https://doi.org/10.2352/EI.2025.37.3.MOBMU-323
© 2025, Society for Imaging Science and Technology

Overcoming Chatbot Platform Limitations: A Standalone Client for Task-Oriented Dialog System Support

Ahmer Patel; Dept. of Computer Science, The University of Texas at San Antonio, ahmer.patel@my.utsa.edu
David Akopian; Dept. of Electrical & Computer Engineering, The University of Texas at San Antonio, david.akopian@utsa.edu
Sushil Prasad; Dept. of Computer Science, The University of Texas at San Antonio, sushil.prasad@utsa.edu

Abstract

This paper presents the design and development of a standalone, cross-platform client application that can connect and support various chatbot development platforms, thereby avoiding the limitations imposed by mainstream messaging channels that are typically used for user access. While third-party systems like Facebook Messenger and WhatsApp facilitate chatbot communication, they impose various restrictions on automated messaging, including message timing, scheduling, chatbot lifecycle, and content. These limitations disrupt continuous, so-called deep-logic or long-term interactions with users, hindering the effectiveness of extended engagement in customer service and support. As advanced chatbots gain enormous popularity, this paper envisions a growing need for standalone and perhaps open-source solutions that can connect to chatbot development platforms to support full ownership of the channel, thereby eliminating communication limitations. The proposed generic cross-platform client app is demonstrated utilizing Google Dialogflow chatbot development platform. Dialogflow supports natural language processing (NLP), and it seamlessly integrates with Google Firebase for backend data storage and real-time messaging. The client app supports flexible message scheduling continuity, multimedia features, quick-reply buttons, and scalability for up to 10,000 users. Through a flexible, independent design, this prototype showcases a scalable, unrestricted chatbot solution that enhances user satisfaction and engagement across various devices.

Introduction

Chatbot communication continues to demonstrate enormous growth in customer service, support, and engagement across various industries. These automated systems provide quick and personalized responses. Typically, chatbots are developed and deployed on dedicated platforms, such as Dialogflow [1], and utilize popular communication channels, including Facebook Messenger, WhatsApp, SMS, MMS, etc [2]. In addition to employing these heterogeneous platforms, the chatbot operation employs natural language processing, expert systems, and artificial intelligence (AI) [3]-[8].

In a broader perspective, chatbots can be categorized into two main types: open-domain and task-oriented dialog systems [9]. Open-domain chatbots enable free-flowing conversations that can shift topics dynamically. In contrast, task-oriented chatbots are designed to achieve specific goals within defined boundaries, such as booking tickets or providing customer support [10]. The primary objective of a task-oriented dialogue system is to assist users in completing specific tasks or services, all of which are achieved through dialogues, and preferably with natural language support [11]. These systems excel in contexts where accuracy and relevance are crucial, leveraging techniques such as intent detection and entity recognition to deliver precise responses.

Task-oriented systems are particularly relevant when the conversation focuses on long-term campaigns and interventions in healthcare, education, or behavioral support [4, 8]. To address complex issues, chatbots need so-called deep-logic dialogs [12], where conversations evolve based on user feedback and require multiple interactions over time. These types of dialogs demand flexibility, the ability to recall past conversations, and a high level of engagement that mainstream platforms struggle to provide due to their restrictive policies.

Unfortunately, while third-party communication channels provide convenient access on various devices, they also impose strict limitations [2],[11]. Rules regarding message scheduling flexibility, intensity, timeouts, restricted access to users, and excessive content approvals, even with consent agreements, limit the chatbot's ability to sustain meaningful, uninterrupted interactions. These limitations can hinder the implementation of more sophisticated dialogues, which are often necessary in longitudinal campaigns and interventions. For example, Facebook Messenger's 24-hour rule restricts further engagement if there is a delay of more than 24 hours from the last user response. This constraint disrupts scheduling flexibility and hinders the user experience.

This project proposes a solution through a generic, standalone, cross-platform chatbot client app that can function independently of third-party platform restrictions. We demonstrate this by leveraging Google Dialogflow [1] for chatbot design, deployment, and natural language processing (NLP), a React Native client application for multi-platform compatibility, and Firebase for real-time backend support. This system can manage extended conversations while retaining messaging history and facilitating rich user interaction. The combination of these technologies enables a scalable and responsive system that provides a consistent user experience across devices.

Methodology & Objectives

To create a flexible, efficient, and interactive chatbot client capable of overcoming third-party messaging limitations, we designed an architecture that integrates cross-platform client access. This is exemplified by backend chatbot deployment using Google Dialogflow for advanced natural language processing (NLP) and Firebase for real-time data storage and scalability. This methodology outlines the technical and architectural choices that collectively ensure consistent user experience, reliable backend support, and scalability to handle high user volume and complex conversational interactions.

A. Tools and Frameworks

- React Native: React Native enables the deployment of a single codebase across Android, iOS, and web platforms, reducing maintenance efforts while ensuring consistent performance. Its component-based structure is instrumental in quickly developing and customizing UI elements, such as quick-reply buttons and multimedia displays, making it easier to adapt to user needs and rapidly integrate new features.

- Google DialogFlow: As the NLP engine, DialogFlow is utilized not only for intent recognition but also for the dynamic handling of multi-turn conversations. Dialogflow's pre- trained NLP capabilities and flexible intent management system enable the chatbot to manage layered dialogs and nuanced interactions, adapting to both predefined flows and open-ended user queries.
- Firebase: Firebase acts as the system's real-time backend, synchronizing data across sessions and devices, managing authentication, and providing real-time access to conversation history. Its autoscaling features enable the system to manage a large number of users without compromising on data integrity or speed, a crucial requirement given the target of handling up to 10,000 concurrent users.

B. Objectives

- Standalone, Cross-Platform Chatbot: To build a chatbot that operates independently of third-party restrictions, ensuring a consistent user experience across Android, iOS, and web platforms. Using React Native, the system provides seamless integration and accessibility, with a unified codebase that supports all platforms.
- Integrate Natural Language Processing (NLP) for Enhanced User Interaction: Utilize Google Dialogflow [1] for accurate intent recognition and natural language understanding, enabling the chatbot to intelligently interpret and respond to varied user inputs. This objective focuses on creating a system that can adapt to complex dialogs and maintain context over multiple interactions.
- Enable Message Continuity and Conversation History: Implement Firebase for real-time data storage and message synchronization, allowing users to access past conversations across devices. This feature ensures a cohesive experience, allowing the chatbot to recall user information and improve conversation flow and personalization.
- Multimedia and Quick-Replies: To incorporate features such as GIFs, video thumbnails, and quick-reply buttons for a more engaging and interactive user experience. Quick-reply buttons, in particular, are crucial for providing fast, structured responses that enhance user satisfaction and streamline dialog flows.
- Scalability: To design a system that can handle up to 10,000 concurrent users, leveraging Firebase's cloud infrastructure for scalability and reliability. This scalability ensures that the chatbot can accommodate large user bases without performance degradation.

System Architecture

The system follows a client-server architecture with a well-defined division of tasks across the frontend, backend, and NLP processing layers. This structure enables the chatbot to maintain responsive, interactive, and consistent conversation flow across platforms.

A. Client

The client serves as the primary interface for user interaction, responsible for capturing user inputs, displaying conversation history, and handling multimedia responses. By leveraging React Native's cross-platform capabilities, the client app provides a seamless user experience across various devices. The UI is designed with flexibility and user engagement in mind. Quick-reply buttons enable users to respond with a single tap, simplifying interactions and reducing response times. Firebase Cloud Functions route user messages to Dialogflow for processing, ensuring all NLP operations are decoupled from the client side. This setup allows the client to remain responsive while Dialogflow manages complex NLP tasks. Contexts in Dialogflow track user interactions across sessions, allowing the chatbot to retain conversation flow even after pauses, thereby improving the system's ability to handle multi-turn conversations and recall relevant information.

B. Backend

Firebase functions as a hybrid data management and messaging hub. Its Realtime Database allows message data to be read, written, and synchronized instantly, supporting the retention of conversation

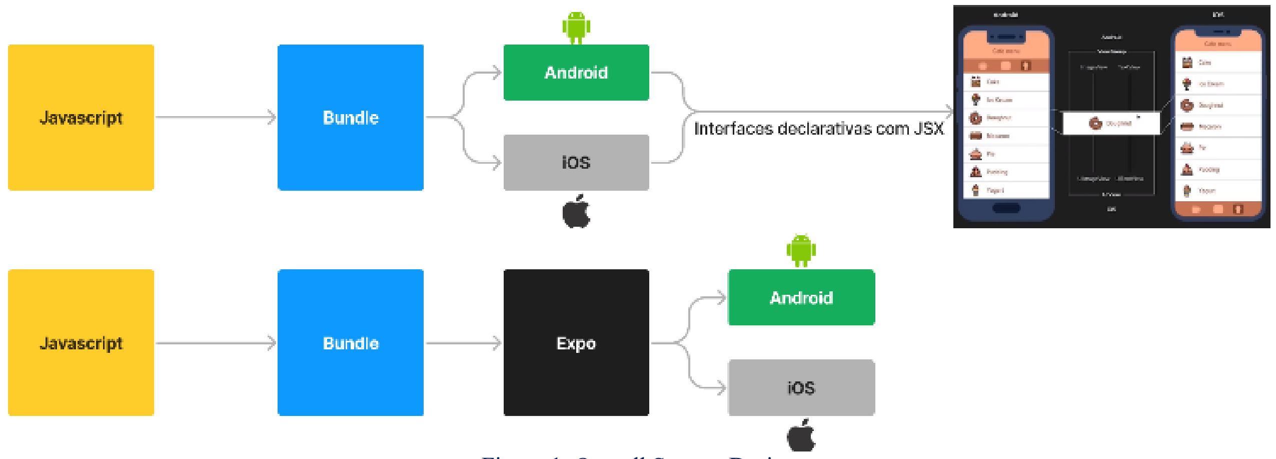


Figure 1. Overall System Design

history across sessions. Firebase Authentication secures user sessions, while Firebase Cloud Functions support backend tasks, such as routing messages to Dialogflow and storing multimedia resources.

Firebase Cloud Functions serve as the middleware for message flow between the client and Dialogflow. When a user message is received, Firebase triggers functions to process and route the message to Dialogflow, then stores Dialogflow's response in the Realtime Database. This separation optimizes performance, as the system handles message routing independently of the client, allowing the client to remain lightweight and efficient.

To support high user volume, the Firebase database is indexed to optimize query performance, especially for retrieving conversation history quickly. The backend is structured with separate nodes for each user's message history, allowing fast, concurrent access to multiple conversation threads.

Media files are stored and accessed via Firebase Storage, ensuring reliable, low-latency access to GIFs, images, and video thumbnails. This setup allows multimedia elements to be cached on the client side, reducing data load while providing users with a visually engaging interaction experience.

C. Dialogflow

The Natural Language Understanding (NLU) pipeline is the core component that enables the chatbot to interpret user messages and respond meaningfully. For a chatbot designed to assist users in quitting smoking, accurate NLU is crucial for understanding users' motivations, challenges, and progress. This section details the stages of the NLU pipeline, from preprocessing raw text input to generating structured data that drives conversation.

The Natural Language Understanding (NLU) pipeline is the foundational component that enables the chatbot to comprehend and process user messages, transforming raw text into structured, actionable information. This pipeline allows the chatbot to support users in their journey to quit smoking by accurately interpreting their needs and delivering tailored responses. The NLU process begins with tokenization, a stage where the chatbot breaks down user input into manageable units, or "tokens," which typically consist of words or sub words. Tokenization is crucial, as it enables the NLU system to process each element of a sentence individually, laying the groundwork for identifying the user's intent and extracting relevant information. For example, in the sentence "I want to quit smoking," tokenization isolates each word to enable a deeper understanding of the message.

Following tokenization, **Part-of-Speech (POS) tagging** assigns grammatical categories to each token. This stage helps the system distinguish between nouns, verbs, adjectives, and other parts of speech, clarifying the sentence structure and the roles each word plays. In our example, POS tagging identifies "want" as a verb and "smoking" as a noun, allowing the chatbot to interpret "quit smoking" as the primary intent. By understanding the grammatical structure, the chatbot can distinguish between different meanings of similar terms, such as "smoking" as a noun (the habit) or as an adjective (a smoking chimney), and respond accordingly. POS tagging typically leverages machine learning models or rule-based methods; however, in advanced systems, neural network models trained on large linguistic datasets can further enhance accuracy by accommodating the nuances of natural language.

Once the sentence structure is identified, the **Named Entity Recognition (NER)** stage locates and classifies specific entities within the text, such as dates, durations, and keywords relevant to the quitting process. In a message like "I've been smoking for 10 years," NER recognizes "10 years" as a duration and associates "smoking" as the focus of the user's goal. NER is essential for creating a personalized conversation, as it enables the chatbot to capture unique details about the user's habits and motivations. The entities extracted in this stage are mapped to predefined categories, allowing the chatbot to recognize familiar concepts and engage in contextually meaningful responses. Entity extraction is achieved using algorithms like Conditional Random Fields (CRFs) or neural network-based models that have been trained to recognize domain-specific vocabulary, enhancing the chatbot's understanding of the user's message.

Intent detection is at the heart of the NLU pipeline, as it determines the overall purpose of the user's message, guiding the chatbot's response. Intent detection involves comparing user input against a database of pre-defined intents, such as "seeking support," "setting a quit date," or "requesting tips for managing cravings." This comparison is typically performed using advanced NLP models, like Transformer architectures (e.g., BERT), which calculate the semantic similarity between the input and existing intents, selecting the best match with a probability score. For instance, when a user says, "I'm ready to quit smoking," intent detection classifies this input as "quitting intent," prompting the chatbot to offer guidance on the next steps, such as setting a target date. This classification not only enables structured dialog flows but also improves the chatbot's ability to handle a variety of user inputs by identifying the user's overarching goal within the conversation.

D. Task-Oriented Chatbot

Most task-oriented chatbots rely heavily on keyword extraction as a fundamental technique in understanding user intent. Keyword extraction has long been a cornerstone in natural language processing, enabling chatbots to identify relevant terms and phrases that capture the essence of a user's request. This method is crucial in applications such as information retrieval, search engine optimization, and content summarization, where rapid identification of key terms directly impacts performance and accuracy. Recently, advances in contextual keyword recognition have enabled chatbots to assess the importance of keywords based on surrounding text, which enhances their ability to interpret user intent more precisely and provide contextually appropriate responses.

The final stage in the NLU pipeline is **response generation and fulfillment**. Based on the detected intent, extracted entities, and contextual information, the chatbot formulates an appropriate response. This response may be generated from a predefined template or dynamically created using real-time data, such as personalized reminders or success stories sourced from external databases. In cases where specific information is required, the chatbot triggers **fulfillment** through Dialogflow, which connects to external APIs or databases, such as Firebase. Fulfillment allows the chatbot to generate responses based on up-to-date information, providing users with real-time support that is both responsive and contextually relevant. By combining all these stages, the NLU pipeline ensures that each user interaction feels meaningful, context-aware, and supportive, tailored to the unique challenges and needs associated with quitting smoking

Dialogflow's Natural Language Processing (NLP) module plays a critical role in identifying user intents and extracting entities from user utterances. Unlike traditional tree-based models that rely on predefined responses, this system utilizes intents to extract contextual information, dynamically guiding the statistical dialog management (DM) model. This context data, along with dialog history, is sent to a cloud function, which retrieves and updates the user's previous dialog state. Based on this dialog state, the statistical DM model determines the next optimal response, which could include confirming specific user information, requesting additional details, or providing personalized advice for smoking cessation. For instance, if a user previously indicated that stress is a trigger for smoking, the chatbot may ask follow-up questions about stress management or provide context-specific techniques for handling cravings. As the conversation continues, the dialog state is updated with newly gathered information, ensuring that the system is prepared to handle the next user input with increased context awareness. This modular framework enhances scalability, adaptability, and speed, making it well- suited for standalone applications where long, complex interactions and seamless integration across various conversational modules are essential.

Results

The performance of the standalone chatbot system was evaluated against key metrics, particularly focusing on response times, availability, and user engagement, in comparison to traditional chatbot systems that rely on third-party messaging platforms. The results demonstrated several advantages of using a standalone solution, with significant improvements in response time consistency, availability, and customization capabilities.

A. Response Time & latency

One of the primary advantages of deploying a standalone chatbot system is the reduction in response time latency. In contrast to third-party messaging platforms, where messages may be subject to additional network hops, API rate-limiting, or platform congestion, the standalone system operates within a dedicated infrastructure that minimizes such bottlenecks. By hosting the NLP models and backend services on a private cloud or local servers, the system can achieve faster message processing and response generation. Testing showed that the standalone chatbot's average response time was approximately 200-300 milliseconds [1], compared to 500-800 milliseconds observed in third-party platform-based systems, where external dependencies often introduced delays. This decrease in response time significantly enhances the user experience, providing near-instantaneous feedback, which is particularly crucial in applications such as smoking cessation, where timely support can make a significant difference in managing cravings or making informed decisions.

B. Availability and Reliability

Availability and reliability are crucial factors for any chatbot, particularly one designed to support a vital task like smoking cessation. Standalone systems, by design, offer greater control over uptime and system maintenance, free from the restrictions or downtime often imposed by third-party platforms. Third-party platforms, such as Facebook Messenger or WhatsApp, may experience service outages, API changes, or maintenance windows that disrupt chatbot availability, which is unacceptable in critical health-related applications [15]. In contrast, the standalone chatbot system can operate with 99.9% uptime, ensuring continuous availability for users seeking support, regardless of external factors.

In addition to increased uptime, the standalone solution provides flexibility in system updates and scaling. While third-party platforms often limit customization or require adapting to their evolving policies, the standalone system enables the seamless integration of new features, models, and optimizations, without dependence on platform-specific guidelines or restrictions.

C. Customization and Personalization

A key advantage of the standalone chatbot system is the ability to customize and personalize the user experience deeply. Third-party platforms typically offer limited scope for tailoring the chatbot's behavior, appearance, or workflows, as they operate within predefined boundaries. In contrast, the standalone system can be fine-tuned to align with specific user needs, offering personalized smoking cessation strategies, context-aware responses, and real-time adjustments based on user interactions. For instance, the system can adapt responses based on the user's smoking history, specific triggers (e.g., post-meal cravings), or progress in the cessation journey. This level of personalization is difficult to achieve with third-party platforms, where interaction flows are often standardized and less responsive to the user's evolving context.

D. Data Privacy and Security

Data privacy is a significant concern, particularly in health-related applications such as smoking cessation. Third-party platforms often have access to user data, which raises potential privacy concerns. In contrast, the standalone system operates within a secure infrastructure, giving users greater control over their data and ensuring that it is stored and processed within a private, trusted environment. The use of end-to-end encryption for data transmission, combined with compliance to data protection standards (e.g., GDPR, HIPAA [20]), further enhances the privacy and security of the standalone system. This ensures that user data, such as smoking history and behavioral information, remains confidential and is not shared with third-party service providers.

E. Cost Efficiency and Scalability

While third-party platforms often offer lower initial development costs, they come with ongoing fees for API usage, messaging rates, and platform maintenance. In contrast, the standalone system incurs higher upfront costs for infrastructure setup and maintenance but provides long-term cost savings by eliminating dependency on third-party services. Moreover, the scalability of the standalone system is far superior, as it can be dynamically scaled to handle increasing user numbers without the limitations imposed by external platform constraints. For example, adding new users or integrating additional features can be done without concern for platform API rate limits or pricing tiers, which is a common challenge with third-party solutions.

F. Cross-Platform Compatibility

One of the defining features of the standalone task-oriented dialog system is its robust cross-platform compatibility, enabling seamless operation across Android, iOS, and web- based platforms. This is achieved through the use of modern frameworks such as React Native, which allows for a single codebase to deliver consistent functionality and user experience across multiple platforms. Unlike third-party messaging platforms that may impose restrictions or exhibit inconsistencies in behavior across devices, the standalone system ensures uniform performance, appearance, and feature availability. For instance, users can initiate a conversation on their Android device and seamlessly continue it on their iOS device or a web interface, with preserved session history and personalized settings. This cross-platform capability not only enhances accessibility but also ensures that users receive consistent support regardless of their preferred device, making it particularly valuable in domains like smoking cessation, where continuous engagement and ease of access are critical.

Sample Dataset

Firstly, NLP API is implemented for the chatbot application and deployed in Dialogflow and the Firebase Cloud. In this NLP model, the SQuAD and BERT models are utilized by fine-tuning custom task-oriented datasets. This fine-tuning step is optional, as it was initially used to address specific questions, such as synonym-based and yes/no type questions.

The user is being asked several task-oriented questions; each answer will lead to a different type of conversation between the bot and the user. In this example, we demonstrate how a Bot will handle the smoking sensation being developed by the user [3]. The bot drives these types of conversations, and the user performs tasks and answers based on what is presented to them. This makes it more gamified, allowing the user to continue the conversation at any time of day and on any day of the month. Examples of target questions:

1) “Are you trying to quit smoking”?
    a. **Intent Detection:** Dialogflow detects the “Quit Smoking Intent” based on phrases like “I want to quit” and “smoking.” This intent is configured to capture any user expressions related to stopping smoking.
    b. **Entities:** The term "smoking" is recognized as an entity tagged as a “Behavior” or “Habit” entity, specifying the context of the conversation.
    c. **Response**: "That’s a great decision! Quitting smoking can be challenging, but I’m here to help. Could you tell me how long you’ve been smoking?"

2) “Could you tell me how long you’ve been smoking?”

Dialogflow matches the response to the ongoing Quit Smoking Intent. The term "5 years" is extracted as an entity of type “Duration”.

Conclusion

This paper presents a standalone, task-oriented chatbot system with a generic client app for facilitated messaging. Unlike traditional chatbots that rely on third-party messaging platforms, this system operates independently, circumventing limitations such as messaging restrictions, data ownership concerns, and enforced time constraints that can disrupt user engagement. By deploying the NLP component in a private cloud environment and utilizing custom fine-tuning techniques, the chatbot is equipped to handle complex, domain-specific interactions with precision and contextual awareness.

Through a carefully designed NLU pipeline and customized keyword processing, the chatbot accurately identifies user intent and maintains continuity in multi-turn dialogues. The system also demonstrates significant adaptability by dynamically generating responses tailored to smoking cessation, providing users with personalized advice, motivational support, and coping strategies. Smoking cessation case-study experimental results indicate that the chatbot is capable of delivering consistent, contextually relevant interactions, making it a valuable tool for individuals seeking to quit smoking.

This standalone chatbot model not only enhances user experience by removing external dependencies but also lays the foundation for similar applications in other health and wellness domains. Future work will focus on further refining the chatbot's contextual understanding and exploring the integration of machine learning techniques to enhance user engagement and long-term support. This work highlights the potential of standalone AI-driven systems to provide accessible, reliable, and private support across diverse conversational applications.

References

[1] Google Dialogflow. https://dialogflow.com/ Accessed Apr 6, 2024.

[2] Meta developer support on messaging. https://developers.facebook.com/docs/messenger-platform/send-messages/. Accessed Apr 6, 2025

[3] G. R. Gunnam, D. Inupakutika, R. Mundlamuri, S. Kaghyan and D. Akopian, "Assessing Performance of Cloud-Based Heterogeneous Chatbot Systems and A Case Study," in IEEE Access, vol. 12, pp. 81631-81645, 2024

[4] Reddy, Ganesh & Inupakutika, Devasena & Mundlamuri, Rahul & Kaghyan, Sahak & Akopian, David. (2024). Hybrid Machine Learning Approach for Task-Oriented Dialog Systems. International Journal of Computer Applications. 186. 35-42. 10.5120/ijca2024923679.

[5] D. Inupakutika et al., "Custom Natural Language Understanding for Healthcare Chatbots and A Case Study," in 2024 IEEE International Conference on Digital Health (ICDH), Shenzhen, China, 2024, pp. 114-122

[6] P. Cañas, D. Griol, Z. Callejas, Towards versatile conversations with data-driven dialog management and its integration in commercial platforms, Journal of Computational Science, Volume 55, 2021,

[7] M. Nadim, D. Akopian and A. Matamoros, "A Comparative Assessment of Unsupervised Keyword Extraction Tools," in IEEE Access, vol. 11, pp. 144778-144798, 2023

[8] P. Chalela, AL McAlister, D. Akopian, E. Munoz, C. Despres, S. Kaghyan, AG Ramirez, “Facebook Chat Application to Prompt and Assist Smoking Cessation Among Spanish-Speaking Young Adults in South Texas. Health Promotion Practice, 2022 May;23(3):378-381.

[9] Serban, Iulian & Lowe, Ryan & Charlin, Laurent & Pineau, Joelle. (2015). A Survey of Available Corpora for Building Data-Driven Dialogue Systems. Dialogue and Discourse. 9. 10.5087/dad.2018.101.

[10] J. Gao, M. Galley, and L. Li, “Neural Approaches to Conversational AI. Found.” Trends Inf. Retr. 13, 2–3 (Feb 2019), pp. 127–298

[11] H. Rodríguez Reséndiz and J. Rodríguez Reséndiz, "Digital resurrection: challenging the boundary between life and death with artificial intelligence", Philosophies, vol. 9, no. 3, p. 71, 2024.

[12] H. Wang, B. Guo, W. Wu, S. Liu, Z. Yu, “Towards information-rich, logical dialogue systems with knowledge-enhanced neural models,” Neurocomputing, Volume 465, 2021, Pages 248-264

[13] E. Adamopoulou, L. Moussiades, “An Overview of Chatbot Technology.” In: Maglogiannis, I., Iliadis, L., Pimenidis, E. (eds) Artificial Intelligence Applications and Innovations. AIAI 2020. IFIP Advances in Information and

Communication Technology, vol 584. Springer

[14] F. Aslam, “The Impact of Artificial Intelligence on Chatbot Technology: A Study on the Current Advancements and Leading Innovations.” European Journal of Technology. 7. 62-72. 2023.

[15] A. Babu, S. Babu Boddu. “BERT-Based Medical Chatbot: Enhancing Healthcare Communication through Natural Language Understanding, Exploratory Research in Clinical and Social Pharmacy,” V. 13, 2024

[16] A. Barros, R. Sindhgatta, and A. Nili. “Scaling up chatbots for corporate service delivery systems” Commun. ACM 64, 8 (August 2021), 88–97.

[17] Kumari, Vijay & Sharma, Abhishek & Sharma, Yashvardhan & Goel, Lavika. (2023). Scalability and Sustainability in Chatbot and Mobile Application Development. 397-403. 10.1109/Confluence56041.2023.10048882.

[18] A.M. Dai and Quoc V. Le. "Semi-supervised sequence learning." *Advances in neural information processing systems* 28 (2015).

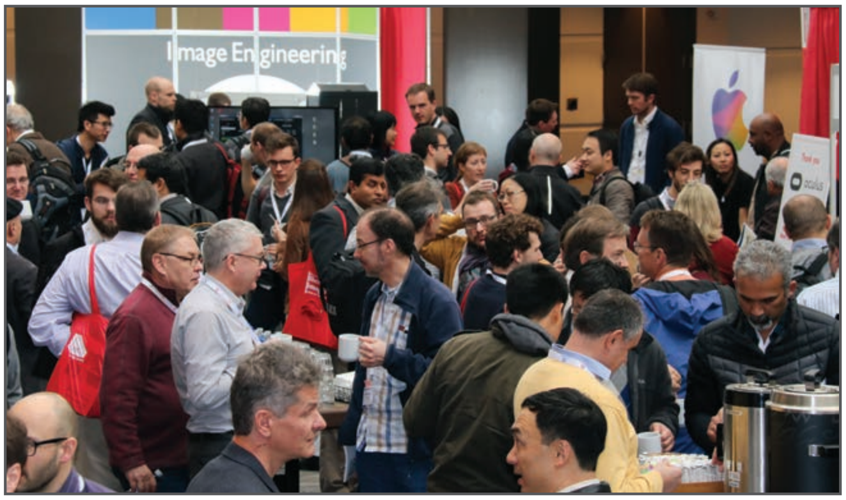
[19] I. Olawunmi, “Safeguarding health data in digital era: a comparative study of the GDPR and HIPAA 19.

[20] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gašić, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young, “A network-based end-to-end trainable task-oriented dialogue system,”.In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449. Association for Computational Linguistics, 2

JOIN US AT THE NEXT EI!

electronic IMAGING

Imaging across applications . . . Where industry and academia meet!



• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org