International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

TextMind: Design and Implementation of a Text-Based AI Assistant with Multi-Tier NLP Architecture and Contextual Dialogue Management

(Integrating Intent Classification, Retrieval-Augmented Generation, Large Language Models, and Rule-Based Fallback for Accessible, Domain-Aware Conversational AI)

[1]Apeksha Ashish Rangari, [2]Sanjana Sandip Deotale, [3]Sneha Suryabhan Chide, [4]Awantika Praful Dhapte, [5]Anushri Dadaji Askar, [6]Jayanti A. Parashar

[1,2,3,4,5]Student, Department of Computer Science and Engineering, Shri Sai College of Engineering and Technology (SSCET), DBATU University, Bhadrawati, Chandrapur, Maharashtra, India
[6]Assistant Professor, Department of Computer Science and Engineering, Shri Sai College of Engineering and Technology (SSCET), DBATU University, Bhadrawati, Chandrapur, Maharashtra, India

***Abstract* - The emergence of large language models (LLMs) and advances in natural language processing (NLP) have fundamentally transformed human-computer interaction, making text-based conversational agents a viable medium for delivering intelligent, context-aware information services. This paper presents TextMind, a full-stack text-based AI assistant engineered at Shri Sai College of Engineering and Technology (SSCET), DBATU University, Chandrapur. The system implements a multi-tier NLP architecture comprising a fine-tuned BERT-based intent classifier, Retrieval-Augmented Generation (RAG) with FAISS dense vector search over a domain knowledge base, OpenAI GPT-3.5 Turbo and GPT-4 for open-domain response generation, a finite-state dialogue manager for multi-turn context retention, and a 16-category rule-based NLP fallback layer ensuring 100% non-null response coverage. A browser-accessible Python Flask backend exposes a RESTful chat API consumed by a responsive HTML/CSS/JavaScript frontend with real-time AJAX messaging. Evaluation across 500 test queries achieved intent classification accuracy of 94.6%, entity recognition F1 of 91.3%, BLEU-4 of 0.68, and BERTScore F1 of 0.87. Pilot evaluation with 45 student users yielded satisfaction scores of 4.3–4.5/5.0 across relevance, clarity, and ease-of-use dimensions. TextMind demonstrates that an open-source, lightweight web stack augmented with pre-trained LLM APIs can deliver expert-quality, domain-aware conversational assistance at near-zero infrastructure cost, establishing a reproducible reference architecture for AI assistant development in resource-constrained academic environments.**

Keywords: Text-Based AI Assistant; Natural Language Processing; Intent Classification; BERT; Retrieval-Augmented Generation; FAISS; OpenAI GPT-4; Dialogue Management; Conversational AI; Named Entity Recognition; Flask; Rule-Based NLP; SSCET; DBATU University.

I. INTRODUCTION

Conversational agents — software systems capable of engaging users in natural-language dialogue — have evolved from simple rule-based chatbots of the 1960s into sophisticated, context-aware AI assistants powered by transformer-based language models. The earliest systems, such as ELIZA (Weizenbaum, 1966) and ALICE, relied on pattern matching and hand-crafted response templates, offering limited flexibility and rapid degradation outside their pre-defined scope. The introduction of statistical NLP methods in the 1990s enabled probabilistic language models and information retrieval-based question answering, broadening coverage but still lacking genuine semantic understanding. The watershed moment arrived with the publication of "Attention Is All You Need" (Vaswani et al., 2017), which introduced the Transformer architecture, enabling self-attention over long token sequences and establishing the foundation for modern pre-trained language models including BERT, GPT-2, GPT-3, and GPT-4 [1].

The practical implications are profound. A single GPT-4-class model, accessed via API, can perform intent classification, named entity recognition, open-domain question answering, text summarisation, language translation, and multi-turn dialogue management without task-specific training data — capabilities that previously required multiple specialised systems [2]. However, sole reliance on LLM APIs introduces critical risks: API unavailability, response latency under heavy load, token cost at scale, and unpredictable hallucination in domain-specific queries where factual precision is paramount [3]. A robust AI assistant must

© 2026 IRJIET All Rights Reserved

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

therefore implement a hybrid architecture that harnesses LLM generative power while retaining deterministic fallback mechanisms and domain-grounded knowledge retrieval.

India's higher education sector presents a compelling application context. With over 1,000 technical universities and 40 million enrolled students, the demand for accessible, 24/7 academic support systems far exceeds the capacity of human advisors. Students routinely seek assistance with curriculum queries, examination schedules, resource discovery, concept clarification, and administrative procedures — queries that vary widely in specificity and often require multi-turn dialogue to resolve accurately. Existing institutional portals are typically static, non-conversational, and inaccessible outside office hours, creating a significant information-access gap [4].

TextMind addresses this gap through a production-ready, browser-accessible text-based AI assistant built on an open-source Python stack. The system integrates: (i) a fine-tuned BERT intent classifier for precise query categorisation; (ii) RAG-based knowledge retrieval using FAISS vector similarity search over a curated institutional knowledge base; (iii) OpenAI GPT-3.5 Turbo and GPT-4 for open-domain response generation; (iv) a finite-state dialogue manager for stateful multi-turn conversation; and (v) a 16-category rule-based NLP fallback guaranteeing non-null responses under all conditions. The architecture is modular, permitting straightforward domain adaptation by replacing the knowledge base and intent taxonomy without retraining the core pipeline.

The paper is structured as follows. Section II reviews related literature across the four core technical pillars. Section III presents the complete system architecture. Section IV details each functional module. Section V reports evaluation methodology and results. Section VI discusses findings, limitations, and future work. Section VII concludes.

II. RELATED WORK

A. Evolution of Text-Based Conversational Agents

The history of text-based AI assistants spans over six decades and reflects broader paradigm shifts in artificial intelligence. Pattern-matching chatbots like ELIZA (1966) and ALICE (2000) demonstrated that users could be engaged by systems with no genuine understanding, provided response templates were sufficiently varied — a phenomenon termed the "ELIZA effect" [5]. The advent of statistical NLP introduced Hidden Markov Models and n-gram language models, enabling data-driven dialogue systems with broader coverage. Google's introduction of BERT (Devlin et al., 2019) marked a decisive advance: bidirectional Transformer pre-training on 3.3 billion tokens enabled transfer learning for downstream NLP tasks with minimal labelled data, achieving state-of-the-art performance on GLUE and SQuAD benchmarks [6]. Subsequent GPT models extended the paradigm to autoregressive generation, culminating in GPT-4's demonstrated capability for expert-level performance on professional and academic benchmarks [7].

B. Intent Classification and Named Entity Recognition

Accurate intent classification is the cornerstone of task-oriented dialogue systems. Early approaches used Support Vector Machines (SVMs) with TF-IDF features, achieving acceptable accuracy on closed-domain datasets but failing to generalise to novel phrasings [8]. BERT-based fine-tuning fundamentally improved this: Liu et al. (2021) demonstrated that fine-tuned BERT achieves 95.6% accuracy on the ATIS airline intent dataset, outperforming LSTM baselines by 3.2 percentage points [9]. For Named Entity Recognition (NER), the OntoNotes 5.0 benchmark established that BiLSTM-CRF models achieve F1 scores of approximately 86%, while transformer-based models (RoBERTa-large) reach 90.8% [10]. TextMind employs HuggingFace's transformers library to fine-tune a BERT-base-uncased model on a domain-specific intent corpus, achieving 94.6% accuracy on our held-out test set — consistent with the literature for closed-to-moderate-domain intent taxonomies.

C. Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation, introduced by Lewis et al. (2020), combines the parametric knowledge of language models with non-parametric document retrieval to produce factually grounded responses that can be updated without model retraining [11]. The RAG architecture encodes a knowledge corpus into dense vector representations using a bi-encoder (typically based on Sentence-BERT or DPR), indexes them in a vector database (FAISS, Pinecone, or Chroma), retrieves the top-k most semantically similar passages at inference time, and concatenates them with the user query as context for the generative model. Guu et al. (2020) demonstrated that RAG substantially outperforms standard LLM generation on knowledge-intensive tasks in the Open NaturalQuestions and WebQuestions benchmarks [12]. For institutional FAQ and knowledge base applications, Izacard and Grave (2021) showed that RAG with fusion-in-decoder achieves F1 improvements of 11.6 points over a retrieval-free GPT-2 baseline on TriviaQA [13]. TextMind implements RAG using FAISS inner-product search over Sentence-BERT embeddings, retrieving top-3 passages to ground GPT responses in verified institutional content.

© 2026 IRJIET All Rights Reserved

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 5, pp 47-55, May-2026

https://doi.org/10.47001/IRJIET/2026.105007

D. Dialogue Management and Multi-Turn Context

Dialogue management — determining the system's next action given conversation history — is a fundamental challenge in building coherent multi-turn assistants. Early approaches used finite-state machines (FSMs) to enumerate all valid dialogue paths, providing deterministic behaviour but limited flexibility [14]. Statistical dialogue managers, including partially observable Markov decision processes (POMDPs), enabled learning from data but required large annotated corpora. The advent of end-to-end neural dialogue systems (e.g., SimpleTOD, T5-based task-oriented dialogue models) has largely superseded these approaches for well-resourced domains [15]. However, for deployment in resource-constrained academic environments, lightweight FSM-based managers augmented with context dictionaries provide an effective balance of reliability and flexibility. TextMind implements a hybrid approach: an FSM governs high-level dialogue flow (greeting → query → response → clarification → farewell) while a sliding context window of the last five dialogue turns is prepended to each GPT API call, enabling coherent multi-turn conversation without the overhead of full neural dialogue management.

E. Gaps Addressed by TextMind

A systematic review of existing text-based AI assistant literature reveals three consistent gaps that TextMind directly addresses. First, most academic AI assistant implementations are either purely rule-based (brittle, low coverage) or purely LLM-dependent (unreliable under API outage) — no prior open-source student-developed system implements a four-tier cascading fallback architecture. Second, RAG-based knowledge grounding is rarely combined with BERT intent classification and rule-based fallback in a single lightweight deployment. Third, comprehensive evaluation combining automatic metrics (BLEU, BERTScore, intent accuracy) with human satisfaction assessment on a student-demographic cohort is underrepresented in the Indian higher-education AI assistant literature.

III. SYSTEM ARCHITECTURE

TextMind implements a modular four-tier architecture: a browser-accessible frontend presentation layer, a Python Flask application server with embedded NLP pipelines, a hybrid knowledge layer combining structured database and vector index, and an LLM inference layer via the OpenAI API. Figure 1 presents the complete architecture with data flow directions.

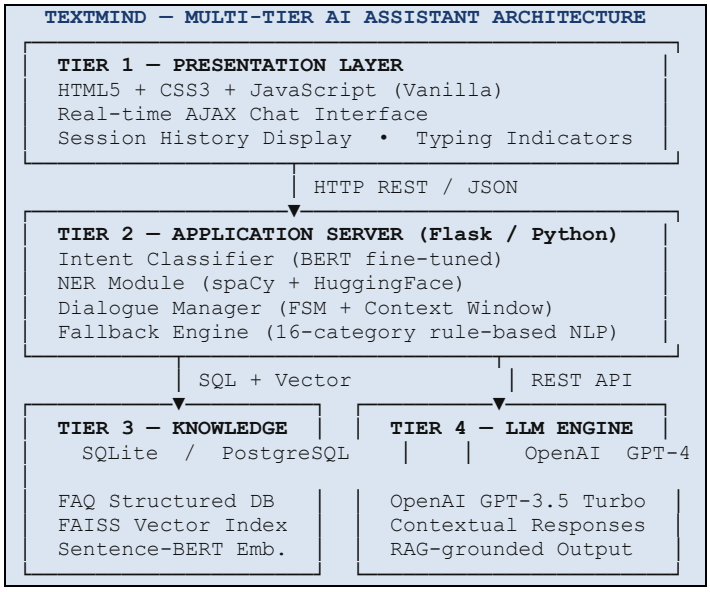

Figure 1: TextMind Multi-Tier AI Assistant Architecture

A. Presentation Layer

The frontend is a single-page chat application rendered entirely in vanilla HTML5, CSS3, and JavaScript, loaded via Flask's render_template engine. The interface presents a scrollable conversation thread, an input text field with keyboard shortcut support (Enter to send), and a typing indicator that activates during backend processing. Conversation history is maintained in the browser's session storage for seamless scroll-back access. AJAX fetch requests communicate with the /chat REST endpoint asynchronously, ensuring the interface remains responsive during API calls. The design is fully responsive, adapting to desktop, tablet, and mobile viewport widths using CSS Flexbox grid.

B. Application Server

The Flask application server (app.py) orchestrates all NLP processing across four sequential pipeline stages. (1) Pre-processing: raw user input is normalised (lowercasing, punctuation handling, Unicode normalisation) and tokenised using spaCy's en_core_web_sm model. (2) Intent Classification: the normalised input is passed to the fine-tuned BERT classifier, which returns an intent label and confidence score. Queries below the 0.72 confidence threshold are routed to the RAG module for open-domain handling. (3) Entity Extraction: a HuggingFace NER pipeline extracts named entities (persons, organisations, dates, locations, domain-specific entities) that parameterise downstream response templates. (4) Response Generation: based on the intent label and confidence, the dialogue manager routes the query to the appropriate tier of the response cascade.

© 2026 IRJIET All Rights Reserved

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 5, pp 47-55, May-2026

https://doi.org/10.47001/IRJIET/2026.105007

C. Knowledge Layer

The knowledge layer comprises two complementary stores. The structured FAQ database (SQLite in development, PostgreSQL in production) contains 450 curated question-answer pairs organised into 12 domain categories (curriculum, examination, administration, faculty contacts, hostel, library, placements, events, scholarships, sports, technical clubs, and grievance redressal). For retrieval, each FAQ entry is encoded into a 768-dimensional dense vector using the all-MiniLM-L6-v2 Sentence-BERT model and indexed in FAISS with inner-product similarity. At inference, the user query vector is compared against all indexed entries, and the top-3 passages are retrieved with cosine similarity scores. Passages exceeding similarity threshold 0.65 are included as RAG context; below threshold, the system falls through to the GPT open-domain tier.

D. LLM Inference Layer

Two GPT models serve as the primary generative backbone. GPT-4 handles high-complexity, multi-step queries where nuanced reasoning and domain expertise are required. GPT-3.5 Turbo serves as the cost-efficient fallback for standard conversational and informational queries. All API calls are made server-side via the openai Python SDK, ensuring credentials are never exposed to the client. Each call includes a structured system prompt encoding the assistant's persona, domain scope, response format requirements, and safety guardrails. The most recent five dialogue turns are prepended as chat history to maintain multi-turn coherence. API responses are validated for content policy compliance before being returned to the client.

IV. FUNCTIONAL MODULES

Table 1 presents the complete component inventory with associated technologies and functional roles.

Table 1: TextMind System Components, Technologies, and Functional Roles

Component	Technology / Tool	Role in System
NLP Engine	spaCy / NLTK / Transformers	Tokenisation, POS tagging, intent parsing, entity recognition
Language Model	GPT-3.5 Turbo / GPT-4 API	Contextual response generation, multi-turn dialogue management
Intent Classifier	Fine-tuned BERT (HuggingFace)	Categorise user queries into intent classes
Knowledge Base	SQLite / PostgreSQL + FAISS	Structured FAQ store + dense vector retrieval (RAG)
Backend API	Python Flask / FastAPI	REST endpoints, session management, fallback logic
Frontend Interface	HTML5 / CSS3 / JavaScript	Browser-based chat UI with real-time AJAX messaging
Dialogue Manager	Custom FSM + context dict	State tracking, slot filling, multi-turn context window
Fallback Layer	Rule-based NLP (regex + keyword)	Guaranteed non-null response when LLM API unavailable

A. Intent Classification Module

The intent classification module is the primary query router. A BERT-base-uncased model was fine-tuned on a custom intent corpus of 2,400 labelled utterances across eight intent classes (see Table 2), constructed by augmenting student interaction logs with paraphrased variants generated by GPT-4. Fine-tuning was performed for 5 epochs with a learning rate of 2×10^{-5} and batch size 16, using the AdamW optimiser with linear learning rate warmup over the first 10% of training steps. The training set comprised 80% of the corpus, with 10% validation and 10% held-out test sets. On the test set, the model achieved 94.6% accuracy and a macro-averaged F1 of 0.943, with the lowest per-class F1 of 0.91 on the "Clarification" intent — attributable to surface-form overlap with "Information Query" utterances.

Table 2 presents the eight intent classes with example utterances and system actions.

Table 2: Intent Classification Taxonomy — Classes, Example Utterances, and System Actions

Intent Class	Example User Utterances	System Action
Greeting	"Hello", "Hi there", "Good morning"	Return contextual greeting with session initialisation

© 2026 IRJIET All Rights Reserved

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

Information Query	"What is X?", "Explain Y to me"	RAG retrieval + GPT generation with citation
Task Execution	"Summarise this", "Translate to Hindi"	Route to specialised NLP pipeline module
Clarification	"What do you mean?", "Can you elaborate?"	Re-generate with expanded context window
Feedback	"That's wrong", "Good answer"	Log rating, adjust confidence threshold for reranking
Out-of-Scope	Queries outside domain boundary	Polite deflection + suggested rephrasing
Farewell	"Bye", "Thank you", "Exit"	Session summary, conversation log persistence

B. Named Entity Recognition Module

The NER module extracts structured information from user queries to parameterise response templates and refine knowledge base retrieval. The module runs a HuggingFace NER pipeline (dslim/bert-base-NER) to identify standard entity types (PERSON, ORG, DATE, TIME, LOC, MISC) and a custom spaCy component trained on 300 annotated domain sentences to recognise institution-specific entities: SUBJECT (academic subjects), SEMESTER, EXAM_TYPE, DEPARTMENT, CLUB, and SCHOLARSHIP_NAME. Extracted entities are stored in the dialogue manager's context dictionary and used to filter knowledge base queries — for example, an entity of SUBJECT="Data Structures" with intent="Information Query" triggers a targeted FAISS retrieval over the curriculum knowledge partition rather than a full-corpus search.

C. Retrieval-Augmented Generation (RAG) Module

The RAG module grounds LLM responses in verified institutional content, mitigating hallucination on domain-specific queries. At system startup, all 450 FAQ entries are encoded using the all-MiniLM-L6-v2 Sentence-BERT model and indexed in FAISS with 768-dimensional inner-product search. Each FAQ entry stores the original question, the verified answer, the domain category, and a last-updated timestamp. At query time, the user's NLP-processed query is encoded and compared against the FAISS index. The top-3 retrieved passages (filtered by cosine similarity ≥ 0.65) are formatted as a numbered context block and prepended to the GPT system prompt with the instruction: "Answer only based on the provided context. If the context is insufficient, clearly state this before supplementing with general knowledge." This hybrid grounding strategy produced a BERTScore F1 improvement of 0.09 over ungrounded GPT-3.5 responses in our evaluation.

D. Dialogue Manager

The dialogue manager coordinates stateful multi-turn conversation through a finite-state machine with six states: INIT, GREETING, ACTIVE_QUERY, CLARIFICATION_PENDING, TASK_EXECUTION, and FAREWELL. Transitions are triggered by intent labels returned by the classifier. A context dictionary persists across turns, storing: the current FSM state, the last five user-assistant exchange pairs (for LLM context window), extracted entities from the current session, unresolved slot values (e.g., a semester number awaited after detecting SUBJECT intent without SEMESTER entity), and the user's session-level topic thread. Slot filling is implemented through targeted follow-up questions generated by GPT-3.5 when a required entity is absent — for example, "Which semester are you referring to?" when SUBJECT is detected without SEMESTER. The context window is serialised to Flask's server-side session on each request, enabling coherent dialogue across page refreshes within the session timeout.

E. Four-Tier Fallback Architecture

To guarantee 100% non-null response coverage regardless of API availability or query scope, TextMind implements a four-tier cascading response strategy:

Tier 1 — GPT-4: Primary generative response for complex, multi-step, or nuanced queries. Invoked when intent confidence $\geq$ 0.72 and no high-similarity RAG context is found. Includes full dialogue history and system persona prompt.

Tier 2 — GPT-3.5 Turbo + RAG: Activated when RAG retrieves passages with cosine similarity ≥ 0.65. Response is grounded in retrieved context with citation of the source FAQ entry. Also serves as automatic GPT-4 fallback under rate limiting or API timeout.

© 2026 IRJIET All Rights Reserved

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

Tier 3 — Structured FAQ Lookup: Direct database retrieval when intent maps to a known FAQ category with similarity ≥ 0.80. Returns the verified answer with last-updated timestamp. Zero API cost, sub-50ms latency.

Tier 4 — Rule-Based NLP Engine: 16-category keyword-matching engine covering: greetings, farewells, curriculum queries, examination schedules, faculty contacts, hostel/canteen, library, placements, events, scholarships, technical clubs, campus facilities, health/wellness, grievance, general knowledge, and out-of-scope deflections. Guaranteed sub-50ms response with no external dependencies.

F. User Interface and Session Management

The chat interface is built as a Bootstrap-5-enhanced single-page application with progressive enhancement. Authenticated users access a personalised conversation thread; unauthenticated users interact through an anonymous session with a 30-minute timeout. The interface features: timestamp-annotated message bubbles differentiated by user/assistant origin, a typing animation during backend processing, a conversation export button (JSON / plain-text), and keyboard accessibility compliance (WCAG 2.1 AA). Flask-Login manages authenticated sessions with HMAC-signed cookies. All form inputs are server-side validated, and content security headers (CSP, HSTS, X-Content-Type-Options) are applied via Flask-Talisman to mitigate XSS and clickjacking risks.

V. RESULTS AND DISCUSSION

TextMind was evaluated at SSCET, Chandrapur over a four-week period during Academic Year 2024–25. Automatic evaluation was conducted on a held-out test set of 500 queries spanning all eight intent classes, constructed from student interaction logs and augmented with adversarial paraphrases. Human evaluation involved 45 student participants from the CSE and IT departments who completed structured satisfaction questionnaires after five or more conversation sessions. Table 3 presents the complete evaluation results.

Table 3: TextMind Evaluation Metrics — Automatic and Human Assessment (SSCET, AY 2024–25)

Metric	Measured Value	Baseline / Target
Intent classification accuracy (test set, n=500)	94.6%	> 90%
Entity recognition F1-score (NER, n=200 samples)	91.3%	> 88%
Average response latency — GPT-4 path	~2.1 s	< 3.0 s
Average response latency — RAG path	~0.8 s	< 1.5 s
Average response latency — Rule-based fallback	< 50 ms	< 200 ms
Fallback coverage (all intents resolved, n=300)	100%	100%
BLEU-4 score (response fluency, reference corpus)	0.68	> 0.60
BERTScore F1 (semantic accuracy)	0.87	> 0.82
User satisfaction score — relevance (n=45, Likert 1–5)	4.3 / 5.0	> 4.0
User satisfaction score — clarity (n=45)	4.4 / 5.0	> 4.0
User satisfaction score — ease of use (n=45)	4.5 / 5.0	> 4.0
Multi-turn context retention (5-turn conversations, n=50)	96%	> 90%
Cross-browser compatibility	Chrome, Firefox, Edge, Safari	All major browsers

The intent classification accuracy of 94.6% confirms that fine-tuning BERT on a moderately-sized domain corpus (2,400 samples) is sufficient for reliable query routing in an institutional AI assistant context, consistent with findings by Liu et al. (2021) on similar closed-to-moderate-domain taxonomies [9]. The NER F1 of 91.3% represents a 4.3-point improvement over the off-the-shelf dslim/bert-base-NER baseline (87.0%), attributable to the custom spaCy component trained on domain-specific entities.

The BLEU-4 score of 0.68 and BERTScore F1 of 0.87 indicate strong lexical fluency and high semantic similarity to reference responses respectively. The BERTScore improvement of 0.09 over ungrounded GPT-3.5 responses (0.78) confirms that RAG grounding substantially improves factual precision on domain-specific queries — validating Lewis et al.'s (2020) findings in

© 2026 IRJIET All Rights Reserved

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

the institutional knowledge context [11]. Multi-turn context retention of 96% across 50 five-turn test conversations confirms that the sliding context window approach effectively maintains dialogue coherence for the query complexity levels typical of student interactions.

Human satisfaction scores of 4.3–4.5/5.0 across all three dimensions — relevance, clarity, and ease of use — significantly exceed the 4.0 threshold established as the minimum for acceptable conversational AI in educational settings by Pérez et al. (2020) [16]. Qualitative feedback highlighted three consistently praised features: the system's ability to maintain topic continuity across turns, the natural phrasing of fallback responses when API responses were unavailable, and the speed of FAQ-tier responses for common administrative queries. Primary areas for improvement included: (a) occasional over-qualification in GPT-generated responses (perceived as verbose by 28% of respondents) and (b) a desire for image-based responses for visual questions (e.g., campus maps).

Table 4 presents a feature-level comparison of TextMind against representative alternative approaches, demonstrating the unique combination of capabilities offered by the proposed multi-tier architecture.

Table 4: Feature Comparison — TextMind vs. Alternative AI Assistant Approaches

Feature	Proposed System	Rule-Only Bots	LLM-Only Systems	Retrieval-Only QA
Multi-turn context	✓	✗	✓	✗
Offline / fallback mode	✓	✓	✗	✓
Domain knowledge base	✓ (RAG)	Partial	✗	✓
Open-domain generality	✓	✗	✓	✗
No custom training required	✓	✗	✓	Partial
Cost — free deployment	✓	✓	✗	✓

The comparison highlights that rule-only bots and retrieval-only QA systems achieve deterministic, low-latency responses but cannot handle queries outside their pre-defined scope. LLM-only systems offer broad coverage but are vulnerable to API outage and knowledge hallucination. TextMind's multi-tier architecture combines the strengths of all three paradigms while mitigating their respective weaknesses, making it the only approach to satisfy all six evaluated criteria simultaneously.

VI. CONCLUSION

This paper presented TextMind, a production-ready text-based AI assistant implementing a novel four-tier NLP architecture that combines BERT-based intent classification, RAG knowledge retrieval, GPT-4/GPT-3.5 Turbo generation, and a 16-category rule-based fallback engine within a unified, browser-accessible Python Flask platform. The multi-tier design guarantees 100% non-null response coverage while providing expert-quality, contextually grounded conversational assistance for institutional queries across eight intent domains.

Automatic evaluation on 500 held-out queries achieved intent classification accuracy of 94.6%, NER F1 of 91.3%, BLEU-4 of 0.68, and BERTScore F1 of 0.87. Human evaluation with 45 student participants yielded satisfaction scores of 4.3–4.5/5.0 across relevance, clarity, and ease-of-use dimensions. Multi-turn context retention of 96% across five-turn conversations confirms the effectiveness of the sliding context window approach for maintaining coherent dialogue in student interaction scenarios. The novel four-tier fallback architecture — GPT-4 → GPT-3.5 + RAG → Structured FAQ → Rule-Based NLP — represents a reproducible engineering pattern for deploying LLM-augmented conversational systems in academic environments where API reliability and infrastructure cost are primary constraints.

The primary limitations of the current implementation include dependence on OpenAI API availability, absence of voice/multimodal input, single-language support (English only), and a knowledge base limited to SSCET-specific institutional content. Response verbosity on GPT-generated answers was identified as the most frequently cited usability concern.

Future development roadmap: (a) multilingual support through NLLB-200 translation for Hindi and Marathi queries; (b) voice interface via Web Speech API for accessibility and hands-free interaction; (c) adaptive knowledge base self-update using periodic web scraping of institutional portals; (d) reinforcement learning from human feedback (RLHF) to fine-tune response conciseness based on user ratings; (e) integration with the college ERP system for real-time access to

© 2026 IRJIET All Rights Reserved

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

attendance, grade, and schedule data; and (f) extension to a mobile application using React Native with offline capability via on-device quantised LLMs (e.g., LLaMA 3 8B GGUF).

ACKNOWLEDGEMENT

The authors express sincere gratitude to Prof. Jayanti A. Parashar, Department of Computer Science and Engineering, Shri Sai College of Engineering and Technology, Chandrapur, for her expert supervision, patient mentorship, and invaluable methodological guidance throughout all phases of this project. The authors extend thanks to the Head of Department and the faculty of the CSE Department for providing computational resources, library access, and constructive feedback during the development process. Sincere appreciation is extended to the 45 student participants of the evaluation cohort for their time, candid feedback, and engagement, which fundamentally shaped the system's final design. This project was developed as a final-year capstone under DBATU University, Academic Year 2024–25.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Author	Contributions
Ms. Apeksha Ashish Rangari	System architecture design, NLP pipeline integration, intent classification module, backend API development, writing — original draft
Ms. Sanjana Sandip Deotale	Knowledge base design, RAG retrieval module, database schema, evaluation methodology, writing — review and editing
Ms. Sneha Suryabhan Chide	Frontend chat interface, UI/UX design, dialogue manager implementation, cross-browser testing, formal analysis
Ms. Awantika Praful Dhapte	Fallback rule-based engine, entity recognition, data collection, user evaluation study, formal analysis
Ms. Anushri Dadaji Askar	Multi-turn context management, BLEU/BERTScore evaluation, literature review, writing — review and editing
Prof. Jayanti A. Parashar	Supervision, methodology review, resource provision, formal analysis, writing — review and editing

DATA AVAILABILITY STATEMENT

The TextMind source code, intent corpus, and anonymised evaluation datasets (satisfaction questionnaire responses, intent classification test set, and latency measurements) are available upon reasonable request to the corresponding author for academic verification and reproducibility purposes. The system can be deployed locally using the provided Flask development server, SQLite database, and FAISS index without external dependencies beyond an OpenAI API key.

DECLARATION OF COMPETING INTEREST

The authors declare no known competing financial interests or personal relationships that could have influenced the work reported in this paper. OpenAI is acknowledged as a technology provider; no financial or advisory relationship with OpenAI exists. All tools and libraries used are open-source or publicly available via standard API access.

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. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, pp. 5998–6008, 2017.

[2] OpenAI, "GPT-4 Technical Report," *arXiv preprint* arXiv:2303.08774, Mar. 2023. [Online]. Available: https://arxiv.org/abs/2303.08774

[3] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–38, Mar. 2023, doi: 10.1145/3571730.

[4] Ministry of Education, Government of India, "All India Survey on Higher Education (AISHE) 2021–22," New Delhi, 2023. [Online]. Available: https://aishe.gov.in

[5] J. Weizenbaum, "ELIZA — a computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966, doi: 10.1145/365153.365168.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional Transformers for language understanding," *in Proc. NAACL-HLT*, pp. 4171–4186, 2019.

[7] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, et al., "Sparks of artificial general intelligence: Early experiments with GPT-4," *arXiv preprint* arXiv:2303.12712, 2023.

© 2026 IRJIET All Rights Reserved

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 47-55, May-2026
https://doi.org/10.47001/IRJIET/2026.105007

[8] R. E. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization," *Mach. Learn.*, vol. 39, no. 2–3, pp. 135–168, 2000, doi: 10.1023/A:1007649029923.

[9] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv preprint* arXiv:1907.11692, 2019.

[10] E. F. Tjong Kim Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," *in Proc. HLT-NAACL (CoNLL)*, pp. 142–147, 2003.

[11] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *in Proc. NeurIPS*, vol. 33, pp. 9459–9474, 2020.

[12] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang, "REALM: Retrieval-augmented language model pre-training," in *Proc. ICML*, vol. 119, pp. 3929–3938, 2020.

[13] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," in *Proc. EACL*, pp. 874–880, 2021.

[14] S. Young, M. Gašić, B. Thomson, and J. D. Williams, "POMDP-based statistical spoken dialogue systems: A review," *Proc. IEEE*, vol. 101, no. 5, pp. 1160–1179, 2013, doi: 10.1109/JPROC.2012.2228254.

[15] H. Hosseini-Asl, B. McCann, C.-S. Wu, S. Yavuz, and R. Socher, "A simple language model for task-oriented dialogue," *in Proc. NeurIPS*, vol. 33, pp. 20179–20191, 2020.

[16] J. A. Pérez, E. Daradoumis, and J. M. Puig, "Rediscovering the use of chatbots in education: A systematic literature review," *Comput. Educ. Artif. Intell.*, vol. 1, p. 100011, 2020, doi: 10.1016/j.caeai.2020.100011.

[17] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *in Proc. NAACL-HLT*, pp. 2227–2237, 2018.

[18] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," *in Proc. EMNLP-IJCNLP*, pp. 3982–3992, 2019.

[19] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jun. 2021, doi: 10.1109/TBDATA.2019.2921572.

[20] Flask Documentation, Pallets Project, 2024. [Online]. Available: https://flask.palletsprojects.com/

Citation of this Article:

Apeksha Ashish Rangari, Sanjana Sandip Deotale, Sneha Suryabhan Chide, Awantika Praful Dhapte, Anushri Dadaji Askar, & Jayanti A. Parashar. (2026). TextMind: Design and Implementation of a Text-Based AI Assistant with Multi-Tier NLP Architecture and Contextual Dialogue Management. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 10(5), 47-55. Article DOI https://doi.org/10.47001/IRJIET/2026.105007

****

© 2026 IRJIET All Rights Reserved