WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

ZERINA: Smart Desktop Assistant with Quantum AI

V. Srinadh[1], P. Ushakumari[2], B. Bhargavi[3], N. Satvik[4], M. Chandra Mohan[5]

Department of CSE-AI & ML, GMR Institute of Technology (GMRIT) – Deemed to be University, Rajam 532127, Andhra Pradesh, India
Email: ushakumaripolumuru@gmail.com, bhargavibonu93112@gmail.com, sathveek12345@gmail.com, manigammohan2006@gmail.com

A B S T R A C T :

The increasing use of digital applications and computer-based activities has created a need for intelligent desktop assistants capable of understanding user commands and performing multiple tasks efficiently. This paper presents ZERINA, a Smart Desktop Assistant with Quantum AI, designed to support voice and text-based interaction, intelligent command understanding, real-time information retrieval, and automated desktop operations. The proposed system integrates a Hybrid Deep Neural Network (DNN) and Rule-Based approach for intent understanding, an Ontology-Augmented Small Language Model (sLLM) for contextual reasoning and task interpretation, and the Quantum Approximate Optimization Algorithm (QAOA) for task scheduling and workflow optimization. ZERINA also integrates external services, including Tavily AI, Google Maps, OpenWeatherMap, and YouTube API, to provide access to real-time information. Its desktop automation capabilities support application control, file management, and document generation. By integrating these AI techniques with quantum-based optimization, ZERINA provides a unified desktop assistance framework that aims to simplify user interactions and automate routine computer operations.

Keywords: *Artificial Intelligence, Desktop Automation, Hybrid Deep Neural Network, Ontology-Augmented sLLM, QAOA, Smart Desktop Assistant.*

1. INTRODUCTION

Artificial Intelligence has become an important area of research with the increasing use of computers, desktop applications, and online services. The continuous growth of digital activities has increased the need for intelligent systems that can understand user commands and perform different computer operations. Tasks such as searching for information, managing files, opening applications, generating documents, and accessing online services are commonly performed by users. However, performing these operations manually can consume time and require interaction with multiple applications. Traditional desktop assistants generally provide limited predefined functionalities and may face difficulties in understanding complex user instructions, managing multiple tasks, and providing context-aware responses. Therefore, developing intelligent desktop assistants capable of simplifying computer operations has become an important area of research. These challenges have encouraged researchers to investigate advanced computational approaches for virtual assistance. Artificial Intelligence techniques such as Deep Learning, Natural Language Processing, and Small Language Models have recently attracted attention for developing intelligent virtual assistants. Hybrid approaches combining Deep Neural Networks and rule-based mechanisms can support user-intent understanding and action mapping. Ontology-based knowledge representation can organize information and relationships, while Small Language Models provide capabilities for natural-language understanding and contextual reasoning. Recent research has also explored quantum computing for solving optimization problems. The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical approach that can be applied to combinatorial optimization and task scheduling. However, integrating these approaches into a single desktop assistant presents challenges related to command understanding, task coordination, workflow optimization, and reliable execution. Therefore, combining AI-based command processing, real-time information retrieval, desktop automation, and quantum-based optimization provides an opportunity to develop more flexible and intelligent desktop assistance systems.

Unlike a conventional desktop assistant that mainly performs predefined operations, the proposed framework is designed to integrate intelligent command understanding, contextual reasoning, automated task execution, and workflow optimization. The main contributions of this work are:

- Integration of **Hybrid DNN and Rule-Based Intent Understanding, Ontology-Augmented sLLM, and QAOA-based task scheduling** within a single intelligent desktop assistant framework.
- Development of voice- and text-based command processing to understand user requirements and perform appropriate desktop operations.
- Integration of real-time information retrieval through external APIs, including Tavily AI, Google Maps, OpenWeatherMap, and YouTube API.
- Implementation of desktop automation functionalities, including application control, file management, document generation, and QAOA-based workflow optimization.

The rest of the paper is organized as follows. Section 2 reviews related work and identifies the research gap. Section 3 describes the proposed system architecture. Section 4 presents the methodology and the integrated AI techniques. Section 5 reports the implementation and experimental results. Section 6 discusses the findings, and Section 7 concludes with future scope.

2. RELATED WORK

Recent research has explored intelligent desktop assistants and virtual assistant systems using artificial intelligence, natural language processing, and automation techniques. Several studies have investigated voice-based interaction, deep-learning-based intent recognition, and rule-based command processing, demonstrating the potential of AI techniques for understanding user instructions and automating desktop operations [1–5]. However, these approaches continue to face challenges related to command interpretation, contextual understanding, task coordination, and integration with different

applications.

Deep-learning and natural language processing techniques remain widely used for developing intelligent conversational systems. Studies have demonstrated the effectiveness of models such as Deep Neural Networks, recurrent neural networks, and language models for intent recognition, command understanding, and response generation [6–10]. Although these approaches provide effective language-processing capabilities, they may face limitations in handling structured knowledge, complex user instructions, and context-dependent tasks.

Ontology-based knowledge representation and Small Language Models have gained importance in improving contextual understanding and intelligent information processing. Research has explored ontology-based reasoning, knowledge graphs, and language-model-based approaches to organize information and improve the interpretation of user queries [11–15]. These approaches demonstrate the importance of combining structured knowledge representation with natural-language understanding for developing more flexible intelligent assistants.

Despite these advances, existing studies generally focus on individual aspects such as voice interaction, intent recognition, knowledge representation, desktop automation, or task optimization. Limited work integrates Hybrid DNN and Rule-Based Intent Understanding, Ontology-Augmented sLLM, real-time API integration, desktop automation, and QAOA-based task scheduling within a single intelligent desktop assistant framework. The proposed ZERINA addresses this gap by combining these approaches to support command understanding, contextual reasoning, real-time information retrieval, automated desktop operations, and workflow optimization.

The reviewed literature indicates that deep-learning and rule-based approaches have been investigated for intent recognition and command processing, while ontology-based methods and language models support structured knowledge representation and contextual understanding. Research on virtual assistants also highlights the importance of integrating external services and desktop automation capabilities. Quantum optimization approaches such as QAOA provide methods for addressing scheduling and combinatorial optimization problems. Table 1 summarizes the reviewed studies.

Table 1. Summary of the reviewed literature

Ref.	Year	Approach / Technique	Domain / Application
[1]	2023	PLS-SEM-based analysis of voice assistant intelligence, including sensing, thinking, and action attributes	Voice assistants; user experience and AI acceptance
[2]	2023	Framework for improving speech interaction, usability, and accessibility of intelligent personal assistants	Speech-based assistance for older adults
[3]	2026	Containerization-based virtual desktop architecture	Virtual desktop environments and desktop resource management
[4]	2022	Experience-based methodology for managing task-oriented virtual assistant software projects	Virtual assistant development and task-oriented applications
[5]	2021	On-device deep learning for device-directed and non-device-directed speech detection	Speech recognition and human-computer interaction
[6]	2025	Ontology-augmented sLLM with knowledge-graph-based intent reasoning	Smart home agents and intelligent device control
[7]	2024	Python-based AI desktop assistant using speech recognition and command execution	Desktop automation and application control
[8]	2026	Voice recognition, rule-based intent detection, and LLM integration	Voice-controlled desktop assistance
[9]	2026	Speech-to-Text, NLP, and Text-to-Speech-based assistant architecture	Desktop applications and system operations
[10]	2024	AI-based virtual desktop assistant with natural-language interaction	Desktop automation, web search, and information retrieval
[11]	2025	Voice-controlled assistant with application integration and automated desktop operations	Desktop productivity and application management
[12]	2025	Conversational AI, document parsing, and LMS integration	Personalized educational assistance
[13]	2024	Multi-feature fitting-based adversarial voice sample generation	Speech recognition, voice synthesis, and biometric systems
[14]	2024	Comparative evaluation of neural networks, TF-IDF, sequential models, and pattern matching	Technical university information chatbots
[15]	2026	PRISMA-based systematic review of LLM-only and hybrid process-model generation approaches	Automated business process modelling and workflow automation

3. PROPOSED SYSTEM ARCHITECTURE

3.1 Overall architecture

The ZERINA Smart Desktop Assistant follows a sequence of processing stages, beginning with user command input and ending with response generation and task execution. The system accepts voice or text commands to understand user requirements and perform different desktop operations. Voice inputs are converted into text for further processing, while text inputs are directly passed to the command-processing module. The system supports operations such as information retrieval, application control, file management, document generation, and access to external services.

The received commands undergo preprocessing to prepare them for intent understanding and further processing. This stage includes text normalization, command interpretation, and identification of relevant information from user instructions. The processed input is passed to the Hybrid DNN and Rule-Based Intent Understanding module. The Deep Neural Network identifies the user's intent, while the rule-based mechanism maps the recognized intent to the corresponding action. This approach helps the system identify the required operation and determine the appropriate processing path.

After intent identification, the system uses the Ontology-Augmented Small Language Model (sLLM) approach to support contextual reasoning and structured knowledge processing. The ontology provides relationships between concepts, commands, and associated tasks, while the sLLM assists in interpreting user requirements and generating appropriate responses. Based on the identified task, the system selects the required execution component, such as real-time information retrieval, desktop automation, document generation, or system operations.

For real-time information retrieval, ZERINA integrates external APIs such as Tavily AI, Google Maps, OpenWeatherMap, and YouTube API to obtain relevant information and provide responses to user requests. The desktop automation module performs operations such as launching applications, managing files, and controlling supported desktop functions. For workflows involving multiple tasks, the QAOA-based optimization module formulates the scheduling problem and determines an optimized task execution sequence based on the defined task dependencies and optimization objective. The selected sequence is then passed to the corresponding execution components.

Finally, the system executes the selected operations and generates responses for the user through the appropriate output interface. The response mechanism provides information about completed tasks or retrieved results. By combining hybrid intent understanding, ontology-augmented reasoning, real-time API integration, desktop automation, and QAOA-based workflow optimization, ZERINA provides a unified framework for intelligent desktop assistance. The overall architecture coordinates these components to simplify user interaction and support automated computer operations.

3.2 Technology stack

Table 2. Technology stack of the proposed system

Layer	Technology	Purpose
Front end	Voice and Text Interface	Accepts user commands through voice and text.
Back end	Python	Manages command processing and coordinates system components.
Preprocessing	Speech-to-Text, Text Preprocessing	Converts voice input into text and prepares commands for processing.
Intent Understanding	DNN, Rule-Based Mechanism	Identifies user intent and maps commands to appropriate actions.
Language Processing	Ontology, sLLM	Supports contextual understanding and command interpretation.
Real-Time Services	Tavily AI, Google Maps, OpenWeatherMap, YouTube API	Retrieves real-time information and online services.
Desktop Automation	Python Automation	Performs application control and file management.
Quantum Optimization	QAOA	Optimizes task scheduling and execution sequence.
Response Generation	Text-to-Speech, Text Response	Provides responses to user commands.

4. METHODOLOGY

4.1 QAOA-Based Task Scheduling and Optimization

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical algorithm used to solve combinatorial optimization problems. In the ZERINA Smart Desktop Assistant, QAOA is implemented to optimize the execution sequence of multiple tasks based on their dependencies and defined optimization objectives. It combines quantum circuits with a classical optimizer to explore different candidate solutions and identify a suitable task schedule. This approach helps organize multiple desktop operations into an optimized execution sequence.

The QAOA-based optimization process begins with the task input received from the rule-based processing layer. The identified tasks and their dependencies are represented mathematically and formulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem. The QUBO formulation represents the scheduling objective through binary variables and a cost function. An initial quantum state is then prepared by applying Hadamard gates to the qubits, creating an equal superposition of possible candidate solutions. This allows the quantum circuit to explore different combinations of task schedules (Fig. 1).

The QAOA circuit consists of alternating cost and mixer unitaries, repeated for a selected number of layers represented by p. The cost unitary applies the problem-specific cost function using the parameter γ (gamma), while the mixer unitary explores different candidate solutions using the parameter β (beta). These operations are applied in sequence to guide the quantum state towards solutions with lower cost. The number of layers determines the depth of the QAOA circuit, and each layer contributes to the optimization process by combining the problem objective with the exploration of possible solutions.

After the QAOA circuit completes its operations, the measurement stage converts the quantum state into a classical bitstring representing a candidate task schedule. The measured bitstring is decoded and evaluated using the defined cost function. A classical optimizer analyses the evaluated cost and updates the parameters γ and β to improve the candidate solution. The updated parameters are then supplied back to the QAOA circuit, and the process is repeated through multiple optimization iterations until the stopping condition is reached.

Finally, the candidate solutions are compared based on their evaluated cost, and the solution with the lowest cost is selected as the optimized task sequence. This sequence is passed to the task execution module, which performs the corresponding operations through the appropriate ZERINA components, such as desktop automation, real-time information retrieval, and document generation. QAOA is responsible for optimizing the task execution order, while the actual operations are performed by the respective execution modules. The optimization process is designed to support organized task scheduling and coordination of multiple desktop operations.

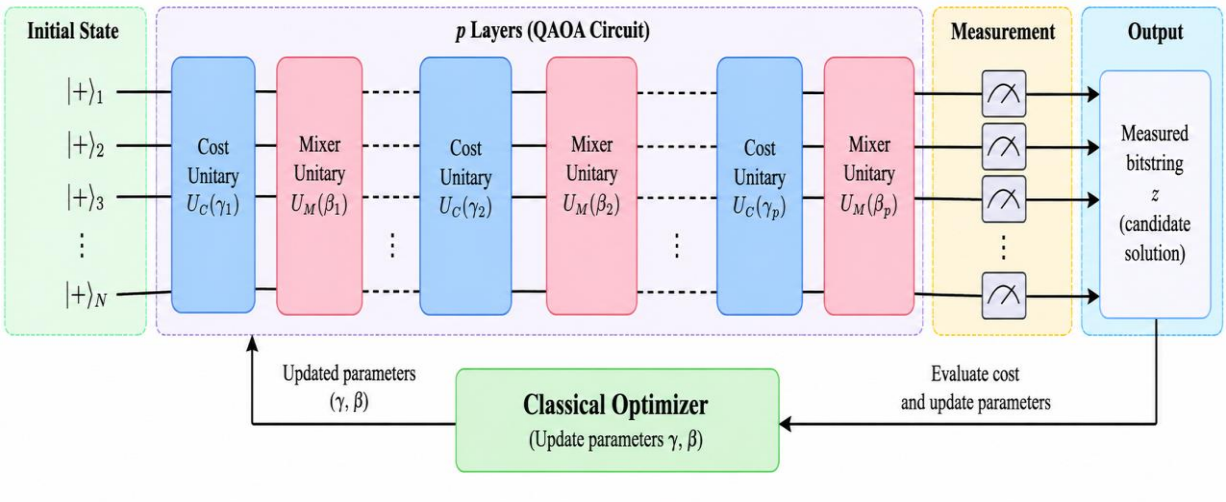


Fig. 1. QAOA-Based Task Scheduling and Optimization Architecture

4.2 Ontology-Augmented Small Language Model (sLLM)

The Ontology-Augmented Small Language Model (sLLM) is an approach that combines language model reasoning with structured knowledge representation to improve command understanding and contextual interpretation. In the ZERINA Smart Desktop Assistant, this approach is used to understand user instructions, identify relevant entities, retrieve associated knowledge, and interpret the required tasks. The ontology provides structured relationships between domain concepts, entities, and tasks, while the sLLM uses this information to support contextual reasoning and task interpretation (Fig. 2).

The processing begins with voice or text commands provided by the user. Voice inputs are converted into text through the Speech-to-Text component, while text commands are directly passed to the preprocessing stage. The input undergoes text cleaning and tokenization to prepare it for further processing. The processed command is then analysed to identify the user's intent, relevant entities, and contextual information. These details provide the foundation for ontology-based knowledge retrieval and language model reasoning.

The ontology component is developed using training and knowledge data, including domain knowledge, task templates, and user preferences. This information is organized into an ontology or knowledge graph consisting of domain concepts, entity relationships, and task knowledge. The knowledge graph represents connections between different concepts and provides structured information that can be used during command interpretation. This representation helps the system associate user instructions with relevant tasks and their corresponding information.

After the knowledge graph is established, the ontology mapping stage maps the identified entities to their corresponding concepts and relationships. It retrieves relevant context from the knowledge graph and adds structured knowledge to the user input. This process provides additional contextual information to the sLLM before reasoning. The mapped information helps connect the user's command with the relevant domain knowledge and task requirements.

The sLLM reasoning component processes the user command along with the retrieved ontology context. It performs intent understanding, context reasoning, and task interpretation to determine the user's actual requirements. The knowledge provided by the ontology supports the language model in interpreting commands using structured relationships rather than relying only on the input text. A context and knowledge feedback mechanism also connects the reasoning component with the knowledge graph to support the exchange of relevant contextual information.

Based on the reasoning output, the decision and task planning component maps the identified intent to an appropriate action. It selects the required operation and generates a task plan for execution. Depending on the user request, the selected task may involve real-time information retrieval through external APIs, desktop automation, document generation, or system operations. The generated task plan is then passed to the corresponding execution component to perform the requested operation.

Finally, the system generates an appropriate response or performs the required desktop action based on the interpreted command. The output is provided to the user through the response interface or by executing the selected operation. By integrating ontology-based knowledge representation with sLLM reasoning, ZERINA aims to improve contextual command interpretation and support structured task planning for intelligent desktop assistance.

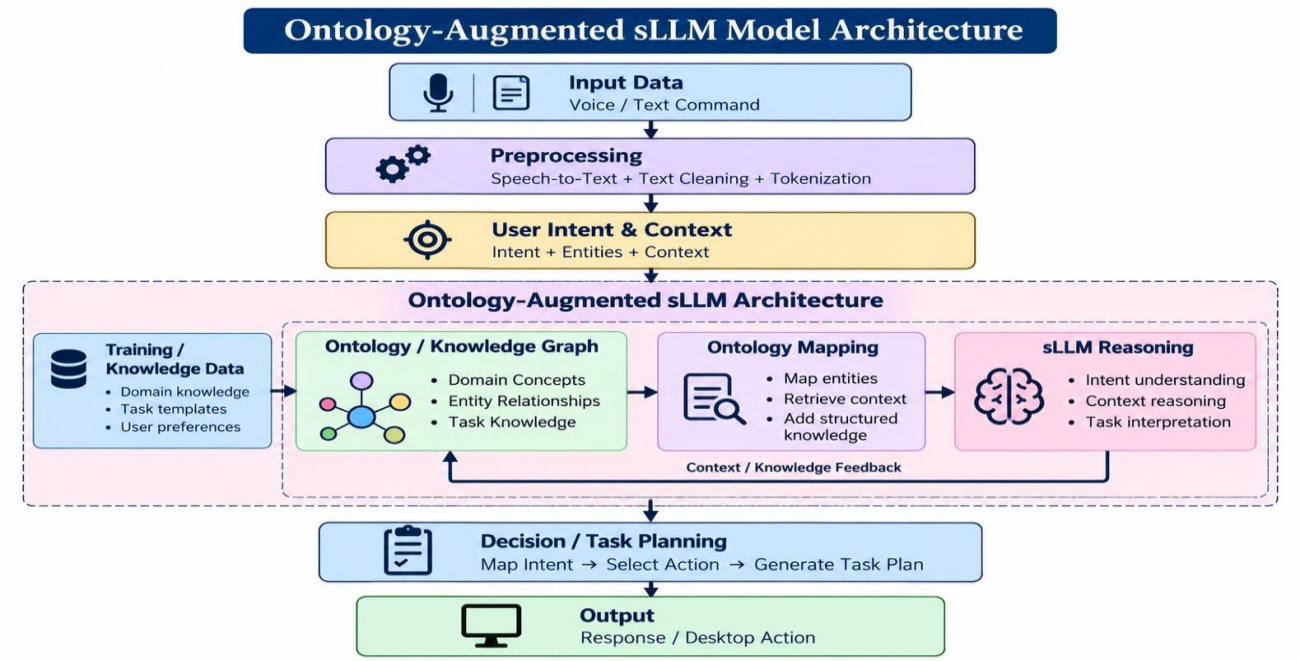


Fig. 2. Ontology-Augmented sLLM Model Architecture

4.3 Hybrid DNN and Rule-Based Virtual Assistant

The Hybrid Deep Neural Network (DNN) and Rule-Based approach combines deep learning-based intent classification with predefined rules to understand user commands and perform appropriate actions. In the ZERINA Smart Desktop Assistant, this approach is used to identify user intentions from voice and text inputs and map them to corresponding desktop operations. The DNN module learns patterns from command data to recognize different intent categories, while the rule-based decision layer matches the detected intent with predefined rules to select the appropriate action (Fig. 3).

The process begins with user input provided through voice or text commands. Voice commands are converted into text, and the resulting input is passed through the preprocessing stage. This stage involves text cleaning, tokenization, and feature representation using word embeddings. The processed features are supplied to the Deep Neural Network, which consists of an input layer, hidden layers with ReLU activation, and an output layer representing the supported intent classes. The network analyses the input representation and predicts the most relevant intent based on the patterns learned during training.

After intent classification, the detected intent is passed to the rule-based decision layer. This layer compares the predicted intent with predefined rules stored in the knowledge base. When a matching rule is identified, the task mapping component selects the corresponding action. If no matching rule is found, the system generates a fallback response or performs re-evaluation. This combination allows the DNN to recognize user intentions while the rule-based mechanism determines the appropriate operation based on predefined conditions.

The selected action is forwarded to the action execution module, which performs different desktop operations. These operations include opening and closing applications, performing file operations such as creating, deleting, and searching files, executing web operations such as information retrieval and weather searches, and controlling supported system settings. The execution module connects the identified user intent with the corresponding system functionality to complete the requested task.

The Knowledge Base and Context Manager supports the overall processing by maintaining user preferences, task templates, predefined rules, and conversation history. This information provides contextual support to the DNN and rule-based decision layer during command processing. By combining learned intent classification with predefined decision rules and contextual information, the hybrid approach enables ZERINA to understand user commands and perform appropriate desktop operations. The resulting responses or completed actions are communicated to the user through the assistant interface.

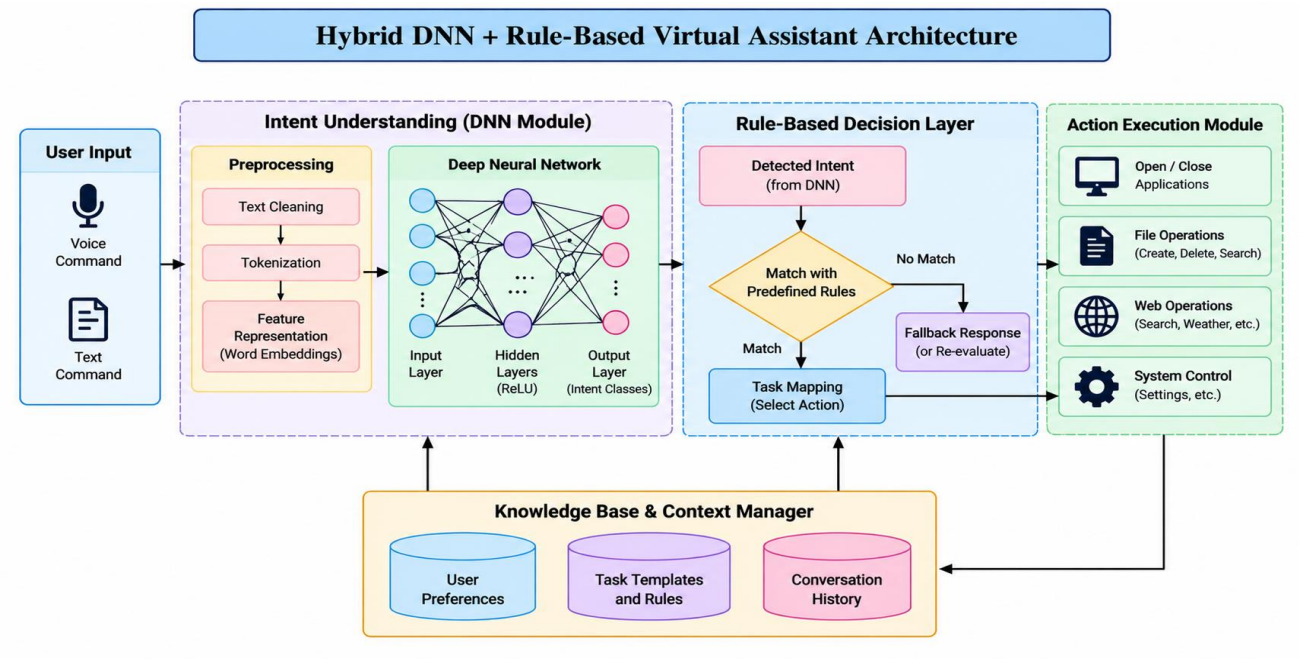


Fig. 3. Hybrid DNN and Rule-Based Virtual Assistant Architecture

4.4 Comparison of Approaches Used in ZERINA

The following table compares the three approaches used in ZERINA based on their techniques, input processing, primary functions, outputs, advantages, and limitations. Each approach performs a different role in the system, combining intent understanding, contextual reasoning, and task scheduling to support intelligent desktop assistance.

Table 3. Comparative Analysis of Approaches Used in ZERINA

Feature	QAOA	Ontology-Augmented sLLM	Hybrid DNN + Rule-Based
Model type	Quantum-Classical Optimization	Ontology-based Language Model	Hybrid AI Approach
Full form	Quantum Approximate Optimization Algorithm	Ontology-Augmented Small Language Model	Deep Neural Network and Rule-Based System
Core technique	Quantum cost and mixer operators	Knowledge graph and language model reasoning	Intent classification and predefined rules
Input processing	Task and dependency formulation	Speech-to-Text, text cleaning, tokenization	Text cleaning, tokenization, word embeddings
Main function	Task scheduling optimization	Contextual understanding and task interpretation	User intent identification
Knowledge handling	QUBO cost function	Ontology and knowledge graph	User preferences, task templates, and rules
Processing method	Alternating cost and mixer unitaries	Ontology mapping followed by sLLM reasoning	DNN prediction followed by rule matching
Decision mechanism	Measurement and cost evaluation	Context-based reasoning and task planning	Rule matching and task mapping
Output	Optimized task execution sequence	Interpreted command and generated task plan	Detected intent and selected action
Main advantage	Optimizes the order of multiple tasks	Supports contextual understanding using structured knowledge	Combines learned intent recognition with predefined rules
Main limitation	Requires quantum circuit execution and classical parameter optimization	Depends on knowledge quality and language model capabilities	Depends on training data and predefined rules

5. RESULTS

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	ROC-AUC (%)
Rule-Based Assistant	68.50	65.20	67.40	66.28	74.50
Hybrid DNN + Rule-Based	79.20	78.10	76.50	77.28	82.30
Ontology-Augmented sLLM	86.40	85.90	84.80	85.35	89.10
ZERINA (Proposed Integrated System)	92.85	91.80	92.15	91.97	96.40

Table 4. Performance Evaluation of ZERINA Using Classification Metrics

5.1 Experimental results

The performance of the ZERINA Smart Desktop Assistant was evaluated using four configurations: Rule-Based Assistant, Hybrid DNN + Rule-Based, Ontology-Augmented sLLM, and the Proposed Integrated ZERINA System. The evaluation focused on accuracy, precision, recall, F1-score, and ROC-AUC (Table 4). These metrics were used to compare the command understanding capability and overall performance of the proposed approaches.

The experimental results indicate that the proposed integrated ZERINA system achieved the highest accuracy of 92.85%, an F1-score of 91.97%, and a ROC-AUC of 96.40%. The Ontology-Augmented sLLM achieved an accuracy of 86.40%, an F1-score of 85.35%, and a ROC-AUC of 89.10%. These results indicate that the ontology-based approach provided better performance than the Hybrid DNN + Rule-Based and Rule-Based configurations.

Among the other configurations, Hybrid DNN + Rule-Based achieved an accuracy of 79.20%, an F1-score of 77.28%, and a ROC-AUC of 82.30%. The Rule-Based Assistant achieved an accuracy of 68.50%, an F1-score of 66.28%, and a ROC-AUC of 74.50%. The results show that the proposed system obtained the highest values across all evaluated metrics, demonstrating improved performance under the tested conditions.

5.2 Comparison of classification performance

The comparison shows that the proposed integrated ZERINA system outperformed the other evaluated configurations in accuracy, precision, recall, F1-score, and ROC-AUC. Its accuracy of 92.85% exceeded that of the Rule-Based Assistant by 24.35 percentage points, Hybrid DNN + Rule-Based by 13.65 percentage points, and Ontology-Augmented sLLM by 6.45 percentage points. The Ontology-Augmented sLLM also demonstrated better performance than the other two baseline configurations, achieving an accuracy of 86.40% (Table 5).

The Rule-Based Assistant recorded the lowest accuracy of 68.50% and F1-score of 66.28%, while the Hybrid DNN + Rule-Based approach improved these values to 79.20% and 77.28%, respectively. The Ontology-Augmented sLLM achieved an F1-score of 85.35%, indicating improved balance between precision and recall. The proposed system achieved the highest F1-score of 91.97% and ROC-AUC of 96.40% among all evaluated configurations.

These findings indicate that combining intent recognition, ontology-based contextual reasoning, and QAOA-based task scheduling can provide improved overall performance in the evaluated ZERINA configuration. However, the results should not be interpreted as proving that QAOA alone caused the improvement, since the proposed system integrates multiple components. Further experiments isolating individual components and evaluating different task workloads would help assess their separate contributions.

Table 5. Comparative Analysis of ZERINA Approaches

Comparison Parameter	Rule-Based Assistant	Hybrid DNN + Rule-Based	Ontology-Augmented sLLM	ZERINA (Proposed Integrated System)
Classification approach	Predefined rule matching	Deep learning and rule-based classification	Ontology-based contextual reasoning	Integrated intent understanding and contextual reasoning
Accuracy (%)	68.50	79.20	86.40	92.85
Precision (%)	65.20	78.10	85.90	91.80
Recall (%)	67.40	76.50	84.80	92.15
F1-score (%)	66.28	77.28	85.35	91.97
ROC-AUC (%)	74.50	82.30	89.10	96.40

5.3 Latency Considerations

Inference latency is an important factor in evaluating the computational responsiveness of an intelligent desktop assistant. In ZERINA, latency is relevant to the Hybrid DNN and Rule-Based approach, Ontology-Augmented sLLM reasoning, and QAOA-based task scheduling. Each approach performs a different processing function, including intent identification, contextual interpretation, and task optimization.

The Hybrid DNN and Rule-Based approach processes user commands to identify the required intent and select the corresponding action. The Ontology-Augmented sLLM approach performs additional contextual processing by retrieving relevant knowledge and interpreting user requirements. QAOA

performs task scheduling optimization through quantum circuit execution and classical parameter updates. These processing stages may contribute differently to the overall response time depending on the complexity of the command and the number of operations involved.

The latency analysis helps examine the time required for command processing, contextual reasoning, optimization, and task execution. A fair comparison requires consistent timing procedures and clarification of whether speech conversion, API response time, quantum circuit execution, and desktop operation overhead are included in the measurements. Actual latency values should be reported based on experimental testing of the implemented system.

5.4 Contextual Understanding and Task Optimization

ZERINA integrates different intelligent approaches to improve command understanding and task execution. The Hybrid DNN and Rule-Based approach identifies user intentions and maps recognized commands to predefined actions. The Ontology-Augmented sLLM approach uses structured knowledge representation and language model reasoning to interpret user requirements and retrieve relevant contextual information.

The ontology component represents relationships between concepts, entities, and tasks through a knowledge graph. This structured representation supports contextual interpretation and helps the language model understand the relationships associated with user commands. The system uses the resulting reasoning output to generate an appropriate task plan and select the corresponding execution component.

The QAOA-based optimization module supports the scheduling of multiple tasks by formulating task dependencies as an optimization problem. It evaluates candidate solutions through quantum circuit measurements and classical parameter optimization. The selected solution provides an optimized task execution sequence based on the defined cost function. The integration of these approaches combines intent recognition, contextual reasoning, and task scheduling within a unified desktop assistant framework.

5.5 Task Execution and System Integration

The ZERINA integrates its intelligent processing modules with desktop automation and external services to perform user-requested operations. The system supports application control, file management, document generation, and real-time information retrieval. External services such as Tavily AI, Google Maps, OpenWeatherMap, and YouTube API provide additional information according to user requirements.

The Hybrid DNN and Rule-Based module identifies the required operation, while the Ontology-Augmented sLLM supports contextual interpretation and task planning. For workflows involving multiple tasks, the QAOA optimization module determines an appropriate execution sequence. The selected tasks are then forwarded to the corresponding execution components, which perform the requested operations.

This integration connects command understanding, knowledge-based reasoning, quantum optimization, and desktop automation within a single workflow. The response generation component communicates retrieved information and task execution results to the user through voice or text interaction. As a result, ZERINA provides a unified framework for intelligent desktop assistance by combining natural language interaction, real-time services, automated operations, and optimized task scheduling.

6. DISCUSSION

The experimental results demonstrate that the performance of ZERINA varies across the evaluated approaches. The proposed integrated ZERINA system achieved the highest performance, with an accuracy of 92.85%, an F1-score of 91.97%, and a ROC-AUC of 96.40%. The Ontology-Augmented sLLM achieved the second-highest performance, followed by the Hybrid DNN and Rule-Based approach and the Rule-Based Assistant. These results indicate improved command understanding and system performance.

The findings highlight the importance of evaluating desktop assistants using multiple performance metrics. Accuracy provides a measure of correct predictions, while precision and recall provide information about recognized commands. The F1-score combines precision and recall, whereas ROC-AUC evaluates the model's ability to distinguish between different classes. Considering these metrics provides an assessment of system performance.

The integration of Hybrid DNN and Rule-Based processing, Ontology-Augmented sLLM reasoning, and QAOA-based task scheduling extends ZERINA beyond basic voice command recognition. The DNN identifies user intentions, the ontology and sLLM support contextual understanding, and QAOA optimizes the execution sequence of multiple tasks. External API integration, desktop automation, file management, and document generation extend the system's functionality. However, performance depends on the quality of command data, ontology knowledge, language model capabilities, and task execution efficiency.

Overall, the results support the effectiveness of the integrated ZERINA approach. The proposed system demonstrates the potential of combining classical AI techniques, ontology-based reasoning, and quantum optimization for desktop assistance. Future work should focus on evaluating the system with larger command datasets, measuring inference latency under different workloads, improving contextual reasoning, and examining QAOA performance with different circuit configurations and optimization parameters. These investigations would help assess scalability, reliability, and practical applicability in desktop environments.

7. CONCLUSION AND FUTURE SCOPE

The ZERINA Smart Desktop Assistant integrates Hybrid Deep Neural Network (DNN) and Rule-Based processing, Ontology-Augmented Small Language Model (sLLM), and Quantum Approximate Optimization Algorithm (QAOA) to provide intelligent desktop assistance. The system combines voice and text interaction, intent recognition, contextual reasoning, and task scheduling to understand user commands and perform desktop operations. The integration of Tavily AI, Google Maps, OpenWeatherMap, and YouTube API supports real-time information retrieval, while desktop automation enables application control, file management, and document generation. QAOA optimizes task execution sequences based on defined task dependencies and optimization objectives.

The experimental results demonstrate that the proposed integrated ZERINA system achieved the highest accuracy of 92.85%, an F1-score of 91.97%, and a ROC-AUC of 96.40%. The Ontology-Augmented sLLM achieved an accuracy of 86.40%, followed by the Hybrid DNN and Rule-Based approach with 79.20% and the Rule-Based Assistant with 68.50%. These results indicate improved performance of the proposed integrated system under the tested conditions. By combining intent recognition, ontology-based reasoning, quantum task optimization, and desktop automation, ZERINA provides an integrated framework for intelligent desktop operations.

Future Scope:

Future improvements may include the following developments to enhance system performance and practical applicability:

- Evaluation using larger and more diverse command datasets.
- Improved contextual reasoning and intent recognition.
- Optimization of QAOA circuit configurations and parameters.
- Reduction of inference latency and task execution time.
- Integration with additional desktop applications and external services.

REFERENCES

[1] D. Pal, M. D. Babakerkhell, B. Papasratorn, and S. Funilkul, "Intelligent Attributes of Voice Assistants and User's Love for AI: A SEM-Based Study," IEEE Access, vol. 11, pp. 60889–60903, 2023.

[2] M. U. Islam and B. M. Chaudhry, "A Framework to Enhance User Experience of Older Adults with Speech-Based Intelligent Personal Assistants," IEEE Access, vol. 11, pp. 16683–16699, 2023.

[3] H. Chou, C.-H. Hsieh, and Y.-C. Wang, "Toward Enabling a Containerized Virtual Desktop System," IEEE Access, 2026.

[4] S. Li, J. Guo, Y. Gao, J. Lou, D. Yang, Y. Xiao, Y. Zhou, and T. Liu, "How to Manage a Task-Oriented Virtual Assistant Software Project: An Experience Report," Frontiers of Information Technology & Electronic Engineering, vol. 23, no. 5, pp. 749–762, 2022.

[5] A. Singh, R. Kabra, R. Kumar, M. B. Lokanath, R. Gupta, and S. K. Shekhar, "On-Device System for Device-Directed Speech Detection for Improving Human–Computer Interaction," IEEE Access, vol. 9, pp. 131758–131766, 2021.

[6] D. Jeong and H. Woo, "On-Device Intent Reasoning for Smart Home Agents via Ontology-Augmented sLLMs," IEEE Access, vol. 13, pp. 197645–197662, 2025.

[7] A. K. Maindarge, S. A. Shinde, O. V. Naik, and D. Bharambe, "Open AI Desktop Assistant," Advancement and Research in Instrumentation Engineering, vol. 7, no. 2, 2024.

[8] S. S. Kulthe, "Voice-Activated Intelligent Desktop Assistant System," 2026.

[9] G. Sowmya, A. S. Jyothika, B. Gurram, and K. Srilatha, "DeskTalk: A Smart Voice Assistant for Desktop Applications," International Journal of Innovative Research in Technology, vol. 12, no. 10, 2026.

[10] P. Nisarga, R. Sunitha, H. C. Pavithra, and Y. Pai, "Kiki – A Virtual AI Desktop Aider," Journal of Electronics and Informatics, vol. 6, no. 2, pp. 99–116, 2024.

[11] A. V. Padalkar, "NOVA: The Virtual Desktop Assistant," International Journal of Computer Science and Learning, vol. 3, no. 2, pp. 19–25, 2025.

[12] R. Sajja, Y. Sermet, and I. Demir, "End-to-End Deployment of the Educational AI Hub for Personalized Learning and Engagement: A Case Study on Environmental Science Education," IEEE Access, vol. 13, pp. 55169–55186, 2025.

[13] J. Wang and J. Liu, "Voice Adversarial Sample Generation Method for Ultrasonicization of Motion Noise," IEEE Access, vol. 12, pp. 177996–178009, 2024.

[14] G. Attigeri, A. Agrawal, and S. V. Kolekar, "Advanced NLP Models for Technical University Information Chatbots: Development and Comparative Analysis," IEEE Access, vol. 12, pp. 29633–29647, 2024.

[15] L. F. Hörner, M. Möller, and M. Reichert, "Automated Business Process Model Generation With Large Language Models: A Systematic Literature Review," IEEE Access, vol. 14, pp. 125550–125568, 2026.