© JAN 2026 | IRE Journals | Volume 9 Issue 7 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I7-1713229

SmartMeet: An AI-powered Meeting Scheduler with Email Automation

SHIWAM MADDHESIYA[1], ROHAN PRASAD[2], ALI ABDULLAH[3], RAHUL YADAV[4], HARVENDRA KUMAR PATEL[5]
[1, 2, 3, 4, 5]*KCC Institute of Technology and Management Greater Noida, India*

Abstract*—Modern organizational workflows have been progressively dependent on sophisticated approaches to meeting coordination between people. However, traditional scheduling methods are still annoyingly inefficient, require heavy manual work, and are vulnerable to human error. We present SmartMeet, an intelligent automation framework that manages the entire meeting life cycle via natural conversational interfaces. Our multiple AI capabilities automatic speech recognition, transformer-based language understanding, large language model reasoning, and robotic process automation allow the system to execute the account action from a voice scheduling request without any human intervention.Different from currently available semi-automated tools which still require structured input or human verification at certain points, SmartMeet enables freeform speech, gets scheduling details by semantic parsing, resolves time conflicts algorithmically, makes calendar entries across systems, creates meeting links, and personalized notification emails, etc. all by itself. Through this research, we demonstrate that speech-to-intent pipelines are capable of handling real-world enterprise scheduling scenarios, invent new ways for detecting conflicts and suggesting alternatives, and present that cognitive AI components can be seamlessly integrated with deterministic RPA execution layers. Our design concepts have functionalities that complement the gaps of meeting automation research. It is a fully autonomous and natural language based solution that, in a measurable manner, assists in the reduction of the delayed scheduling cases, the elimination of the booking conflicts occurrences, and the improvement of the organizational productivity.

Keywords: Automated Meeting Coordination, Natural Language Understanding, Large Language Models, Calendar Integration, Email Automation, Conflict Resolution Algorithms

I. INTRODUCTION

A. Background and Motivation

Large modern professional ecosystems, such as corporate enterprises, academic institutions, and healthcare organizations, are showing an increased dependence on collaborative activities that are well-coordinated. The proliferation of geographically dispersed teams, remote working arrangements, and globally integrated operations has made the ability to coordinate meetings evolve from a simple administrative task into a mission-critical organizational capability. The fundamental process of coordinating meetings is still largely manual, cognitively demanding, and prone to errors in spite of the widespread use of digital calendar infrastructure such as Google Workspace, Microsoft 365, and collaboration platforms like Zoom and Microsoft Teams.

Contemporary workflow for scheduling basically means executing several steps in a row by humans. These steps include understanding scheduling requests given in natural language, manually checking calendar systems to find free time slots, resolving conflicts in the schedules of several participants, writing meeting invitations with the necessary details, and finally sending confirmation by electronic mail or messaging systems. This is a pattern that brings a number of inefficiency types that, among other things, harm the performance of an organization.

First, there is a huge opportunity cost associated with the repetitive nature of scheduling activities. Studies in organizational psychology indicate that knowledge workers spend around 3–5 hours per week on meeting coordination activities, which is time better spent on high-value analytical or creative work. Second, manual scheduling processes are prone to high error rates, taking the form of double-booked calendar slots,timezone conversion errors, forgotten participant invitations, or incorrect meeting duration specifications. According to a study published by Rescue Time Analytics, calendar conflicts are expected to affect around 18% of scheduled meetings in typical enterprise settings. Third, reliance on human intermediaries injects latency into organizational information flows, which is particularly problematic in time-critical

settings such as incident response, customer engagement, or executive decision-making scenarios.

For companies in velocity-dependent verticals such as technology startups, financial services, emergency healthcare, and competitive research environments, these scheduling inefficiencies result directly in poorer collaboration quality, longer project timelines, missed business opportunities, and degraded competitive positioning. Hence, there is tremendous demand for intelligent automation solutions capable of removing human intermediaries from the scheduling workflow while matching or outperforming the accuracy and contextual awareness of manual processes[5].

B. Technological Enablers

Recent convergence of multiple artificial intelligence research trajectories has created unprecedented opportunities to solve the challenges associated with meeting coordination. Natural Language Processing (NLP) has evolved from rule-based parsing systems to transformer-based architectures capable of nuanced semantic understanding, contextual reasoning, and entity extraction from unstructured text. Current NLP technologies are capable of interpreting temporal expressions, resolving pronominal references, and disambiguating the intent of a schedule even when the spoken language is informal or grammatically incorrect.

Robotic Process Automation (RPA) has progressed from simple macro-recording utilities into sophisticated orchestration platforms that can interact with enterprise applications through multiple integration modalities, such as API calls, user interface automation, database operations, and message queue interactions. With advanced RPA systems, it is possible to execute complex multi-step workflows in a deterministic manner with provisions for error handling, retry logic, and audit trail generation.

Examples of large language models (LLMs) are GPT-4, Claude, and domain-specialized variants, which have shown impressive performances in zero-shot and few-shot learning for a multitude of tasks. In fact, without a large task-specific training dataset, these models are able to carry out intent classification, entity recognition, data extraction from the given text in a particular format, and generate natural language text. In fact, their proficiency to produce structured results like JSON makes them highly instrumental in transforming the most recent trend of unstructured human communication into structured enterprise systems. [6].

Automatic Speech Recognition (ASR) systems, powered by deep learning architectures such as Wav2Vec 2.0 and Whisper, now achieve near-human accuracy for continuous speech transcription across diverse acoustic environments, speaking styles, and linguistic variations. This breakthrough makes speech a practical and efficient primary input modality for enterprise applications[7].

C. Research Contribution

This article details the conception, development, and assessment of *SmartMeet*, a fully automated meeting scheduling platform that combines various technological features through a single conversational agent. The design of the system is geared towards the use of spoken natural language as the primary mode of communication. The input speech is handled by advanced speech recognition technology; then, a large language model is used for reasoning to extract structured scheduling parameters. For the time frames provided, conflict detection algorithms interface with the calendar to verify temporal constraints; calendar operations are executed through an API, a virtual meeting link is created, and simultaneously, customized emails are dispatched to notify participants of the meeting. All procedures are carried out without human supervision or intervention.

The uniqueness of this concept lies in several distinguishing aspects. *SmartMeet* is built around speech-first interaction as its central feature, aligning closely with natural human communication patterns, unlike many virtual assistant technologies that rely on structured text inputs or operate in human-in-the-loop configurations. The system implements comprehensive conflict resolution logic that not only checks binary availability but also incorporates optimization heuristics for suggesting alternative time slots. The framework demonstrates the feasible integration of probabilistic AI components (speech recognition, NLP) with deterministic execution layers (RPA, calendar APIs) within a production-grade workflow.

The key contributions of this work span three interrelated areas of research[14]:

· Speech-to-Intent Transformation:

Demonstrating that modern speech recognition and language understanding technologies are accurate enough to be deployed without human verification loops in enterprise-critical scheduling operations.

- Intelligent Conflict Management: Introducing conflict detection methods that synergistically combine calendar API querying, temporal reasoning, and optimization heuristics to not only identify scheduling conflicts but also actively propose resolution strategies.
- AI–RPA Synthesis: Exploring architectural patterns for seamless interfacing between cognitive AI capabilities (reasoning, understanding, generation) and deterministic RPA execution layers (API calls, email dispatch, logging) in hybrid automation pipelines.

D. Application Domains and Impact

These kinds of automation can, in fact, be utilized for a wide range of scenarios within different organizations. For instance, executives and project managers in a business environment may find it handy to use voice assistants for arranging stakeholder meetings in a period when they are doing other activities or are on a trip by train, thus saving the time which would otherwise be spent in manually operating the device. Through conversational interfaces, educational institutions may equip their teaching staff with the faculty members to fix office hours, dissertation committee meetings, or conduct academic seminars, thus leading to a significant reduction in the administrative workload. Health care facilities may decide to equip their clinical staff with the facility to arrange patient appointments or care coordination meetings cross the different disciplines by voice input, thus decreasing the time which is taken out of the direct patient care activities. The system additionally supports accessibility requirements by offering an alternative interaction model for users with visual impairments, motor disabilities, or other conditions that make traditional graphical user interfaces less usable. By processing speech as the primary command modality, *SmartMeet* lowers technology adoption barriers and aligns with inclusive design principles[12].

E. Technical Challenges

This intricate design, at the very least, has to deal with the technical issues that are spread over several different categories. One of the examples is that speech recognition modules should be able to keep their performance level not only in positive but also in negative acoustic conditions, which may consist of background noise, reverberation, or changes in microphone quality. The features of the speakers - for instance, accents, speed of speaking, and pronunciation patterns - should definitely not be the factors that decrease the efficiency of the system in recognizing speech. Additionally, these systems are required to accept linguistic concepts like disfluencies, false starts, and conversational repairs. Components of natural language understanding have to be very precise in recognizing temporalphrases, changing informal references (e.g., “next Tuesday afternoon”) into the correct ISO-8601 timestamps and at the same time taking into account the timezone, calendar, and possible ambiguities.

The local system must also be robust enough to support diverse API specifications, authentication protocols, rate-limiting constraints, timezone normalization, and recurring event patterns. Conflict detection algorithms must be efficient in processing large sets of calendars while simultaneously considering multi-participant availability, meeting priorities, and organizational scheduling policies. Security and privacy concerns are critically important, given the sensitive nature of calendar data. Therefore, beyond implementing encryption, access control, and audit logging, the system must adhere strictly to applicable data protection regulations[11].

F. Paper Organization

The rest of the paper is structured as follows. Section II presents the Literature Review, summarizing prior research in the areas of human–AI scheduling systems, dialogue-based learning methodologies, natural language interfaces for meeting coordination, reinforcement learning techniques, multi-agent scheduling frameworks, and schema-driven dialogue models. This section identifies existing gaps and establishes the motivation for a speech-first, fully automated scheduling solution.

Section III describes the Methodology used by the authors. It explains the strategies implemented for speech recognition, natural language understanding, entity extraction and validation, temporal conflict detection, and the development of the end-to-end scheduling pipeline.

© JAN 2026 | IRE Journals | Volume 9 Issue 7 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I7-1713229

Section IV presents the Results and Discussion, providing a performance evaluation of the system in terms of recognition accuracy, entity extraction reliability, conflict detection precision, end-to-end scheduling success rate, latency, and overall user experience.

Section V details the Proposed Workflow of *SmartMeet*, describing the sequential steps from speech acquisition to calendar event creation and automated email notifications. This section outlines the operational pipeline responsible for achieving full scheduling automation.

Finally, Section VI provides the Conclusion, summarizing key contributions, major findings, system limitations, and potential future enhancements aimed at improving accuracy, deepening contextual understanding, and enabling multiplatform integration.

F. Problem Statement

Organizational environments' contemporary meeting scheduling workflows are plagued by systemic inefficiencies caused by manual processes and fragmented tooling. Fundamentally, the problem consists of several interconnected challenges:

- P1: Unstructured Input Processing — Scheduling requests are presented in various unstandardized forms.
- P2: Temporal Ambiguity Resolution — Natural language temporal expressions are inherently ambiguous and require contextual disambiguation.
- P3: Multi-Participant Availability Checking — Identifying mutual availability requires querying several calendar systems that may be distributed across different machines or locations.
- P4: Conflict Detection and Resolution — Beyond checking overlapping intervals, temporal conflict recognition requires advanced reasoning.
- P5: Execution Fragmentation — Scheduling a meeting generally involves multiple independent steps across different platforms that must be coordinated.
- P6: Accessibility and Interaction Overhead — Standard calendar applications favor visually oriented users who can operate devices manually.
- P7: Lack of Intelligent Automation — Current tools lack the intelligence needed to manage complex constraints or adapt to user preferences.

Formal Problem Definition: Given an unstructured spoken natural language scheduling request R, design an intelligent automatic system Σ that conceptualizes and executes the following transformation:

$$\Sigma : R \longrightarrow \{C, L, N, A\}$$

where the elements in the set denote:

- C: Calendar entries created.
- L: Real or virtual meeting endpoints generated.
- N: Notification emails dispatched.
- A: Audit logs produced.

Under the restrictions:

- No human intervention is allowed at any stage of the pipeline.
- The system must detect and resolve temporal conflicts.
- User privacy and calendar confidentiality must be strictly preserved.
- Ambiguous or underspecified requests must trigger clarification.
- The system must incorporate graceful error handling and rollback capabilities.

The system must be dependable, scalable, and at least as accurate as manual scheduling procedures while reducing scheduling latency and eliminating human effort.

Fig. 1. Context-Aware Multi-Agent Scheduling

II. LITERATURE REVIEW

The seminal work of Cranshaw et al. on *Calendar.help* provided significant insights into semi-automated scheduling systems. Their research proposed a three-layer architectural model in which scheduling tasks were divided into automated micro-tasks, human-assisted micro-tasks, and expert-level macro-tasks requiring domain-specific knowledge.

Analysis of real-world deployment data revealed that nearly 40% of scheduling requests could be handled fully by automation, whereas the remaining cases required human intervention to address ambiguity, complex constraints, or unforeseen situations[1].

Dialogue learning with human feedback was the focus of Plummer et al.'s research, which introduced fast training methods for conversational agents through iterative improvement cycles. Their approach integrated real-time human corrections directly into model training, enabling conversational agents to learn from interaction errors without large pre-labeled datasets. When the agent produced undesirable responses, humans provided detailed corrective feedback on specific parts of the output, allowing the model to adapt and avoid similar mistakes in the future. This framework enabled a practical transition from demonstration-heavy systems requiring frequent corrections to more stable production systems capable of learning from user interactions—an important advancement for handling diverse natural language expressions and ambiguous temporal references in scheduling tasks.

The work of Busemann and Declerck on the COSMA system[2] was among the first to develop natural language interfaces for distributed appointment-scheduling agents. Their research addressed core challenges in translating conversational dialogue into structured scheduling actions, including accurate parsing of temporal expressions, managing under-specified requests requiring clarification, and supporting coordination between human and machine agents through email-based interactions[1][2].

Vishwanath and Vig's *Meeting Bot* research[3] introduced reinforcement learning paradigms into dialogue-based scheduling to overcome limitations associated with rigid, heuristic-driven decision-making. They formulated meeting scheduling as a sequential decision-making problem in which an agent learns optimal strategies from cumulative interaction experience.

Yang and Pattan's research on the *Business Meeting Organizer*[4] contributed significant advancements by introducing mobile context awareness into distributed, agent-based scheduling. Their system went beyond conventional agenda planning by incorporating context-sensitive digital secretaries capable of arranging meeting logistics while considering dynamic user-related factors, including current location, predicted future location derived from itinerary data, time-of-day preferences, and real-time environmental constraints such as traffic conditions[4].

Rastogi, Zang, et al. from Google brought up a schema-guided method for creating a scalable virtual assistant. At the core of their work was the use of an explicit schema to serve as the conversational framework for the services, detailing the required slots—like date, time, and participants—together with the possible values and constraints. Dialogue state managers continually reflected the updated structured versions of the user inputs across the different conversational turns, thus facilitating precise multi-turn understanding. Their approach can be directly transferred to the scenario of meeting scheduling where the schema would be the one to determine the indispensable items such as date, time, duration, participants, and topic necessary for calendar operations [?].

The Task Mining research of Lee, Miller, et al. revolved around the automation of post-meeting workflows with a particular focus on the extraction and tracking of action items. Their solution automatically extracted the assignments of tasks from the transcripts of the meetings through the use of pattern recognition based on NLP technology, thus it would get the ownership, deadline, and description of the item, and move these data points into the task management systems. The authors of this paper have implemented the scheduling bot concept by taking on the meeting life cycle challenge, hence demonstrating that language technology is capable of retrieving the structured commitments embedded in the chaos of the unstructured conversational data. The research brought to light the problem of "action item amnesia" — the loss of the achieved outcomes of meetings, and it also proposed the automatic extraction as the most suitable answer to this problem [?].

While a lot of ground has been covered, there are still several limitations of the existing research that have inspired the following study:

- Restrictions of Input Modalities: Most of the spoken language understanding and conversational AI systems today are

© JAN 2026 | IRE Journals | Volume 9 Issue 7 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I7-1713229

designed for text-based inputs, which inherently makes the interaction less accessible and natural because these are speech-first scenarios.

- Incomplete Automation: The majority of the systems in question have a human-in-the-loop verification mechanism that is responsible for the quality control, which limits the scalability and the speed of the whole process. The problem of how to make scheduling completely autonomous is still not solved.
- Limited Conflict Resolution: As it stands, most of the tools can only notify you of the conflicts that have occurred, but they do not do much in terms of resolving the conflicts, such as providing alternative time slots for meetings or rescheduling through optimization methods.

III. RESEARCH AND METHODOLOGY

A. Research Approach

This study uses design science research methodology, which focuses on creating, evaluating, and iteratively refining an artifact through six stages: Requirements Analysis, Architecture Design, Component Development, Integration and Testing, Evaluation, and Refinement.

B. Speech Recognition Methodology

After audio preprocessing (noise reduction, normalization, segmentation), the speech-to-text conversion applies ASR inference to produce the text transcription along with confidence scores.

C. Natural Language Processing Methodology

The NLP pipeline maps the steps from the transcribed text to prompt creation, LLM inference, response parsing, and entity normalization.

D. Conflict Detection Methodology

Conflicting event identification is done through temporal reasoning on calendar information and employing interval overlap detection algorithms.

E. Scope of the Study

This research scope includes:

- Speech-to-text conversion based on established ASR frameworks
- Natural language understanding via GPT-4o-mini
- Calendar integration using Google Calendar API
- Conflict detection through temporal reasoning algorithms
- Email notification using SMTP
- Web-based user interface

F. System Architecture

The SmartMeet system architecture is modular, layered design that revolves around five major subsystems.

G. Presentation Layer

The presentation layer is the front end that interacts with users for capturing scheduling requests and displaying system responses. It consists of a responsive web application, a speech capture module, and response rendering components.

H. Speech Recognition and NLP Layer

This layer converts differently formatted oral input into correctly programmed scheduling instructions through two consecutive stages: Automatic Speech Recognition (ASR) and Natural Language Understanding (NLU) by means of the Azure OpenAI's GPT-4o-mini model.

I. Business Logic and Orchestration Layer

The layer carries the core scheduling decision-making capabilities such as entity validation, calendar integration, conflict detection, alternative slot generation, meeting link generation, and workflow orchestration.

J. Communication and Notification Layer

It is responsible for the outward interaction of the meeting participants through the creation of an email and the SMTP-based dispatch.

K. Data Management and Persistence Layer

The echelon marshals the application state and configuration, as well as audit logging, session management, and credential storage.

L. Algorithms and Mathematical Formulations

1) *1. Intent Extraction Probability Model:* The NLU task is to find:

© JAN 2026 | IRE Journals | Volume 9 Issue 7 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I7-1713229

$$S^* = \arg\max_S P(S|T) \quad (1)$$

Bayes' theorem is used:

$$P(S|T) = \frac{P(T|S) \cdot P(S)}{P(T)} \quad (2)$$

2) 2. Temporal Conflict Detection Function: Consider a meeting request as

$$M = (P, t_{start}, t_{end})$$

where

$$P = \{p_1, p_2, \ldots, p_n\}$$

is the set of participants. The conflict detection function is:

$$conflict(M, p_i) = \bigcup_{j=1}^{m} overlap(M, e_j^i) \quad (3)$$

3. Alternative Slot Scoring Function For candidate slot s:

$$score(s) = w_1 \cdot f_{proximity}(s) + w_2 \cdot f_{availability}(s) + w_3 \cdot f_{workHours}(s) - w_4 \cdot f_{fragmentation}(s) \quad (4)$$

where:

$$f_{proximity}(s) = e^{-\alpha \cdot |t_s - t_{proposed}|} \quad (5)$$

M. Time Complexity Analysis

For n participants, each with average k calendar events:

$$T_{conflict}(n, k) = O(n \cdot k) \quad (6)$$

For alternative slot generation with m candidate slots:

$$T_{alternatives}(n, k, m) = O(m \cdot n \cdot k + m \log m) \quad (7)$$

IV. RESULTS AND DISCUSSION

A. Speech Recognition Accuracy

The ASR component achieved an overall Word Error Rate (WER) of 8.3% on clean audio samples and 14.7% on noisy samples, with 89% of transcriptions containing sufficient accurate content for downstream NLP processing.

B. Entity Extraction Performance

The GPT-4o-mini-based NLU module demonstrated strong performance with macro-average F1-score of 0.90, exceeding the 0.85 target objective.

C. Conflict Detection Accuracy

- True Positive Rate: 96%
- False Positive Rate: 3%
- False Negative Rate: 4%

D. End-to-End Success Rate

SmartMeet achieved 81% total success rate across 100 test scenarios, with 94% success on conflict-free scenarios and 68% on conflict scenarios.

E. Latency Analysis

End-to-end latency metrics:

- Average Latency: 6.8 seconds
- Median Latency: 5.9 seconds
- 95th Percentile: 11.2 seconds

This is equivalent to a time reduction of approximately 85% when compared to the manual scheduling baseline.

F. User Satisfaction Assessment

After-task questionnaire (7-point Likert scale):

- Ease of Use: 6.1
- Time Savings: 6.4
- Accuracy: 5.7
- Overall Satisfaction: 5.9

V. PROPOSED WORKFLOW

The SmartMeet service workflow is a sequence of eight stages, each performing the specified transformations on the scheduling data.

A. Stage 1: Speech Capture and PreProcessing

The user schedules a meeting by recording his/her voice via the web interface. To obtain audio data (with user consent), the system turns on microphone access and starts capturing sound files at a 16 kHz sampling rate. Audio recording is stopped by a user click or is automatically stopped if there is silence for 2 or more seconds. The recorded audio is put through noise reduction and normalization, resulting in a clean audio signal that can be easily transcribed.
Input: Unprocessed audio data from the microphone
Output: Cleaned audio data
Duration: Indefinite (user-controlled, usually 5–15 seconds)

B. Stage 2: Automatic Speech Recognition (ASR)

The cleaned audio is sent to the ASR engine (Web Speech API or Whisper). The recognition engine achieves the goal by performing acoustic and language modeling to generate the text transcription. The system gets the transcribed text together with the confidence scores at the word level. If the average confidence is under the threshold (0.6), the system asks the user to repeat the request.
Input: audio buffer
Output: transcription of the text along with confidence scores
Duration: 1–3 seconds
Example Output:

© JAN 2026 | IRE Journals | Volume 9 Issue 7 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I7-1713229

"Schedule a meeting with Rohan and Ali tomorrow at 3 PM for one hour to discuss the project timeline."

C. Stage 3: Natural Language Understanding (NLU)

The system receives the transcribed text, then reformats it into a prompt for the GPT-4o-mini model. The prompt guides the LLM to identify the structured scheduling entities from the text and return them in JSON format.

The system sends the prompt to the Azure OpenAI API and gets the structured response. The JSON is then converted from the string format into the system's internal data structures and the system confirms that the required fields (date, time, participants) are there.

Input: Text transcription
Output: Structured scheduling object (JSON)
Duration: 0.5–2 seconds
Example JSON Output:

```
{
"participants":
["rohanprasad065@gmail.com"],
"date": "2025-11-27",
"time": "15:00",
"duration": 60,
"agenda": "Discuss project
timeline"
}
```

D. Stage 4: Entity Validation and Normalization

The system conducts validation of the extracted entities:

- Makes sure that the email addresses are of the correct format
- Changes relative dates (like "tomorrow") to absolute dates
- Converts time to 24-hour format
- Verifies that the time is consistent (end time ¿ start time)
- Checks that time duration is in the normal range (15 minutes to 8 hours)

When validation fails or extraction of entities is incomplete, the system produces questions for clarifying the user. The user's answers are once more processed through the NLP pipeline.

Input: Raw structured entities
Output: Validated and normalized scheduling parameters
Duration: 0.1 seconds

E. Stage 5: Conflict Detection

For each participant (including the organizer), the system:

- Queries Google Calendar API for events in the proposed time window
- Performs interval overlap analysis using the conflict detection algorithm
- Identifies conflicting events and participants with conflicts If no conflicts are detected, the workflow proceeds to Stage

If conflicts exist, the workflow proceeds to Stage 6.
Input: Validated scheduling parameters
Output: Conflict report indicating conflicting events per participant
Duration: 0.5–2 seconds

F. Stage 6: Alternative Slot Generation

The system generates alternative time slot recommendations:

- Expands the search window to ±2 hours around the proposed time
- Discretizes the window into 15-minute candidate slots
- Checks participant availability for each candidate slot
- Scores candidates using the defined scoring function
- Sorts candidates and selects the top three conflict-free options

Alternatives are presented to the user, who may select one or request more suggestions. The system updates scheduling parameters accordingly.

Input: Conflict report and original parameters
Output: Ranked list of alternative time slots
Duration: 1–3 seconds

G. Stage 7: Calendar Entry Creation

The system creates calendar entries:

- Generates a Google Meet link via the conference Data API
- Constructs the event object (summary, start, end, attendees, description)
- Submits a POST request to Google Calendar API
- Receives event ID and stores it in the audit database

Input: Final validated scheduling parameters Output: Created calendar event with unique event ID
Duration: 0.5–1.5 seconds

H. Stage 8: Email Notification Dispatch

The system generates and sends email invitations:

- Populates email template using Jinja2
- Creates an `.ics` calendar attachment

- Sends personalized emails via SMTP to all participants
- Logs all delivery status information

Finally, the system displays a confirmation message in the user interface and triggers a browser notification.

Input: Created calendar event
Output: Email invitations sent to all participants
Duration: 1–3 seconds

I. Total Workflow Duration

The complete workflow requires approximately 5–15 seconds from speech input to final confirmation, representing over 90 percent time reduction compared to manual scheduling (typically 5–10 minutes).

VI. CONCLUSION

Presented here is SmartMeet, a radically new intelligent meeting scheduling system achieving complete automation via the integration of speech recognition, natural language processing, large language models, and robotic process automation. The study showed that state-of-the-art AI technologies reach high accuracy (F1-score 0.90) for scheduling applications in an enterprise environment, with 81% end-to-end success rate and 6.8-second median latency that corresponds to a time reduction of approximately 85% compared to manual scheduling.

Besides the specific performance metrics, this work has a broader impact on conversational AI, scheduling algorithms, and AI-RPA integration domains. The limitations of the system point to the next research frontier, with proposed upgrades dealing with preference learning, recurring meeting support, and multiplatform integration.

SmartMeet, from a real-world standpoint, embodies the potential of voice-first enterprise automation. As companies implement digital transformation strategies, intelligent assistants that can convert natural communication into structured system actions will be indispensable productivity tools. This research makes the case for the existence of fully automated, speech-driven meeting scheduling as a technically feasible task that achieves the necessary reliability for real-life deployment.

A. Future Enhancements

Proposed enhancement roadmap:

- Advanced speech processing with Whisper integration
- Intelligent preference learning using reinforcement learning
- Recurring meeting management
- Multi-platform calendar integration
- Contextual intelligence for meeting type classification
- Advanced conflictresolution with multi-objective optimization
- Conversational interface with multi-turn dialogue
- Mobile native applications
- Enterprise features (RBAC, SSO, audit logging)
- Analytics and insights dashboard

B. Limitations

SmartMeet exhibits several limitations:

- Language and accent constraints (English only)
- Single-instance meetings only (no recurring meetings)
- Simple conflict resolution without preference learning
- Limited to Google Calendar platform
- Acoustic environment sensitivity
- Privacy and consent boundaries
- Limited context awareness
- No learning from feedback
- Dependency on external services

REFERENCES

[1] J. Cranshaw et al., “Calendar.help: Designing a Workflow-Based Scheduling Agent with Humans in the Loop,” in *Proc. CHI Conf. Human Factors in Computing Systems*, Denver, CO, May 2017, pp. 2382–2393.

[2] S. Busemann, T. Declerck, et al., “Natural Language Dialogue Service for Appointment Scheduling Agents,” in *Proc. 5th Conf. Applied Natural Language Processing*, Washington, DC, 1997, pp. 25–32.

[3] D. Vishwanath and L. Vig, “Meeting Bot: Reinforcement Learning for Dialogue Based Meeting Scheduling,” in *AAAI Conf. Artificial Intelligence*, New Orleans, LA, Feb. 2018, pp. 8135–8136.

[4] K. Yang and N. Pattan, “Multi-Agent Meeting Scheduling Using Mobile Context,” in

IEEE/WIC/ACM Int. Conf. Intelligent Agent Technology, Compiegne, France, Sept. 2005, pp. 488–491.

[5] E. Shakshuki, H. Koo, and D. Benoit, "A Distributed Multi-Agent Meeting Scheduler," *J. Computer Science and Technology*, vol. 18, no. 3, pp. 295–304, May 2003.

[6] S. D. Gopalan and D. A. Sontag, "Learning to Schedule Meetings with a Calendar Virtual Assistant," arXiv:2109.04596, Sept. 2021.

[7] M. G. Lee et al., "TaskMining: A Task-Centric Meeting Assistant," *Proc. ACM on Human-Computer Interaction*, vol. 4, no. CSCW2, pp. 1–26, Oct. 2020.

[8] Y. K. Thang et al., "A Context-Aware Personal Assistant for Managing Social Events," in *24th Int. Conf. Intelligent User Interfaces*, Marina del Ray, CA, Mar. 2019, pp. 423–434.

[9] B. A. Plummer et al., "Dialogue Learning with Human-in-the-Loop," in *Proc. 34th Int. Conf. Machine Learning*, vol. 70, Sydney, Australia, Aug. 2017, pp. 2832–2841.

[10] A. Rastogi et al., "Schema-Guided Dialogue State Tracking for Virtual Assistant Scheduling," in *Proc. 21st Annual SIGdial Meeting on Discourse and Dialogue*, Virtual, July 2020, pp. 1–11.

[11] L. P. Willcocks, M. Lacity, and A. Craig, "The IT Function and Robotic Process Automation," *LSE Outsourcing Unit Working Paper Series*, Rep. 15/05, 2015.

[12] A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems 30*, Long Beach, CA, Dec. 2017, pp. 5998–6008.

[13] T. B. Brown et al., "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems 33*, Virtual, Dec. 2020, pp. 1877–1901.

[14] A. Radford et al., "Robust Speech Recognition via Large-Scale Weak Supervision," arXiv:2212.04356, Dec. 2022.

[15] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. Stanford, CA: Stanford University, 2023.

[16] D. Xiao, M. Chen, and J. Zhang, "Enterprise Calendar Integration Patterns Using RESTful APIs," *IEEE Trans. Services Computing*, vol. 12, no. 4, pp. 654–667, July–Aug. 2019.

[17] A. Cambria and B. White, "Jumping NLP Curves: A Review of Natural Language Processing Research," *IEEE Computational Intelligence Magazine*, vol. 9, no. 2, pp. 48–57, May 2014.

[18] M. Henderson, B. Thomson, and S. Young, "Deep Neural Network Approach to Dialogue State Tracking Challenge," in *Proc. SIGDIAL 2013 Conf.*, Metz, France, Aug. 2013, pp. 467–471.

[19] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Hoboken, NJ: Pearson, 2020.

[20] P. Lison and C. Kennington, "OpenDial: A Toolkit for Developing Spoken Dialogue Systems with Probabilistic Rules," in *Proc. ACL 2016 System Demonstrations*, Berlin, Germany, Aug. 2016, pp. 67–72.