sense of trust and enhance the user's connection with the assistant and consequently create a strong emotional bond with the user.

VII. Methodology

The methodology is given below :

7.1 Development Stack

Layer	Technology
Frontend	React
Backend	Python
Document Parsing	Llama Parser
Retrieval Engine	Qdrant
Voice Output	Coqui TTS
Platform	Web Application
Architecture	RAG

7.2 Pipeline Augmented by Retrieval

Step 1: Collecting Data
Relevant materials from outside the system will be gathered and entered into it.

Step 2: Parsing
Llama Transformation Parser will extract structured text from the parsed content of the relevant documents.

Step 3: Chunks
The large, unstructured file will be broken down into small, manageable chunks for processing.

Step 4: Creating The Vector
Each chunk will be processed and converted by using a vector embedding.

Step 5: Storing The Vector
The created vectors will be inserted into an instance of Qdrant.

Step 6: Query Matching
The user will provide input. This input will be translated into a vector embedding for comparison to the stored vector(s).

Step 7: Generating The Completion
Using the input from the user and the retrievable material(s) stored in the system generates a readable text response to the user's initial prompt.

VIII. Experimental Evaluation

8.1 Testing Environment

The prototype was evaluated in an academic setting via a professor (one user) as a web application, to determine usability, relevance, responsiveness, and how practical it is.

8.2 Evaluation metrics

Parameter	Result
Response Relevance	High
Accuracy	High
User Interface Quality	Good
Voice Quality	Good
Ease of Use	High
Practical Utility	High

8.3 Observations
- The user found the system interface easy to operate.
- The user found that retrieval data improved the relevance of responses.
- The user found that having a voice component improved engagement and efficiency.
- The system demonstrated great potential as an educational tool.
- Users preferred multimodal interaction over text-based only systems.

IX. Applications of Twinity

A. Education
- Virtual Study Partner
- College Help Desk
- Course Advisor
- Document Assist

B. Business
- Customer Service
- Employee Onboarding
- Knowledge Assistant

C. Healthcare
- Patient Information Assistance
- Appointment Assistance

D. Personal Productivity
- Daily Task Assistance
- Information Retrieval
- Smart Co-pilot

X. Advantages of the Proposed System

1) RAG for accurate responses
2) Minimized hallucination
3) Multimodal support of voice and text
4) Contemporary web UI
5) Scalable architecture
6) Open-source compatible components
7) Enhanced access
8) Avatar embodiment ready for the future
9) Cross-industry application

XI. Limitations

1. Testing will take place in a limited sample of users.
2. Implementing face components has not yet been completed.
3. Performance relies upon the server resource capabilities.
4. Requires internet access for functionality.
5. The quality of the response is based on the selected language model for responses.

XII. Future Scope

The Twinity project is going to evolve into a digital personal assistant for everyone over time. Key technologies that are coming in the future include:

A. Facial Intelligence
- Talking Avatar
- Lip-Syncing
- Emotion-Based Gestures
- Creating a Unique Look for Each User

B. Advanced Artificial Intelligence
- Memory-Based Personalization
- User Learning of Preferences
- Automatic Task Execution

C. Greater Access to All Users
- Voice Interaction with Users in All Languages
- Regional Language Options

D. Immersive Technology
- AR/VR Avatars
- Metaverse Integration

E. Enterprise Rollout
- Knowledge Assistants for Internal Use
- Bots for Corporate Training

XIII. Conclusion

The authors have created a prototype called Twinity, which is a Retrieval-Augmented AI powered avatar based co-pilot assistant. Twinity is a web-based application that utilizes Python, React, Llama Parser, Qdrant and Coqui TTS for voice interaction with a chat interface. This combination of features allows Twinity to provide intelligent answers that are context aware, relevant, and human-like in nature.

Using Retrieval-Augmented Generation increases factual accuracy and domain flexibility; the Voice Module improves user experience and accessibility; and initial testing shows how effective and efficient it can be as a learning tool in your academic environment.

With the introduction of the planned Face Module, the Twinity Assistant has the potential of evolving into a true multimodal digital human assistant, fundamentally changing how people engage with all types of AI-assisted solutions in educational settings, through business transactions, the delivery of healthcare services, and in everyday life.

References

[1] **Zhao, Q., & Sun, M. (2024).** Retrieval-Based Dialogue Systems with Contextual Understanding. Springer.

[2] **Wang, H., & Liu, Y. (2024).** Human-Computer Interaction with AI Avatars: A Survey. ACM Computing Surveys.

[3] **Singh, A., & Patel, D. (2024).** Semantic Search using Vector Databases in AI Systems. Springer.

[4] **LlamaIndex Team. (2024).** Llama Parser for Structured Data Extraction in AI Applications. LlamaIndex Documentation.

[5] **Li, X., & Zhao, H. (2024).** Conversational AI Systems with Knowledge Retrieval Integration. Springer.

[6] **Zhang, Y., & Lee, K. (2024).** Multimodal AI Assistants: Integrating Voice and Text Interaction. IEEE Access.

[7] **Qdrant Team. (2024).** Qdrant: Vector Database for Next-Generation AI Applications. Qdrant Documentation.

[8] **Meta AI. (2023).** LLaMA: Open and Efficient Foundation Language Models. arXiv.

[9] **OpenAI. (2023).** GPT Models and Their Applications in Conversational Systems. OpenAI Technical Report.

[10] **Johnson, J., & Kumar, R. (2023)**. Enhancing Conversational AI with Retrieval-Augmented Generation Techniques. IEEE Xplore.

[11] **Coqui AI. (2023).** Coqui TTS: Open-Source Neural Speech Synthesis Framework. Coqui Documentation.

[12] **Sharma, P., & Verma, R. (2023).** AI-Based Virtual Assistants in Education: Opportunities and Challenges. Elsevier.

[13] **Kumar, S., & Gupta, R. (2023).** Web-Based AI Assistants Using Modern Frontend and Backend Frameworks. IEEE.

[14] **Park, S., & Kim, J. (2023).** Voice-Enabled Intelligent Systems using Neural Speech Synthesis. IEEE.

[15] **Gao, L., & Callan, J. (2021).** Dense Passage Retrieval for Open-Domain Question Answering. ACM.

[16] **Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., & Riedel, S. (2020).** Retrieval-Augmented Generation for Knowledge- Intensive NLP Tasks. arXiv.

[17] **Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., & Dhariwal, P. (2020).** Language Models are Few-Shot Learners. OpenAI.

[18] **Miller, T. (2019).** Explanation in Artificial Intelligence: Insights from the Social Sciences. Artificial Intelligence Journal.

[19] **Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017).** Reading Wikipedia to Answer Open-Domain Questions. ACL.

[20] **Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., & Gomez, A. (2017).** Attention is All You Need. NeurIPS.