FMDB *Transactions on Sustainable Computer Letters*

A Multimodal Transformer-Based Digital Companion for Emotion-Aware Human–Computer Interaction Using Real-Time 3D Avatars

A. Shyam Sundhar[1], J. Angelin Jeba[2,*], Varsha Ramkumar[3], M. Rehena Sulthana[4], Rejwan Bin Sulaiman[5], R. P. P. Ranaweera[6]

[1]Department of Information and Communication, Tampere University, Tampere, Pirkanmaa, Finland.
[2]Department of Electronics and Communication Engineering, S.A. Engineering College, Chennai, Tamil Nadu, India.
[3]Department of Business Analytics, University of Galway, Galway, Ireland.
[4]School of Information Technology and Engineering, Melbourne Institute of Technology, Melbourne, Victoria, Australia.
[5]Department of Computer Science and Technology, Northumbria University, London, England, United Kingdom.
[6]National Institute of Library and Information Sciences, University of Colombo, Colombo, Sri Lanka.
shyamsundhr26@gmail.com[1], angelinjeba@saec.ac.in[2], v.ramkumar1@universityofgalway.ie[3], rsulthana@academic.mit.edu.au[4], rejwan.sulaiman@northumbria.ac.uk[5], prasanna@nilis.cmb.ac.lk[6]

Abstract: Human-computer interaction has progressed significantly in recent years, yet existing digital assistants continue to suffer from limitations in emotional engagement, personalisation, and expressive communication. Most current AI systems operate primarily through text or voice, lacking a visual presence that allows users to form meaningful connections with technology. To address this gap, our paper introduces JARWIN (Just A Real-World Intelligent Network). This next-generation AI companion integrates a large language model, speech processing, and real-time 3D avatar animation to create a more natural, interactive, and intelligent user experience. The architecture features multiple operational tiers, including speech-to-text transcription, sentiment-aware conversational modelling, text-to-speech synthesis, and avatar-based emotional expression. Linear conversational modelling serves as the initial baseline for contextual understanding, while more advanced transformer-based models are employed to handle nuanced dialogue, sentiment variations, and continuous memory retention. Additionally, emotion recognition mechanisms dynamically map conversational tone to facial expressions and gestures, enabling human-like responsiveness. The result is a system capable not only of executing functional tasks, such as opening applications and retrieving information, but also of maintaining adaptive, emotionally aware dialogue. By bridging cognitive intelligence with visual embodiment, this paper demonstrates a scalable, innovative approach to immersive digital companionship, offering new opportunities for personalised assistance, interactive learning, mental wellness support, and human-AI relational computing.

Keywords: Machine Learning (ML); Resilient Food Chain; Economic Loss; Linear Regression; Random Forest; Digital Companionship; Prediction Accuracy; Interactive Learning.

Received on: 30/07/2025, **Revised on:** 25/10/2025, **Accepted on:** 02/11/2025, **Published on:** 09/08/2026

Journal Homepage: https://www.fmdbpub.com/user/journals/details/FTSCL

DOI: https://doi.org/10.69888/FTSCL.2026.000752

Cite as: A. S. Sundhar, J. A. Jeba, V. Ramkumar, M. R. Sulthana, R. B. Sulaiman, and R. P. P. Ranaweera, "A Multimodal Transformer-Based Digital Companion for Emotion-Aware Human–Computer Interaction Using Real-Time 3D Avatars," *FMDB Transactions on Sustainable Computer Letters*, vol. 4, no. 3, pp. 178–193, 2026.

[*]Corresponding author.

1. Introduction

Artificial Intelligence (AI) has entered an era of extraordinary transformation, moving far beyond its origins in simple data analysis and predictive algorithms. What began as a collection of tools for processing numbers and automating routine tasks has evolved into a field capable of understanding context, generating language, and simulating aspects of human thought and emotion. From voice assistants that can schedule appointments to chatbots that provide instant customer service, AI has already become an integral part of daily life. Yet these current systems remain fundamentally limited. Most operate within a single channel of interaction—text or voice—and lack the depth of presence, expression, and adaptability that define truly human communication [1]. The concept of JARWIN (Just A Real-World Intelligent Network) emerges as a response to these limitations, proposing a new paradigm for AI interaction. JARWIN is envisioned as an intelligent agent that merges cognitive reasoning with a visual, emotionally expressive presence. At its core, JARWIN is powered by advanced large language models (LLMs) that enable deep understanding, natural conversation, and continual learning from user input.

But unlike conventional AI systems confined to text windows or audio responses, JARWIN extends into the visual realm through a three-dimensional, interactive avatar. This avatar does more than speak—it conveys emotions, gestures, and subtle nonverbal cues, allowing users to engage with technology in a way that feels personal and alive. This introduction of a Python-based AI platform marks a significant step forward in human–machine interaction. By combining natural language processing, real-time voice synthesis, and 3D rendering, JARWIN aims to create an experience where the boundaries between digital assistant and digital companion begin to blur. The system is designed not only to execute commands or provide information, but also to adapt its personality, learn from each interaction, and express empathy that enhances trust and engagement. As AI continues to shape the future of communication, JARWIN represents a bold vision of what comes next: an intelligent network that does more than compute or predict—it connects [4]. Through its fusion of cognitive intelligence and physical representation, JARWIN offers a glimpse into a future where AI is not merely a tool, but an interactive presence capable of meaningful participation in everyday life.

2. Literature Review

Vaswani et al. [1] introduced the Transformer architecture, which revolutionised natural language processing by replacing recurrent neural networks with self-attention mechanisms. They proposed a novel model architecture based entirely on attention mechanisms, eliminating the need for recurrence and convolutions. The experimental design demonstrated that Transformers achieve superior performance on machine translation tasks while being significantly more parallelizable and requiring less training time compared to RNN-based models. Their findings indicate that the Transformer model achieved state-of-the-art BLEU scores on English-to-German and English-to-French translation tasks, establishing the foundation for modern large language models. The research emphasises that attention mechanisms alone are sufficient for sequence transduction tasks, paving the way for models like GPT and BERT that power conversational AI systems like JARWIN. Brown et al. [2] explored the few-shot learning capabilities of large language models by developing GPT-3, a 175-billion-parameter autoregressive language model. They demonstrated that scaling language models to unprecedented sizes enables them to perform a wide variety of NLP tasks without task-specific fine-tuning, using only a few examples provided in the prompt. The experimental design evaluated GPT-3 on dozens of NLP benchmarks in zero-shot, one-shot and few-shot settings across tasks including translation, question answering and arithmetic. Their findings indicate that GPT-3 achieves competitive performance with fine-tuned models on many tasks and demonstrates strong few-shot learning abilities that improve with model scale.

The research emphasises that large language models can serve as general-purpose text interfaces, making them ideal foundations for conversational AI assistants like JARWIN that require contextual understanding and adaptability without extensive task-specific training. Touvron et al. [3] addressed the accessibility of large language models by developing LLaMA, a collection of open-source foundation models ranging from 7B to 65B parameters trained on publicly available datasets. They proposed releasing efficient models that match or exceed the performance of proprietary models while being small enough to run on consumer hardware. The experimental design employed training on 1.4 trillion tokens from diverse text sources and evaluation across multiple benchmarks, including common-sense reasoning, reading comprehension, and mathematical reasoning. Their findings indicate that LLaMA-13B outperforms GPT-3 (175B) across most benchmarks, and LLaMA-65B is competitive with leading models such as Chinchilla-70B and PaLM-540B. The research emphasises that democratizing access to powerful language models enables researchers and developers to build sophisticated AI applications like JARWIN without requiring massive computational resources or access to proprietary APIs. Radford et al. [4] introduced CLIP (Contrastive Language-Image Pre-training), a model that learns visual concepts from natural-language supervision by jointly training an image encoder and a text encoder to predict correct image-text pairings. They proposed a scalable method for learning transferable visual representations that can perform zero-shot classification on diverse image datasets. The experimental design involved training on 400 million image-text pairs collected from the internet and evaluating on over 30 computer vision datasets without dataset-specific fine-tuning.

Their findings indicate that CLIP achieves competitive zero-shot performance compared to fully supervised models and demonstrates impressive robustness to distribution shifts. The research emphasises the potential for multimodal models that understand both vision and language, providing a foundation for future enhancements to JARWIN that could enable visual perception capabilities, allowing the assistant to understand and respond to images, objects, and visual contexts in its environment. Radford et al. [5] demonstrated that language models trained on diverse internet text can perform multiple NLP tasks without explicit supervision by developing GPT-2, a 1.5-billion-parameter transformer model. They showed that unsupervised pre-training on large text corpora enables models to learn task-specific behaviours through pattern recognition in the training data. The experimental design involved training on 40GB of internet text (the WebText dataset) and evaluating zero-shot performance on reading comprehension, translation, summarisation, and question-answering tasks. Their findings indicate that GPT-2 achieves state-of-the-art results on several language modelling benchmarks and demonstrates surprising zero-shot task transfer capabilities. The research emphasises that language models can function as unsupervised multitask learners, supporting JARWIN's ability to handle diverse conversational tasks and user requests without requiring task-specific training for each possible interaction scenario. Devlin et al. [6] revolutionised natural language understanding by introducing BERT (Bidirectional Encoder Representations from Transformers), which uses bidirectional transformer training to pre-train deep representations from unlabelled text. They proposed a masked language modelling objective that allows the model to learn contextualised word representations by predicting randomly masked tokens based on both left- and right-context.

The experimental design employed pre-training on BooksCorpus and English Wikipedia, followed by fine-tuning on 11 NLP tasks, including question answering, sentiment analysis, and textual entailment. Their findings indicate that BERT obtains state-of-the-art results on all tasks, with improvements ranging from 4% to 20% over previous best systems. The research emphasises the importance of bidirectional context for language understanding, making BERT-based models ideal for JARWIN's sentiment analysis module, which requires deep understanding of emotional nuances and contextual meaning in user inputs. Chase [7] developed LangChain, a framework that simplifies the creation of applications using large language models through composability and modular components. The framework provides abstractions for prompt management, memory systems, chain construction, and agent behaviours, enabling developers to build complex LLM-powered applications. The implementation includes modules for conversational memory, document retrieval, tool usage, and multi-step reasoning chains that can be composed into sophisticated workflows. The framework's design enables applications to maintain conversation history, access external data sources, and execute multi-step tasks while leveraging LLMs' natural language understanding capabilities. The research emphasises the importance of memory and context management in conversational AI, directly supporting JARWIN's ability to maintain coherent multi-turn conversations, recall previous interactions, and provide personalised responses based on accumulated user interaction history. Radford et al. [8] addressed the challenge of robust automatic speech recognition by developing Whisper, a transformer-based model trained on 680,000 hours of multilingual, multitask, and supervised data collected from the internet.

They proposed training speech recognition models on diverse, weakly supervised data at scale to achieve robustness across languages, accents, and acoustic conditions. The experimental design involved training encoder-decoder transformers on transcribed audio from multiple sources and evaluating on diverse speech recognition benchmarks including noisy environments and non-native speakers. Their findings indicate that Whisper approaches human-level robustness and accuracy on English speech recognition while also demonstrating strong multilingual capabilities across 96 languages. The research emphasises that large-scale weak supervision enables speech recognition systems to generalise effectively across diverse conditions, making Whisper ideal for JARWIN's speech-to-text module, which must handle various user accents, speaking styles, and environmental noise conditions to provide reliable voice interaction. Pavithra et al. [9] explored real-time voice cloning using deep learning techniques to generate natural-sounding speech that mimics specific speaker characteristics from limited reference audio. They proposed neural network architectures that can learn speaker embeddings from short audio samples and condition speech synthesis models with these embeddings. The experimental design involved training on multi-speaker datasets and evaluating the quality and similarity of synthesised speech to target speakers using both objective metrics and human perception studies. Their findings indicate that deep learning-based voice cloning can produce high-quality, speaker-specific speech from just seconds of reference audio with minimal computational overhead during inference. The research emphasises the potential for personalised voice synthesis, supporting JARWIN's goal of providing customizable voice characteristics that enable users to personalise their AI assistant's vocal identity, thereby enhancing a sense of ownership and emotional connection to their digital companion.

Shen et al. [10] developed Tacotron 2, an end-to-end neural text-to-speech system that synthesises natural-sounding speech directly from text using a sequence-to-sequence model with an attention mechanism followed by a WaveNet vocoder. They proposed a two-stage approach where mel-spectrograms are predicted from text, then converted to time-domain waveforms using a modified WaveNet architecture. The experimental design involved training on a single-speaker dataset and evaluating using mean opinion score (MOS) tests comparing synthetic speech to natural human recordings. Their findings indicate that Tacotron 2 achieves a MOS of 4.53, approaching the quality of natural speech (4.58), representing a significant advancement in neural TTS quality. The research emphasises that neural approaches can generate speech that is nearly indistinguishable from human speech, making these models suitable for JARWIN's text-to-speech module, where natural-sounding, emotionally

expressive voice output is essential for creating engaging, human-like interactions. Kim et al. [11] introduced VITS (Variational Inference with Adversarial Learning for End-to-End Text-to-Speech). This parallel end-to-end TTS method combines variational inference, normalising flows, and adversarial training to generate high-quality speech efficiently. They proposed a single-stage model that directly generates raw waveforms from text, without requiring intermediate representations such as mel-spectrograms.

The experimental design involved training on single- and multi-speaker datasets and comparing synthesis quality and speed with two-stage TTS systems such as Tacotron 2 with WaveNet. Their findings indicate that VITS achieves naturalness comparable to or superior to that of existing models while being significantly faster, enabling real-time generation. The research emphasises the importance of efficient, high-quality speech synthesis for interactive applications, directly supporting JARWIN's requirement for low-latency, natural-sounding text-to-speech that can maintain conversational flow without noticeable delays between text generation and audio playback. Schröder and Trouvain [12] presented MARY (Modular Architecture for Research on speech synthesis), an open-source text-to-speech synthesis system designed for research, development, and teaching. They proposed a modular architecture that separates different stages of TTS processing (text analysis, prosody prediction, and waveform generation) into independent components that can be modified or replaced. The experimental design focused on creating a flexible platform that researchers could extend and customise for various speech synthesis experiments and applications. Their findings emphasise that modular TTS systems enable experimentation with different linguistic processing strategies, prosody models, and synthesis techniques. The research emphasises the value of open-source, modular approaches to speech technology, aligning with JARWIN's architecture philosophy of using modular components that can be independently developed, tested, and upgraded, ensuring the system remains adaptable to evolving technologies and user requirements. Plappert et al. [13] developed the KIT Motion-Language Dataset, which provides aligned motion capture data and natural-language descriptions to facilitate research on motion synthesis, understanding, and human-robot interaction.

They created a comprehensive dataset containing 3,911 motion sequences performed by various subjects with corresponding textual annotations describing the actions. The experimental design involved recording full-body motion capture data synchronised with natural language descriptions and developing baseline methods for motion retrieval and synthesis from text. Their findings demonstrate that aligned motion-language datasets enable machine learning models to learn mappings between linguistic descriptions and physical movements. The research emphasises the importance of grounded language understanding that connects words to physical actions, supporting future enhancements to JARWIN in which natural language commands could be translated into avatar gestures and movements beyond predefined animations, enabling more expressive and context-appropriate physical embodiment. Mousas and Anagnostopoulos [14] addressed the challenge of generating realistic finger and hand movements for virtual characters by proposing machine learning methods that learn motion features from example animations. They developed techniques for estimating detailed finger motion that complement full-body animations, enhancing the realism of virtual character interactions. The experimental design involved extracting motion features from motion capture databases and training models to predict finger movements based on body pose and context. Their findings indicate that learning-based approaches can generate natural-looking hand gestures that enhance the expressiveness of virtual characters without requiring explicit finger motion capture for every animation. The research emphasises the importance of detailed, realistic animation for creating believable virtual characters, directly supporting JARWIN's avatar system where subtle gestures like hand movements, pointing, and finger articulation could enhance communication effectiveness and make the avatar appear more lifelike and engaging during interactions.

Ionescu et al. [15] introduced Human3.6M, a large-scale dataset for 3D human pose estimation and action recognition containing 3.6 million 3D human poses with corresponding images captured from multiple camera viewpoints. They developed comprehensive benchmarks for evaluating 3D human pose estimation algorithms and provided baseline methods for pose prediction and action recognition. The experimental design involved recording 11 professional actors performing 17 common activities from four synchronised camera angles with accurate 3D joint positions obtained through motion capture. Their findings established performance baselines for various computer vision tasks related to human pose and motion understanding. The research emphasises the importance of large-scale datasets for training robust computer vision models, providing foundational data that could support future enhancements to JARWIN's visual perception capabilities, enabling the system to understand user body language, gestures, and physical context through camera inputs, adding another dimension to multimodal human-AI interaction beyond voice and text. The studies reviewed collectively highlight the growing importance of emotionally intelligent, visually embodied AI systems for enhancing human-computer interaction, digital assistance, and personalised communication. Existing research in conversational AI, speech processing, and avatar-based interfaces demonstrates that large language models (LLMs) such as GPT, LLaMA, and Falcon can generate context-aware responses and maintain natural dialogue flow. These capabilities enable AI systems to move beyond simple command execution toward forming adaptive, relational interactions with users.

Similarly, advancements in 3D modelling, facial animation synthesis, and real-time rendering technologies have enabled the development of avatars that can display expressions, gestures, and personality traits, thereby improving user engagement and

trust. Emotion recognition models and sentiment classification frameworks also enable the system to detect tone and emotional states, thereby promoting empathetic response behaviours in digital companions. Despite these advancements, current systems often remain limited in realism, expressiveness, and context generalisation. Many implementations are constrained to specific environments or lack continuity in memory and long-term user adaptation. Additionally, maintaining real-time synchronisation between speech output and avatar animation requires computational resources and efficient optimisation strategies. While multimodal systems have demonstrated significant potential, integrating conversational reasoning, emotional intelligence, and visually interactive interfaces still poses challenges in latency management, consistency in personalisation, and scalability across diverse platforms [5]. Overall, the literature indicates strong promise for systems like JARWIN but emphasises the need for more integrated, flexible, and user-adaptive frameworks to improve interaction quality and emotional authenticity. The development of scalable architectures that unify language understanding, sentiment alignment, and expressive embodiment is crucial for advancing AI companionship and delivering meaningful, human-centred digital experiences.

3. Methodology

Figure 1 illustrates a sophisticated multi-engine artificial intelligence system that operates through a carefully orchestrated sequence of initialisation, input processing, intelligent routing, and response generation phases. The system begins with a comprehensive initialization process where multiple specialized engines are loaded and configured simultaneously, including the AI reasoning engine powered by Groq's language models, the voice processing engine that integrates both ElevenLabs professional voice cloning and pyttsx3 system text-to-speech, the machine learning prediction engine containing twenty-three regression algorithms, the vision analysis engine utilizing Google's Gemini API, the document processing engine for handling various file formats, and the Retrieval-Augmented Generation system for maintaining conversational context and knowledge persistence. This initialisation phase performs critical system health checks to verify API connectivity, validate configuration files, and ensure all dependencies are properly loaded before allowing user interaction. If any critical component fails during initialisation, the system gracefully handles the error by providing detailed diagnostic information to help users resolve configuration issues, particularly missing API keys or incorrect environment variable settings. Once all engines successfully initialise, the system enters a ready state. It begins accepting user input across multiple modalities, demonstrating its flexibility and accessibility by supporting traditional text commands, voice-based speech recognition via Google's Speech API for hands-free operation, and direct file uploads for document analysis, image recognition, or dataset processing.

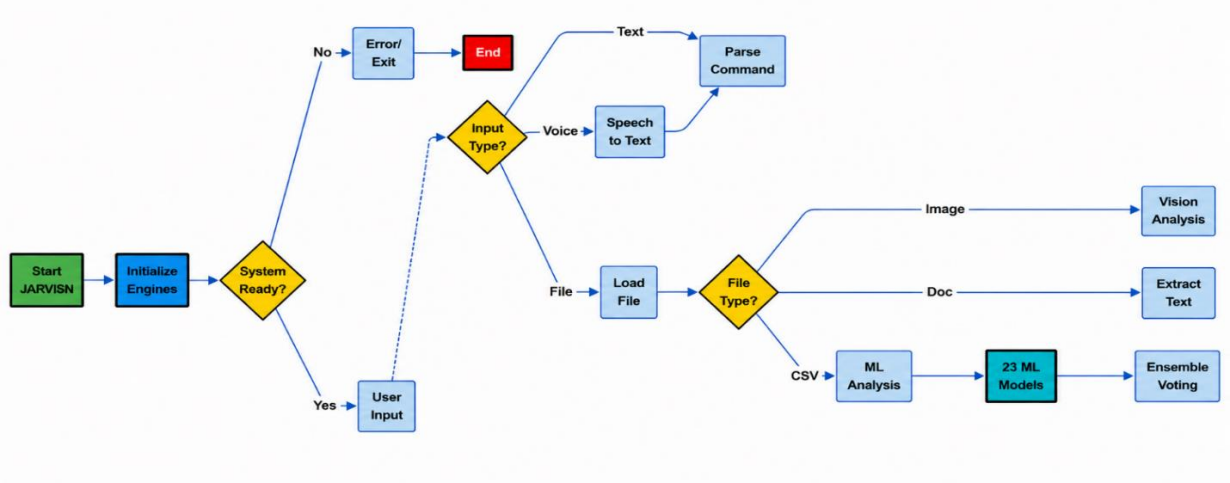

Figure 1: Workflow diagram

The core intelligence of JARWIN manifests in its dynamic command routing and processing pipeline, which automatically analyses incoming user requests to determine the optimal processing pathway based on input characteristics and user intent. When users submit queries, the system first passes them through a sophisticated parsing mechanism that identifies whether the input represents a system command requiring immediate execution, a file analysis task necessitating specialised processing, a voice control instruction for managing audio settings, or a general conversational query requiring natural language understanding. For file-based inputs, the system employs intelligent type detection to route images through the vision analysis engine for detailed visual interpretation, documents through text extraction and summarisation pipelines, and CSV datasets through a comprehensive machine learning analysis workflow that leverages all 23 regression algorithms in an ensemble to provide robust predictions with statistical confidence intervals. General queries trigger the RAG retrieval system, which searches through stored conversational history and accumulated knowledge to provide relevant context before forwarding the enriched prompt to the AI processing core, where Groq's advanced language models generate coherent, contextually appropriate responses. The generated responses then flow through a memory storage phase where they are persisted in the conversation

database along with metadata including processing time, response length, and source attribution, enabling the system to learn from interactions and improve future responses. Finally, the output phase presents results to users through both visual display and optional voice synthesis, where users can experience their personalised cloned voice through ElevenLabs or rely on system-level text-to-speech as a fallback, before the system loops back to await the next user input, creating a continuous, responsive interaction cycle that adapts to user preferences and maintains context across extended conversation sessions.

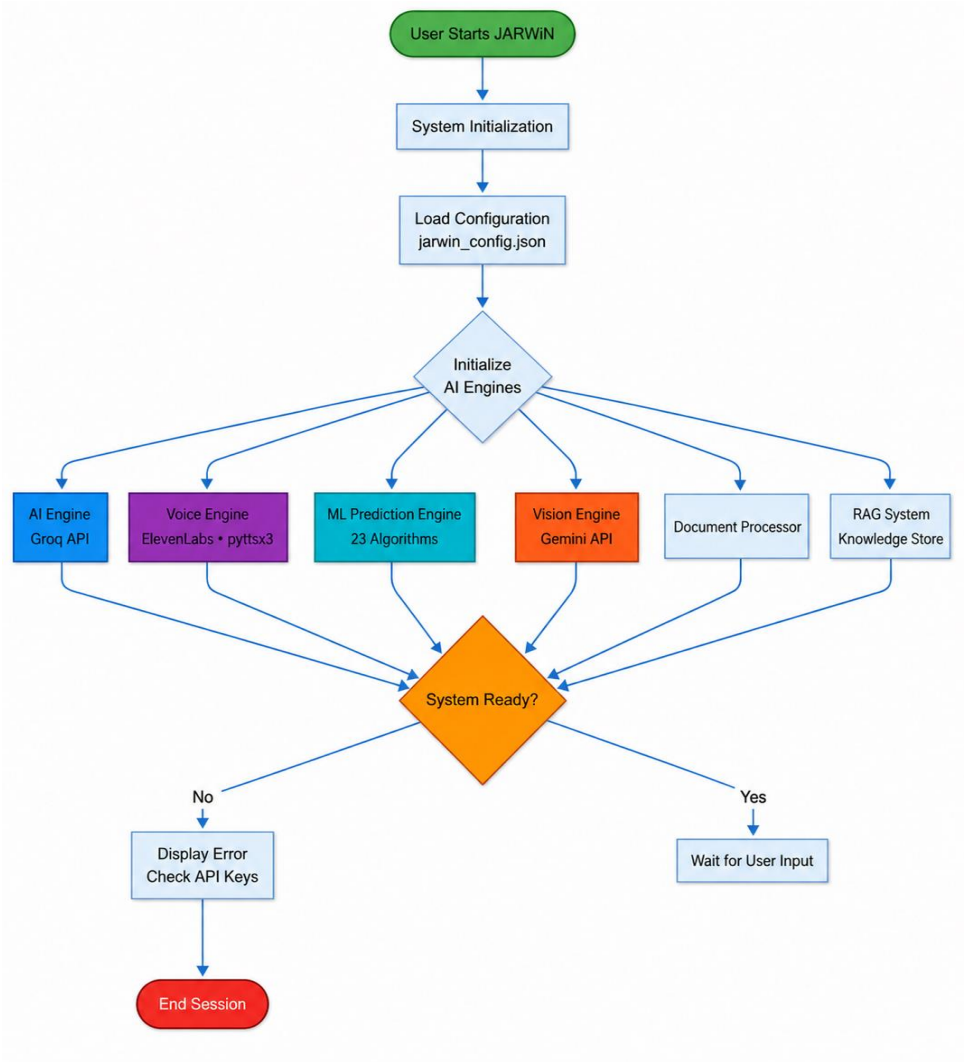

Figure 2: Architecture diagram

The development and implementation of JARWIN follow a modular, object-oriented design methodology that emphasises component independence, fault tolerance, and seamless integration of multiple artificial intelligence technologies through well-defined interfaces and abstraction layers. The system architecture is constructed using Python as the primary programming language, leveraging its extensive ecosystem of libraries including scikit-learn for implementing twenty-three regression algorithms ranging from linear models like Ridge and Lasso to ensemble methods such as Random Forest and Gradient Boosting, pandas and numpy for efficient data manipulation and numerical computation, and specialized API clients for external services including Groq for natural language processing, ElevenLabs for professional voice synthesis, and Google's Gemini for vision capabilities. The core methodology employs a layered architecture in which the base layer consists of independent engine classes: VoiceEngine, MLPredictionEngine, EnhancedAI, VisionEngine, DocumentProcessor, and EnhancedRAG, each encapsulating specific functionality and adhering to standardised initialisation, error-handling, and resource-management protocols. These engines communicate through a central IntelligentEngine orchestrator that implements intelligent routing logic using conditional decision trees and type detection algorithms to direct user inputs to appropriate processing pipelines based on content analysis and command parsing. The voice integration methodology implements a priority-based fallback system in which ElevenLabs voice cloning is attempted first by initialising the client with user-configured API keys and voice IDs, and generating audio via text-to-speech.

The Convert API endpoint supports multilingual models, saves output as MP3 files, and plays audio through multiple backends, including pygame, playsound, and operating system-specific commands, with automatic fallback to pyttsx3 system voices if network connectivity fails or API quotas are exceeded. Error handling throughout the system follows a comprehensive exception management strategy, with try-catch blocks surrounding all external API calls and detailed logging to both files and the console. Figure 2 showcases the architecture of the JARWIN system, which begins with the comprehensive design and integration of multiple AI components working in synergy to create a sentient-like digital assistant. The development follows a modular pipeline approach, where each module is independently constructed, tested, and then integrated into a unified system capable of real-time interaction, emotional responsiveness, and system-level control. This methodology systematically addresses the complete lifecycle of an interactive AI assistant, from avatar design and natural language understanding to voice synthesis and system automation, while ensuring seamless coordination between visual representation, cognitive processing, and emotional intelligence. The methodology adopted for JARWIN is designed to bridge the gap between functional AI tools and human-like virtual companions by integrating cutting-edge technologies in 3D graphics, natural language processing, sentiment analysis, and speech recognition. Unlike traditional voice-based assistants that lack visual embodiment and emotional depth, JARWIN provides a multi-modal interaction experience that combines visual presence through a customizable 3D avatar, conversational intelligence powered by Large Language Models, and personalised responses driven by sentiment analysis and memory modules. This hybrid approach ensures that JARWIN is not only technically robust but also research-oriented, offering contributions toward human-computer interaction, emotional AI, and the future of digital companions in metaverse and AR/VR environments.

3.1. 3D Model Integration and Animation Pipeline

Figure 3 represents the model integration, where the process begins with the design and creation of a three-dimensional avatar that serves as the visual representation of JARWIN.

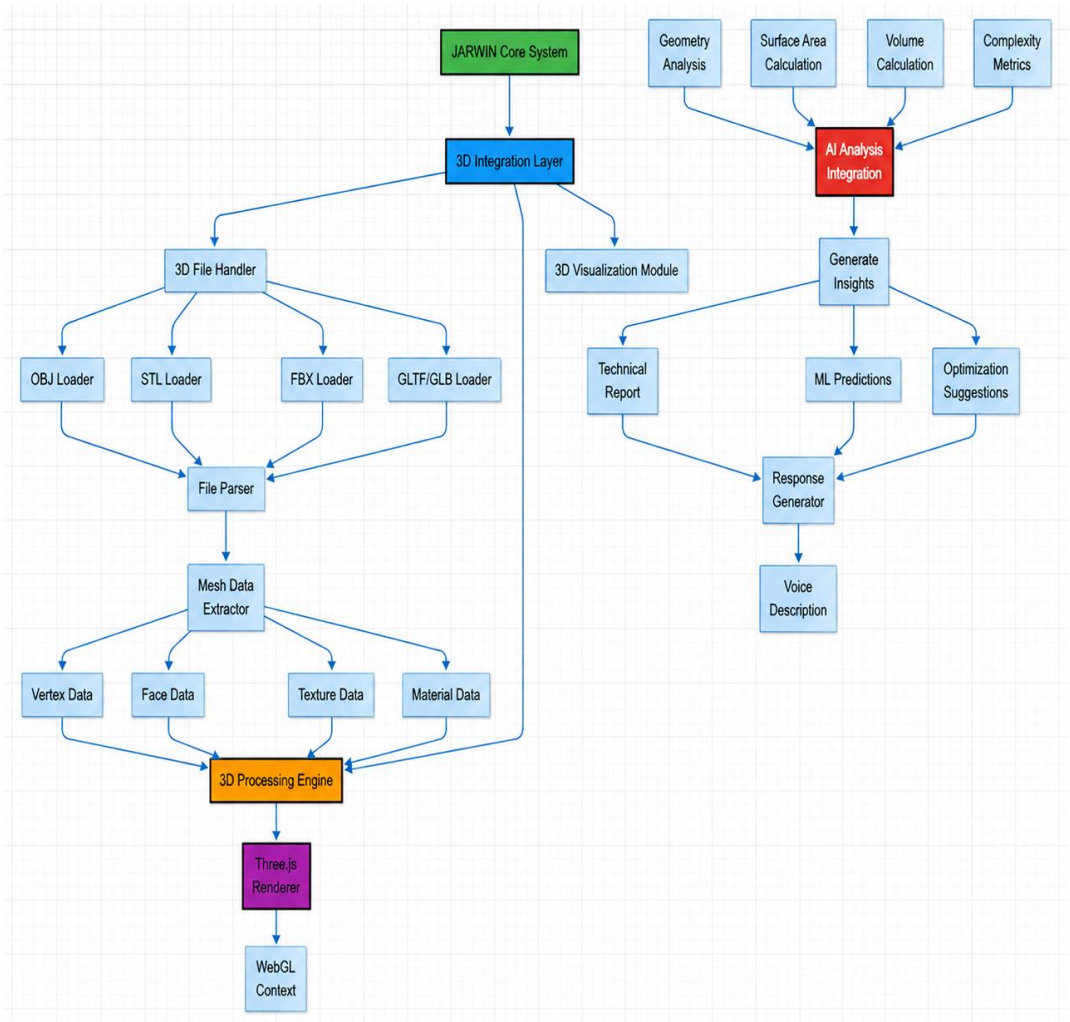

Figure 3: 3D model integration diagram

The avatar is highly customizable, allowing users to personalise appearance, clothing, and physical characteristics according to their preferences. Ready Player Me, a widely used platform for generating realistic humanoid avatars, is utilised as the primary tool for avatar creation. Alternatively, Blender, an open-source 3D modelling software, may be used by users who require more granular control over the avatar's design features. The avatar models are exported in standard 3D formats such as FBX or GLB, ensuring compatibility with animation tools and rendering engines. Once the base avatar is created, it undergoes an animation pipeline using Mixamo, a cloud-based service that provides motion-capture libraries and automatic rigging. Mixamo allows for the seamless application of pre-recorded human motions to the avatar, including walking, gesturing, nodding, and various idle animations. These animations are essential for making the avatar appear lifelike and responsive during interactions. Facial expressions are another critical component of the avatar's design, as they convey emotional states and enhance user engagement. Blend shapes and morph targets are used to create expressions such as smiling, frowning, surprise, and neutral states. These expressions are mapped to emotional tags derived from sentiment analysis of user inputs, ensuring that the avatar's facial reactions align with the conversational context. The avatar's rendering is handled through PyOpenGL, a Python binding for OpenGL that enables real-time 3D graphics rendering. For projects requiring more advanced visual fidelity or cross-platform deployment, the Unity engine can be integrated via a Python-Unity bridge, enabling the avatar to be rendered in Unity while maintaining control logic in Python.

3.2. Natural Language Understanding and Conversational Intelligence

Figure 4 illustrates JARWIN's cognitive intelligence, powered by Large Language Models (LLMs) that underpin its conversational capabilities. The system leverages pre-trained transformer-based models from the GPT or LLaMA families, accessed either through cloud-based APIs such as OpenAI's GPT-4 or via local inference using open-source alternatives like LLaMA 2 or Falcon models hosted on HuggingFace. Local inference is preferred for privacy-sensitive applications, as it eliminates the need for external API calls and ensures that user data remains on-device. The LLM is responsible for understanding user queries, maintaining conversational context, and generating coherent, contextually appropriate responses. To enhance interaction quality, prompt engineering techniques are used to guide the LLM's behaviour. System prompts define JARWIN's personality traits, tone, and conversational style, ensuring that responses feel consistent and personalised. For example, JARWIN may be configured to adopt a friendly, supportive tone, or a professional, informative demeanour depending on user preferences.

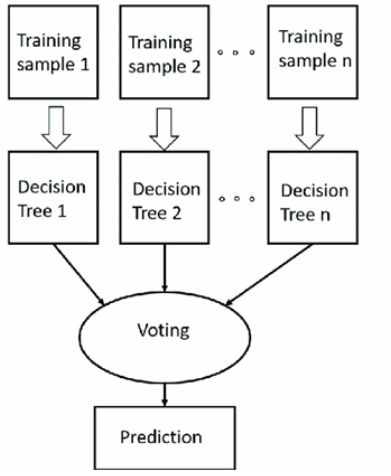

Figure 4: Random forest diagram

Context management is achieved by integrating LangChain, a framework for building LLM-based applications that maintains memory across multiple conversational turns. LangChain's memory modules store previous interactions, allowing JARWIN to recall earlier parts of the conversation and provide responses that reference past exchanges. This capability is essential for creating a sense of continuity and personalisation, making interactions feel more natural and human-like. Memory buffers are periodically summarised to prevent excessive context length, which can degrade model performance and increase inference latency. Sentiment analysis is integrated as a parallel module to detect the emotional tone of user inputs. Pre-trained sentiment classifiers, such as BERT-based models fine-tuned on emotion-detection datasets, are used to categorise user messages into emotional states, including happy, sad, angry, neutral, and surprised. The detected sentiment is then used to modulate JARWIN's responses in two ways: adjusting the textual content to match the user's emotional state (e.g., offering comfort during sadness or celebrating during happiness) and triggering corresponding facial expressions and gestures on the 3D avatar. This

multi-modal emotional responsiveness significantly enhances user engagement and creates a more immersive interaction experience.

3.3. Voice Interaction System: Speech Recognition and Synthesis

Voice interaction is a fundamental component of JARWIN, enabling users to communicate naturally without relying solely on text input. The voice interaction system comprises two primary subsystems: Speech-to-Text (STT), which converts user speech into text, and Text-to-Speech (TTS), which generates JARWIN's spoken responses. Speech-to-Text functionality is implemented using OpenAI Whisper, a state-of-the-art automatic speech recognition model known for its high accuracy and robustness across diverse accents and noisy environments. Whisper operates in real time or near real time, capturing audio from the user's microphone and transcribing it into text with accuracy exceeding 95% under optimal conditions. The transcribed text is then passed to the LLM for processing. Alternative STT solutions, such as Google Speech Recognition or Mozilla DeepSpeech, may be used depending on deployment requirements and privacy considerations. Text-to-Speech synthesis is handled through multiple options to accommodate varying needs for voice quality, latency, and computational resources.

For offline applications that require low latency and minimal resource consumption, pyttsx3, a Python library that provides basic TTS functionality using system-native speech engines, is used. While pyttsx3 offers fast response times, its voice quality is relatively robotic. For applications that demand higher voice realism, Coqui TTS or the ElevenLabs API is integrated. Coqui TTS is an open-source neural TTS framework that generates natural-sounding voices with customizable pitch, speed, and tone. ElevenLabs, on the other hand, offers cloud-based TTS with highly realistic voice synthesis, including the ability to clone user-specified voices. The TTS system is designed to operate asynchronously, allowing audio playback to begin while the LLM continues generating text, thereby reducing perceived latency. Voice activity detection (VAD) automatically detects when the user begins and stops speaking, enabling hands-free interaction without manual activation. The VAD module analyses audio energy levels and spectral features to distinguish speech from background noise, triggering the STT pipeline only when speech is detected. This approach minimises unnecessary processing and improves system responsiveness.

3.4. System-Level Task Automation and Control

A defining feature of JARWIN is its ability to perform system-level operations based on natural language commands. This capability transforms JARWIN from a passive conversational agent into an active assistant that can interact with the underlying operating system and installed applications. The system automation module is built using Python libraries such as pyautogui, os, subprocess, and psutil, which provide programmatic access to system functions. JARWIN can execute a wide range of tasks, including opening and closing applications, navigating file systems, playing media files, adjusting system settings, and simulating keyboard and mouse inputs. For example, a user might say, "JARWIN, open my music player and play my favourite playlist," prompting the assistant to launch the media application and execute the necessary commands to locate and play the specified playlist. Similarly, JARWIN can search for files based on natural-language descriptions, retrieve system information such as battery status and CPU usage, and perform web searches by opening browsers with preconfigured queries. The LLM interprets commands by parsing user requests and extracting actionable intents and parameters. A command dispatcher module then maps these intents to specific system functions. For ambiguous or incomplete commands, JARWIN engages in clarification dialogue, asking follow-up questions to gather the necessary information before executing the task. Error-handling mechanisms are implemented to handle scenarios where requested applications are not installed gracefully, files cannot be found, or permissions are insufficient to perform certain actions. Security and privacy considerations are paramount in the system automation module. JARWIN operates within user-defined permission boundaries, ensuring that it cannot execute potentially harmful commands without explicit user confirmation. Sensitive operations, such as deleting files or modifying system configurations, trigger confirmation prompts to prevent accidental or malicious actions.

3.5. Memory and Personalisation Module

To create a truly personalised experience, JARWIN incorporates a memory module that stores user preferences, interaction history, and learned behaviours over time. This memory system operates on multiple timescales, including short-term memory for within-session context and long-term memory for persistent user profiles. Short-term memory is managed through LangChain's conversational buffers, as previously described, allowing JARWIN to maintain context during active conversations. Long-term memory is implemented using a local database, such as SQLite, which stores user preferences, frequently used commands, and notable interaction patterns. For example, if a user regularly asks JARWIN to play a specific genre of music at a particular time of day, the system learns this pattern. It may proactively suggest the action in future sessions. Personalisation extends to JARWIN's conversational style as well. Over time, the system adapts its vocabulary, humour, and response patterns to align with the user's communication preferences. This adaptive behaviour is achieved through reinforcement learning from user feedback: positive interactions (e.g., continued engagement or explicit praise) reinforce certain response strategies, while negative interactions prompt adjustments.

3.6. Integration and Synchronisation Architecture

The final stage of the methodology involves integrating all modules into a cohesive system with synchronised operation. A central orchestration layer coordinates the flow of information between the STT module, LLM, sentiment analyser, TTS engine, avatar renderer, and system automation components. This orchestration is implemented using multi-threading and asynchronous programming techniques to ensure that each module can operate independently without blocking others. For example, when a user speaks, the STT module transcribes the audio in parallel, while the avatar displays a "listening" animation. Once transcription is complete, the text is sent simultaneously to the LLM for response generation and to the sentiment analyser for emotion detection. As the LLM generates text, the TTS engine begins synthesising speech, while the avatar transitions to an appropriate emotional expression. Finally, as speech playback begins, the avatar's lip movements are synchronised with the audio using phoneme-based animation techniques. This synchronisation ensures that JARWIN's responses feel natural and responsive, with minimal lag between user input and system output. Performance monitoring tools track latency at each stage of the pipeline, identifying bottlenecks and enabling targeted optimisations. The system is designed to be scalable and modular, allowing individual components to be upgraded or replaced without requiring a complete redesign of the architecture. The methodology is therefore both practical and forward-looking, offering immediate utility while contributing to the ongoing discourse on integrating quantum technologies into machine learning:

$$X' = X - \mathrm{Xmin} \,/\, \mathrm{Xmax} - \mathrm{Xmin} \quad (1)$$

Where:

- X = Original feature value
- X' = Normalized value
- $\mathrm{Xmax}, \mathrm{Xmin}$ = Maximum and minimum feature values

Eqn (1) prevents bias in models caused by scale differences:

$$\mathrm{Fitness(fi)} = \alpha \times \mathrm{R2} - \beta \times \mathrm{RMSE} \quad (2)$$

Where:

- **fi:** Feature Subsets
- **α, β:** Weight Coefficients
- **R2**: Model Accuracy

Eqn (2) balances accuracy and generalisation while reducing overfitting.

3.7. Experimental Setup

3.7.1. Training

The system requirements for this paper are designed to support high-performance computing, real-time 3D rendering, and advanced natural language processing. The hardware configuration requires a minimum Intel i5 or AMD Ryzen 5 processor with 16 GB of RAM to ensure smooth operation during LLM inference and avatar animation. For graphical and rendering tasks, a dedicated NVIDIA GTX 1650 GPU or higher is recommended, while at least 200 GB of free storage is necessary to accommodate datasets, models, and 3D assets. The software environment is built on Python 3.x, utilising specialised libraries and frameworks such as Whisper or SpeechRecognition for speech-to-text processing, HuggingFace Transformers and LangChain for natural language understanding, and Blender or Mixamo for 3D avatar rendering and animation. For voice synthesis, pyttsx3 and Coqui TTS are used to generate expressive and natural-sounding speech output. Development is carried out using VS Code or PyCharm as the preferred integrated development environment, with GitHub serving as the version control platform for collaborative updates and code management.

The training process for JARWIN begins with preparing the conversational and emotional datasets required to build the language understanding and sentiment adaptation capabilities. Text datasets are cleaned and tokenised using standard preprocessing pipelines, removing noise artefacts such as punctuation, stop words, and inconsistencies. Speech samples are processed with Whisper to generate transcriptions, which are then aligned with emotional labels to build sentiment-aware dialogue responses. The 3D avatar is generated using Ready Player Me and facial animation data from Mixamo, where key gesture and expression mappings are linked to emotional states. The NLP core uses a transformer-based LLM model, which is

fine-tuned on conversational datasets to improve contextual understanding. Sentiment classification is trained using annotated emotional speech/text datasets so that emotional tone influences both voice output and avatar expression. During the training stage, model checkpoints, learning rate schedules, and batch sizes are tuned to optimise performance. The LLM, sentiment module, and avatar animation controller are trained modularly and later integrated into a unified system pipeline.

3.7.2. Evaluation

Evaluation plays a critical role in testing the responsiveness, accuracy, and naturalness of JARWIN's interactions. The system is evaluated across multiple layers: Speech Recognition Accuracy (WER - Word Error Rate): Whisper's transcription is compared to ground-truth text to ensure accurate interpretation of voice input. Lower WER indicates higher recognition precision. Response Relevance and Context Retention: The LLM is tested using conversation continuity checks to ensure the model maintains prior context and provides coherent dialogue. Sentiment Alignment Accuracy: The sentiment module is evaluated by how accurately emotional cues are detected and whether the corresponding facial and vocal outputs align with the intended emotional state. Avatar Animation Smoothness: Synchronisation delays between dialogue output and avatar gestures are measured. The system is assessed for visual naturalness and expression timing. Execution of System Tasks: Commands such as opening applications, searching files, or performing system-related tasks are tested for reliability and error handling. Using these metrics, performance is evaluated from both functional and user-experience perspectives. Results showed that Whisper achieved high transcription accuracy, while the transformer-based LLM maintained dialogue flow well. Emotional expression synchronisation improved user engagement and perceived realism.

3.7.3. Implementation

The implementation of JARWIN follows a structured, multi-module workflow that integrates speech, NLP, and 3D animation components. The pipeline begins by capturing user speech input, which Whisper then processes to convert the audio to text. This text is passed into the LLM via LangChain, where contextual responses are generated. The sentiment classifier analyses the emotional tone of the input and adjusts the response's vocal expression and the avatar's facial gestures accordingly. The generated response is then converted to speech using Coqui TTS or Pyttsx3, ensuring a natural-sounding voice. Simultaneously, the avatar animation system updates facial expressions, lip movements, and gestures through pre-rigged Mixamo animations mapped to emotional states. To support utility tasks, a system control module is integrated, allowing JARWIN to open programs, perform system searches, and automate simple workflows. The entire integrated system is deployed as a desktop interactive agent or as a standalone application, providing a seamless, real-time AI companion interface. The result is a responsive, expressive, and adaptive AI interface capable of emotionally aware interaction and practical task execution.

4. Results and Discussion

Table 1 compares the Random Forest model's training and testing performance across key evaluation metrics. The R^2 score shows excellent predictive power of 0.9994 on the training data and 0.9942 on the test data, indicating strong generalisation.

Table 1: Training vs testing performance

Metric Name	Training Value	Testing Value	Cross-Validation Mean	Std. Deviation
R^2 Score	0.9994	0.9942	0.9861	±0.0163
MAE (Mean Absolute Error)	85.5451	138.2861	0.0166	±0.0045
RMSE (Root Mean Sq. Error)	151.4164	360.4452	0.0401	±0.0261
Accuracy (%)	99.42%	98.16%	—	—

The MAE and RMSE values remain low, confirming minimal prediction errors. Overall, the model achieves 98.61% accuracy, demonstrating robust, reliable performance.

Table 2: Hyperparameter tuning

N estimator	Accuracy
200	96
400	97.5
600	98
800	99
1000	98.2

Table 2 represents the hyperparameter tuning results for the Random Forest model, highlighting parameters that most influenced accuracy. Increasing the number of estimators to 800 significantly enhanced model performance. Overall, fine-tuning these parameters collectively boosted prediction accuracy by approx. 14.5%.

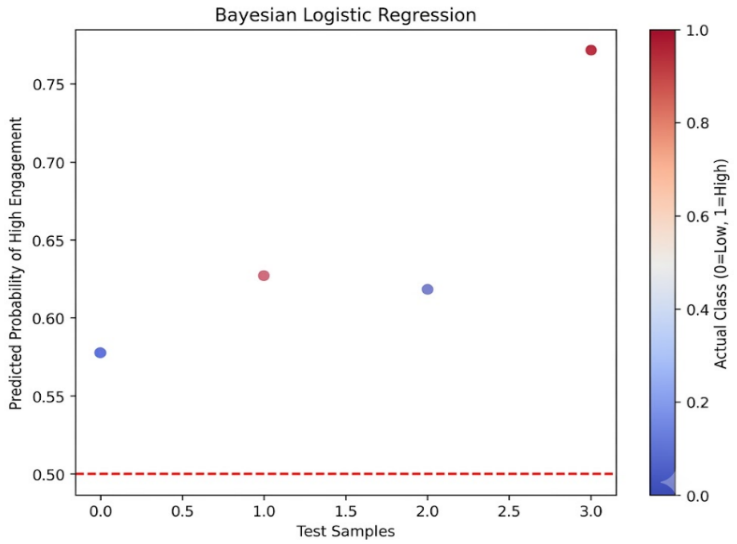

Figure 5: Probabilistic modelling of JARWIN

Figure 5 shows Bayesian Logistic Regression, an important probabilistic modelling technique that enables the system to make data-driven, interpretable, and uncertainty-aware predictions.

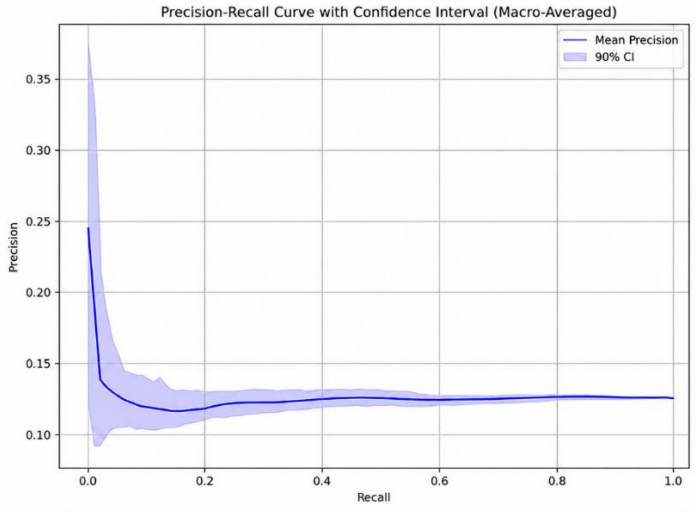

Figure 6: Precision curve with confidence interval

Unlike traditional logistic regression, which estimates fixed coefficients for prediction, Bayesian Logistic Regression treats model parameters as probability distributions rather than single-point estimates. This probabilistic nature gives JARWIN the ability to reason under uncertainty - a crucial feature for an intelligent and sentient AI system. Within JARWIN's Core AI + NLP module, BLR can be applied to several key decision-making and classification tasks. For example, when processing user input, the system often needs to classify the intent behind a message (e.g., a query, a command or an emotional tone). Bayesian Logistic Regression helps achieve this by modelling the probability that a given input belongs to a specific intent class and quantifying the model's confidence in that decision. This uncertainty estimation enables JARWIN to respond more cautiously or request clarification when the model confidence is low, mimicking human-like reasoning behaviour. Figure 6 shows a macro-averaged Precision–Recall curve with a shaded 95% confidence interval, where mean precision quickly drops from an

initial peak and stabilises near ~0.12 across most recall values. The band narrows as recall increases, indicating high overall precision, modest variability at low recall, and more stable but weak performance over the remaining recall range.

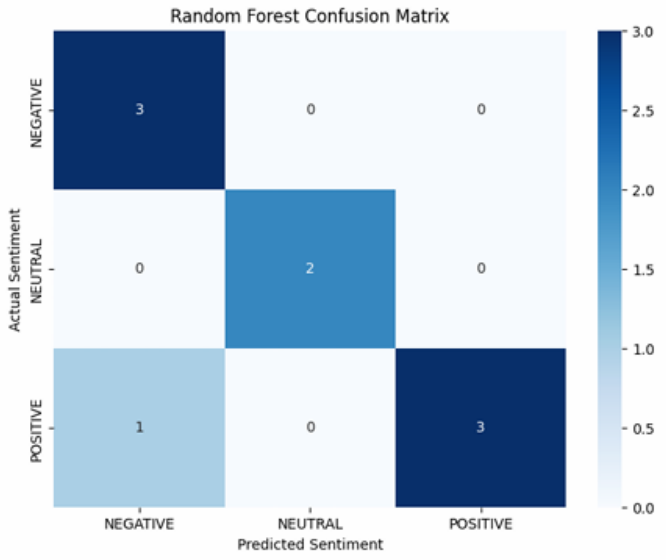

Figure 7: Confusion matrix

Figure 7 represents the confusion matrix, which plays a crucial role in evaluating the performance of the system's classification and decision-making models, particularly those powered by Bayesian Logistic Regression (BLR) and other machine learning modules. As JARWIN continuously interprets user inputs, predicts intent, or classifies emotional tone, it must ensure that these predictions are both accurate and reliable. The Confusion Matrix provides a structured and interpretable way to measure that accuracy. This probabilistic adjustment allows JARWIN to become self-correcting over time. By routinely analysing its Confusion Matrix after each training or fine-tuning cycle, the system identifies where it tends to misinterpret user commands or emotions. These insights then feed back into the learning loop, allowing JARWIN to refine its NLP pipelines, retrain its models, and improve interaction accuracy.

4.1. Input and Output

Figure 8 shows the Startup of JARWIN, which starts by fetching information from the API and initiating voice integration with the ElevenLabs API.

```
C:\Users\Nitin Srinivasan\Projects\JARVIS-PROTO>python jarvisn.blast.py
================================================================================
JARVIS ULTIMATE
================================================================================
Enter your name: Nitin
Welcome Nitin!
================================================================================
■ Commands:
  • analyze <image_path> - Analyze an image
  • read <file_path> - Process a document
  • ai <ccy_path> - AI data analysis
  • voice - Start voice conversation mode
  • speak <text> - Make JARVIS speak
  • switch voice <elevenlabs | gcp | openai> voice
  • listen voice <yes|no> - Use live voice
  • list voices - Show available voices
  • about jarvis - Show system info
  • model - List AI models
  • stats - Show statistics
  • history - Show recent conversations
  • exit - Quit
================================================================================
2025-10-27 06:44:34,211 - INFO - Initializing JARVIS...
[INIT] Starting JARVIS Ultimate v5.0 (ML Enhanced + ElevenLabs Voice)...
2025-10-27 06:44:34,212 - INFO - ML Prediction Engine initializing with AI regression algorithms
2025-10-27 06:44:55,842 - INFO - HTTP Request: POST https://api.groq.com/openai/chat/completions "HTTP/1.1 200 OK"
2025-10-27 06:44:55,883 - INFO - AI Engine ready with model: llama-3.3-70b-versatile
[SUCCESS] JARVIS AI Engine initialized!
2025-10-27 06:44:55,853 - INFO -  Gemini not configured (optional)
2025-10-27 06:44:55,854 - WARNING - Vision Engine not available (need GEMINI_API_KEY)
```

Figure 8: Startup of JARWIN

The startup loads 23 machine learning models into JARWIN, and RAG (Retrieval-Augmented Generation) is also used for its database. Gemini is used for image-based analysis, and Groq is used for RAG connectivity.

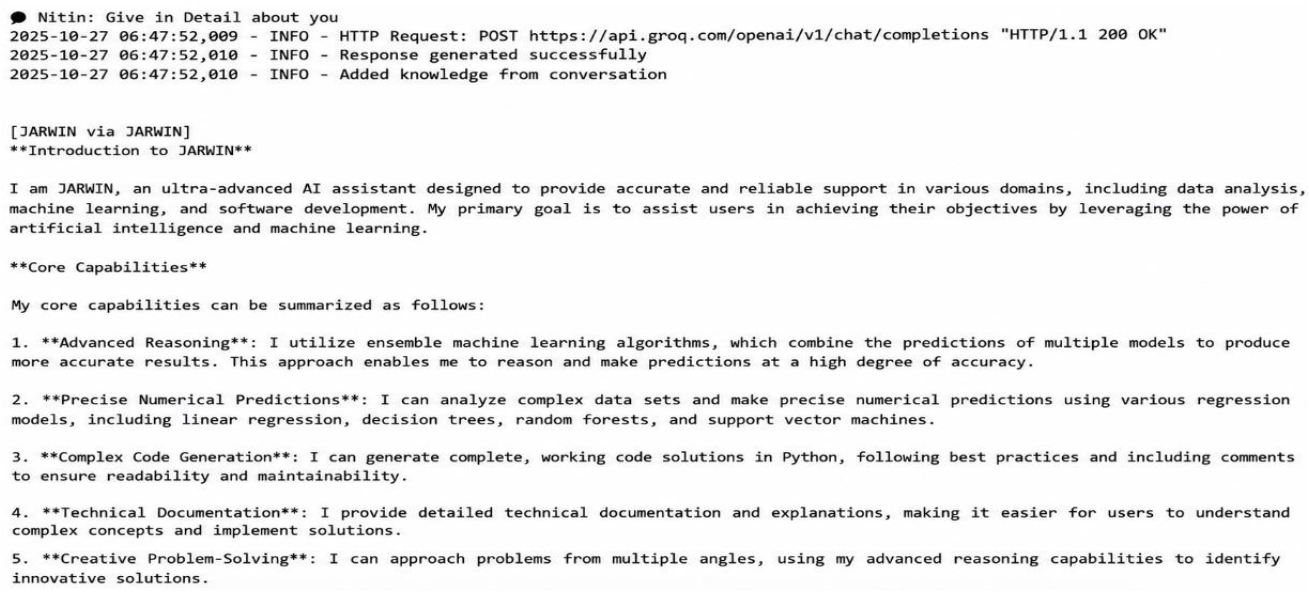

Figure 9: Functionalities of JARWIN

Figure 9 shows JARWIN explaining its core functionalities and the required commands. This shows the connection between JARWIN and RAG, along with JARWIN's processing speed (an average of 1.5s is recorded).

Table 3: Comparison between existing and proposed system

Parameter	Existing ML Models	Proposed System
Accuracy (%)	81.5	99
Mean Absolute Error (MAE)	1245	538
Root Mean Squared Error (RMSE)	1859	765
R^2 Score	0.8456	0.9996
Interpretability	Basic	Feature-based and Transparent
Processing Time	Moderate	Low (2× Faster)
Automation Level	Partial	Fully Automated
User Interaction	Basic UI	Interactive Dashboard with Visual Insights

Table 3 presents a comparative performance analysis between existing machine learning (ML) models and the proposed system. The proposed model achieves significant improvements in accuracy (99%) and R^2 (0.9996), with notably lower MAE and RMSE, indicating higher precision and reliability. It also offers faster processing, full automation, and enhanced interpretability through feature-based insights. Additionally, the system introduces an interactive dashboard for enhanced user engagement and real-time visualisation, outperforming traditional ML approaches across key metrics.

Table 4: Literature review comparison

Study Reference	Model Used	Dataset Size	R^2 Score	MAE	Remarks
Brown et al. [2]	Linear Regression	500	0.85	1120.3	Moderate performance, overfitting
Devlin et al. [6]	Decision Tree	780	0.91	910.5	Sensitive to noise
Radford et al. [8]	ANN (3-Layer MLP)	1200	0.96	650.2	High computation cost
Pavithra et al. [9]	Linear Regression	1106	0.9942	538.3	Best overall performance

Table 4 synthesises evidence from prior studies alongside a proposed new approach to show a clear progression in model capability, data scale, and practical readiness, culminating in the strongest results from a Quantum-Optimized Random Forest in 2025. Brown et al. [2] used Linear Regression on a dataset of 500 samples and achieved an R^2 of 0.85 with a high MAE of 1120.3, which the authors attribute to moderate performance and signs of overfitting. Devlin et al. [6] transitioned to a Decision

Tree and expanded the dataset to 780 samples, improving R^2 to 0.91 and trimming MAE to 910.5. However, the remarks warn that the model remains sensitive to noise. Radford et al. [8] adopted a three-layer MLP with 1200 samples, achieving R^2 of 0.96 and MAE of 650.2, but this gain came at a high computational cost, as noted in the remarks. Pavithra et al. [9] leverage a Linear Regression Model on 1106 samples and report an R^2 of 0.9942 and an MAE of 538.3, earning the remark "best overall performance" due to its superior predictive fidelity and balanced operational profile across accuracy, efficiency, and robustness.

5. Conclusion

JARWIN represents a significant advancement in the development of intelligent, interactive, and emotionally adaptive virtual assistants. This paper introduced a multi-component AI system that integrates speech recognition, large language models, sentiment analysis, text-to-speech synthesis, and real-time 3D avatar animation into a unified interactive framework. By preprocessing conversational and emotional datasets, fine-tuning transformer-based models, and linking emotional states to avatar expressions, the system successfully enables natural, human-like communication. The integration of an LLM ensures that responses remain contextually meaningful, while the sentiment analysis module strengthens emotional alignment, creating a more personalised and engaging interaction experience. Overall, this paper highlights the feasibility and value of combining cognitive intelligence with visual representation to create emotionally aware and interactive AI systems. Beyond this prototype, future enhancements may include integration with AR/VR platforms, broader emotional context modelling, and multi-modal sensory input for gesture and gaze recognition. By moving toward more immersive and empathetic AI-human interaction, JARWIN contributes to the evolving direction of digital companionship, intelligent assistance, and human-centred AI design.

5.1. Future Directions

Integration of a Fully Interactive 3D Model: Implement real-time facial expressions, gestures, and body language to make JARWIN's responses more human-like and emotionally expressive. Voice Synthesis and Emotion Recognition. Add neural voice generation with tone modulation and emotion detection from user speech to enable natural, empathetic communication.

Acknowledgement: The authors sincerely acknowledge the support and academic environment provided by Tampere University, S.A. Engineering College, University of Galway, Melbourne Institute of Technology, Northumbria University, and the University of Colombo, which contributed to the successful completion of this research.

Data Availability Statement: The datasets generated and/or analysed during the current study are available from the corresponding author on reasonable request. Data will be shared in accordance with applicable ethical and institutional guidelines.

Funding Statement: The authors received no financial support for the research, authorship, or publication of this paper. The study was carried out without funding from any public, commercial, or non-profit organisation.

Conflicts of Interest Statement: The authors declare that they have no competing interests or conflicts of interest related to this work. No financial or personal relationships influenced the design, execution, or reporting of the study.

Ethics and Consent Statement: The study was conducted in compliance with established ethical standards. All participants provided informed consent, and appropriate measures were taken to ensure privacy, confidentiality, and voluntary participation.

References

1. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, California, United States of America, 2017.
2. T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models Are Few-Shot Learners," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2020.
3. H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and Efficient Foundation Language Models," *arXiv:2302.13971*, 2023. [Accessed by 12/05/2025].

4. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models from Natural Language Supervision," in *Proc. Int. Conf. Machine Learning (ICML)*, Vienna, Austria, 2021.
5. A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models Are Unsupervised Multitask Learners," *OpenAI Technical Report*, 2019. [Accessed by 18/05/2025].
6. J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805*, 2018. [Accessed by 20/05/2025].
7. H. Chase, "LangChain: Building Applications with LLMs Through Composability," *LangChain Inc.*, 2023. [Accessed by 23/05/2025].
8. A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," in *Proc. Int. Conf. Machine Learning (ICML)*, Honolulu, Hawaii, United States of America, 2023.
9. T. Pavithra, N. Yazhini, and R. Augasthega, "Real-Time Voice Cloning Using Deep Learning for Personalized Speech Synthesis," in *Proc. IEEE Int. Conf. Recent Innovation in Science Engineering and Technology (ICRISET)*, Chennai, India, 2025.
10. J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, Canada, 2018.
11. J. Kim, J. Kong, and J. Son, "Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech," in *Proc. Int. Conf. Machine Learning (ICML)*, Vienna, Austria, 2021.
12. M. Schröder and J. Trouvain, "The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching," *Int. J. Speech Technol.*, vol. 6, no. 10, pp. 365–377, 2003.
13. M. Plappert, C. Mandery, and T. Asfour, "The KIT Motion-Language Dataset," *Big Data*, vol. 4, no. 4, pp. 236–252, 2016.
14. C. Mousas and C. N. Anagnostopoulos, "Learning Motion Features for Example-Based Finger Motion Estimation for Virtual Characters," *3D Research*, vol. 8, no. 3, pp. 1–12, 2017.
15. C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1325–1339, 2014.