United States Patent Application Publication	20250259021
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	Kelkar; Amol et al.

CONTEXT-AWARE ACTION EXECUTION SYSTEM FOR CONVERSATIONAL AI

Abstract

Techniques for executing system actions during conversations between a human user and an autonomous conversational system are disclosed. A first generative language model processes user messages to determine user intent, while a dialog management model analyzes the intent and conversation context to identify required system actions. The system executes actions by retrieving parameters from context, performing database queries or API calls to obtain response data, and storing results in conversation context variables. A second generative language model generates natural language responses using the action results. The system maintains conversation context including message history, action results, and state information, validates action execution, and initiates human agent handoff when needed. The system improves over time by detecting poor performance, gathering problematic conversations, and retraining using updated configurations.

Inventors: **Kelkar; Amol (Redmond, CA), Varghese; Nikhil (San Francisco, CA), Khatri; Chandra (San Francisco, CA), Mittal; Utkarsh (Foster City, CA), Rajpurohit; Nachiketa (San Mateo, CA), Relan; Peter (Atherton, CA), Tran; Hung (Foster City, CA)**

Applicant: **GICRM AI LLC.** (Palo Alto, CA)

Family ID: **1000008563547**

Appl. No.: **19/193429**

Filed: **April 29, 2025**

Related U.S. Application Data

parent US continuation 19171764 20250407 PENDING child US 19193429
parent US continuation 17570281 20220106 parent-grant-document US 12282743 child US 19171764
us-provisional-application US 63135840 20210111

Publication Classification

Int. Cl.: **G06F40/49** (20200101); **G06F40/247** (20200101); **G06F40/289** (20200101); **H04M3/527** (20060101)

U.S. Cl.:

CPC **G06F40/49** (20200101); **G06F40/247** (20200101); **G06F40/289** (20200101); **H04M3/527** (20130101);

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS [0001] This application is a continuation of U.S. patent application Ser. No. 19/171,764, titled “TECHNIQUES FOR AUTOMATED DISCOVERY OF SYSTEM ACTIONS FOR CONVERSATIONAL AI, FROM CONVERSATION LOGS,” filed on Apr. 7, 2025, which is a continuation of U.S. patent application Ser. No. 17/570,281, titled “AUTONOMOUS CONVERSATIONAL AI SYSTEM WITHOUT ANY CONFIGURATION BY A HUMAN,” filed on Jan. 6, 2022, which claims the benefit under 35 U.S.C. § 119 (e) of U.S. Provisional Patent Application No. 63/135,840, titled “AN AUTONOMOUS CONVERSATIONAL AI SYSTEM WITHOUT ANY CONFIGURATION BY A HUMAN,” filed on Jan. 11, 2021. Priority is claimed under 35 U.S.C. § 120 to each of the above-referenced non-provisional applications. The disclosures of U.S. patent application Ser. Nos. 19/171,764 and 17/570,281, and U.S. Provisional Patent Application No. 63/135,840 are hereby incorporated by reference in their entireties.

FIELD OF THE INVENTION
[0002] The present invention relates generally to the field of Conversational AI systems that handle user questions or conversations with an entity, which in one embodiment is a business, using deep neural networks based AI models to learn to converse with users in a very similar fashion that a human would. In one embodiment the human could be a call center agent or simply, an agent, and the user could be a customer of a business.
[0003] The present invention allows an entity to create an autonomous Conversational AI system (ACAi) by letting a software system inspect past conversations available in email exchanges or chatbots or voice call logs. The entity provides historical conversations and the system and method of the invention trains an AI system that can mimic, in one embodiment, the way human agents in the business have interacted with users in the past, thus creating an ACAI system through this system and method. This process requires no human involvement to configure the ACAI system by the entity user. Finally, the ACAI system is also capable of improving its own accuracy over time by automatically updating its configuration once it is in use.
BACKGROUND
[0004] Natural Language Processing or NLP using Deep Neural Networks (DNNs) has become an active field of research and commercial work in recent years. DNNs in Natural Language Processing are AI models that are capable of probabilistic pattern matching to see if a natural language utterance resembles a previously seen utterance they have been exposed to during a training phase, either exactly or similarly phrased, and produce an output or prediction of what to do or how to respond, based on this input utterance pattern match. Many types of applications of NLP models are being developed: including sentiment analysis of social media posts (negative, positive, neutral), summarization of large bodies of text, extraction of named entities from text (getting names and addresses out of a written mortgage contract) etc.

[0005] Human agents have been servicing users and customers of a business or an organization for a long time. Whether in sales, where a sales agent can help a customer select a product they want to buy from a business, or in customer service where a service agent can help a customer address an issue like refunding their payment if they are not happy with a purchase.
[0006] Even inside a business, a user may use a service offered by the business to its employees, such as an HR department acting as the entity, in which human resources agents can service a request the user, an employee, may have: such as how much vacation balance they have left.
[0007] With artificial intelligence or AI and NLP, it is now possible to offload some of the requests users may have of human agents to artificial intelligence based Conversational AI systems. This field is generally known as Conversational AI, and the problem is creating a Conversational AI system that can “converse” with a user or employee and address their sales, service or other questions and requests while making the user feel like they are talking to a human being.
[0008] Conversational AI (CAI) systems typically consist of three major components: Natural Language Understanding (NLU), Dialog Manager (DM) and Natural Language Generation (NLG). There is also an important piece of technology tied to the DM, which is called a Fulfillment or an Action Server. FIG. **3** showcases one embodiment of the proposed system containing these components.
[0009] NLU or Natural Language Understanding, involves understanding what the user is communicating, and NLG or Natural Language Generation which involves creating an utterance that would likely be what a human would say, transmitting it back to the user. Dialog Manager is the component that keeps track of the overall progress of the conversation, since it can involve multiple back and forth turns, history of what the user has said and the NLU has understood, and what the NLG has responded with, and uses the history and state of the conversation as context for enabling the Conversational AI system to taking certain actions if needed, such as retrieving information from a database or an API. For example purposes only, the user may utter: “My account number is 123456.” and “What is my balance?” and the NLU would understand these first two user turns and their intents, the Dialog Manager would use the account number to request an action be fired off to retrieve account details from a database, using the account number 123456, and store the results in context variables. The NLG would respond “Let me pull up your account details”. And finally the NLG would state: “Your account balance is $250.00”.
[0010] To further understand each component, let's start with the NLU. The NLU component deals with understanding the messages sent by the user. One approach is to identify intents and slots in the user's message. An intent refers to a particular user intention that may be conveyed by a variety of different utterances. For instance, a cancel subscription intent may be indicated by utterances like “I want to cancel”, “Please cancel my account”, “I don't want my subscription anymore”. A slot or entity is a value extracted from a user utterance. These entities could have values such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc. For instance, a user utterance “I want a refund of $10 by 6 pm” has two entities: monetary value of $10 and a time expression of 6 pm. Intent detection and entity extraction are standard tasks in NLP literature, typically performed using DNNs.
[0011] Once the NLU pipeline identifies user intent and any entities from a user utterance, the DM identifies any system actions that should be executed by an Action Server, and a response that will be sent to the user, along with. In some cases, the DM predicts a response template from a set of predefined response templates. The template may be of the form “You will be refunded $number by $date-time”. The NLG module uses the predicted template and other contextual information, such as the amount and the day and time to construct the final response message. In other cases, the NLG module generates a text message using contextual information including but not limited to the intent and slot values detected by the NLU module and dialog state.
[0012] Recently, Conversational AI systems have been implemented to handle many use cases. In the field of voice and telephony, as one example, systems have been built where the user can say a

phrase into the phone, and the system recognizes the phrase and responds appropriately. In the field of chatbots, as another example, a user may go to a website or a smartphone and chat through a chatbot going back and forth with the Conversational AI system on the other side of the chatbot.
[0013] The user experience offered by such systems is only as good as the sophistication of the technology in the Conversational AI system and, very importantly, the configuration information used to configure the system for a particular use case in an entity. Existing Conversational AI systems require substantial initial and ongoing human effort towards identifying and compiling the configuration information. Many tasks are involved in this effort. For example, a human needs to identify the conversation topics that the Conversational AI system should handle, collect examples of such conversations, identify various intents that the user may utter and specify various ways that intent may be uttered, and so on. These are complex tasks that require specialized human skills. As the Conversational AI system goes into live operation, deficiencies in coverage of user request types, and understanding need to be identified and incorporated into the configuration and the Conversational AI system needs to be re-configured constantly. These configuration changes are done manually with the help of some partial automation tools. Prior art exists for partially generating and suggesting topic, intent and slot values, though not for automatically generating configuration for the complete Conversational AI system.
[0014] The present invention is the first invention that requires no human involvement in the full creation, configuration and improvement processes of an autonomous Conversational AI (ACAI) system that can service users and their conversational requests in natural language for an entity. One key idea in the invention is that most entities have logs or records of past conversations that have already occurred between users and human agents, and an AI based system that uses a discovery based approach to inspect these logs and bootstrap itself into an ACAI system by learning to mimic what real human agents have communicated and done in the past when interacting with users.
[0015] Also enabled, without involving a human in the entity to manually configure anything, is the ACAI system's ability to get better and better in coverage of user request types, accuracy of its language understanding, and its generated actions and responses over time. In one embodiment, each time the ACAI system makes a mistake, and the user says “connect me to a human agent”, or “you are not understanding”, the ACAI system realizes it made a mistake and over time will try other ways of mimicking what human agents have done in the past, by re-learning from conversation logs. And when it does not make a mistake, the conversation patterns that are working with the user are reinforced in the ACAI system.

Description

BRIEF DESCRIPTION OF THE DRAWINGS
[0016] FIG. **1** is an overview block diagram of the major components and personas of the Conversational AI system and method for supporting user (**111**) conversations with an entity using an ACAI System (**103**) instead of a human agent (**112**), in which the ACAI System (**103**) is created entirely by using the Automatic Conversational AI Configuration (**101**).
[0017] FIG. **2** is an overview block diagram of the major components of the proposed system and method in which the Automatic Conversational AI Configuration is created automatically by inspecting past conversations between users (**111**) and human agents (**112**) (shown in FIG. **1**) that is generating Automatic Automatic Conversational AI Configuration (**101**) as output which is used in training the ACAI System (**103**).
[0018] FIG. **3** is the representation of the ACAI System (**103**) being used by a User (**111**) over the internet.
[0019] FIG. **4** represents the Deep Neural Network based Generative Models (Generative DNN

Models **205**-*a*, **206**-*a*, **208**-*a*) which generates Auto-Annotations for a given Input Text
[0020] FIG. **5** shows a sample PII-removed conversation between a User and a Human Agent found in the conversation logs.
[0021] FIG. **6** shows a sample PII-removed, annotated conversation between a User and the Agent where the User begins the conversation by wanting to cancel their subscription and rejects the Agent's counter-offer, prompting the Agent to cancel their subscription. The figure shows the turn-level auto-intent, turn-level auto-response and Sentence-level auto-intent predicted at each turn of the conversation. N/A indicates that no annotation is necessary in that turn.
[0022] FIG. **7** shows a sample PII-removed conversation between a User and the Agent indicating the actor, turn and annotations made by the Generative DNN models (**205**-*a*), (**206**-*a*), and (**208**-*a*) at each turn.
[0023] FIG. **8** shows a sample PII-removed conversation between a User and the Agent indicating the actor's turn and the ACAI System (**103**) predictions at each turn.
SUMMARY
[0024] In one embodiment, our invention pertains to the observation that using conversational logs, it is possible to bootstrap an entire ACAI system (**103**) using a discovery process without any configuration involvement from a human in the entity which has these conversational logs. This ACAI system is capable of interacting with humans by mimicking the conversations observed in the conversation logs (**201**), without any humans configuring it on how to do so. The ACAI system also automatically improves over time by incorporating information acquired while it is being used.
[0025] Conversation logs include, but are not limited to, the transcript of the conversation, voice recordings, and meta data pertaining to the conversation with information such as the timestamp when each message was sent, the details of the sender and receiver of the message, the channel through which the message was sent. In one embodiment of the present system, conversation logs are retrieved from a well-known location. This data includes but is not limited to multimodal conversational logs like chat logs, emails, voice recordings of conversations across multiple languages and omni-channel streams.
[0026] The invention, in one embodiment, preserves the data privacy requirements of the entity, if the entity wishes to not expose personally identifiable information (PII). This data is absorbed by the PII Anonymizer (**202**). In this embodiment, it is an existing service which automatically replaces the user's Personally Identifiable Information (PII) with dummy values. See FIG. **5** for sample conversations which identify each of these types of annotations.
[0027] Once the conversation logs are uploaded, the log preprocessing module normalizes them into the invention's normalized canonical representation.
[0028] The normalized conversation logs go through a series of annotation steps to enrich the conversation logs with NLU annotations including but not limited to turn-level auto-intent, turn-level auto-response, auto-topic, auto-subtopic and sentence-level auto-intent.
[0029] The NLU annotations are executed by a Generative DNN model (FIG. **4**). The Generative DNN model is a language model described in the NLP literature in many applications. Some applications where a language model is used to generate text include summarization of text, story telling, programming code generation, etc.
[0030] In one embodiment of the present invention, the Annotation module for Turn-level Auto-Intent and Auto-Response uses a Generative DNN model to annotate the user utterances with a turn-level auto-intent. This turn-level auto-intent represents the main purpose of the user's utterance.
[0031] FIG. **6** lists a sample PII-removed, annotated conversation between a User and the Agent where the User begins the conversation by wanting to cancel their subscription to an entity's services and rejects the Agent's counter-offer for lowering the price of the subscription, prompting the Agent to finally cancel their subscription. The figure shows the turn-level auto-intent, turn-level auto-response and sentence-level auto-intent annotations predicted by the Generative DNN model

at each turn of the conversation. N/A indicates that no annotation is necessary in that turn. The topic of this conversation is “User wanting to cancel their subscription” and the subtopic of the conversation is “Agent cancels the subscription”.

[0032] In FIG. **6**, the user shows the turn-level auto-intent of “cancel subscription” while saying “I want to cancel my plan”. Similarly, in turn 3 sentence **1**, the user says “I just dont need it anymore” where the tum-level auto-intent is “cancel reason” because the user is explaining the reason why they don't need the subscription. It is important to note that the Generative DNN model is capable of annotating this user utterance with the intent “cancel reason” even though it has never seen the “cancel reason” intent or the user utterance, or even “I just dont need it anymore” in the training data when the Generative Model was trained.

[0033] The Annotation module for Conversation Level Auto-Topics and Auto-Subtopics (**206**) (FIG. **2**) uses a Generative DNN model (**206**-*a*) to annotate each conversation with an auto-topic and auto-subtopic. The topic is the general theme of the conversation, and the subtopic is the primary task that was accomplished or the primary outcome of the conversation. For example, in FIG. **6**, the conversation is of the topic User wanting to cancel their subscription and within that particular topic, since the user's account was canceled by the human agent, it is of subtopic Agent cancels the subscription. As opposed to the primary topic being User wanting to cancel their subscription and the subtopic being User accepts a lower price offer”

[0034] The Generative DNN model (**206**-*a*) is trained on a dataset from entities with conversations already having similar topic and subtopic annotations. The Generative DNN model (**206**-*a*) produces auto-topics and auto-subtopics for each conversation seen in the conversation logs.

[0035] The conversation logs obtained after annotation from the Generative DNN model (**206**-*a*) are used by the Conversation Ranker module (**207**) to discover conversation flows that maximize coverage of the auto-topics and auto-subtopics seen in the conversation logs. In this context, coverage refers to the measure of the proportion of unique User-Agent conversation flows identified by the system to the unique User-Agent conversation flows found in the conversation logs. A conversation flow represents a User-Agent conversation that leads to the same outcome. For example, we regard all the conversations where the User requested a cancellation and rejected the Agent's counter-offers as one unique conversation flow. There can be duplicate flows discovered from the logs, which do not add much value since we identify unique flows.

[0036] Furthermore, the Generative DNN (**206**-*a*) discovers most auto-topics and auto-subtopics from the logs and it is good enough to be deployed, however, it is possible that some topics and subtopics are missed and will result in the user requiring a hand-off to a human agent. This can be addressed in the automatic retraining stage. As new flows are discovered with new historical conversation logs, the Generative DNN (**206**-*a*) discovers more auto-topics and auto-subtopics thereby increasing the coverage with time and eventually leading to convergence.

[0037] In one embodiment of the present invention, since Users are unrestricted in what they can say to an agent, the present invention focuses on identifying the most frequent conversations found in the conversational logs, between the User and the Agent. To maximize coverage of topics and subtopics the ACAI system will handle, while using a small number of initial conversations, a Conversation Ranker module identifies the top k most frequent auto-topic and auto-subtopic pairs, and selects the n most representative conversations for each auto-topic and auto-subtopic pair. It does so by calculating a normalized mean conversation embedding of all the conversations in each auto-topic and auto-subtopic pair, and choosing the n conversations closest to the normalized mean conversation embedding. A conversation embedding is a vector representation of the conversation in a multi-dimensional space with the property that similar conversations will have conversation embeddings that are closer to each other in the multi-dimensional space and conversely, dissimilar conversations will have conversation embeddings that are farther apart in the multi-dimensional space.

[0038] An Annotation module for Sentence-Level Auto-Intents (**208**) is also utilized in one

embodiment of this invention. This module uses another Generative DNN model to annotate the segmented user utterances from the selected conversations with a sentence-level auto-intent. A sentence-level auto-intent is identified for each sentence in the user utterance. For example, in FIG. **6**, the user replies with two sentences in turn 3 “Yes.” and “I just dont need it anymore”. The user says “Yes” in the first sentence and is thus annotated with an affirm sentence-level auto-intent. The second sentence is annotated with a cancel reason sentence-level auto-intent because the user explains why they don't need the subscription by saying “I just dont need it anymore”. This process is identical to the first Generative DNN model with the difference being that the third Generative DNN model is trained on a dataset consisting of sentence-level auto-intents.

[0039] Once all the conversations are annotated, in one embodiment of the present invention, the AutoFlows Generator (**209**) generates the conversation flows per auto-topic and auto-subtopic in a graph where the root node denotes the start of all conversation flows having the same auto-topic and auto-subtopic. A conversation flow is a sequence of sentence-level auto-intents and turn-level auto-responses.

[0040] In one embodiment of the present invention, the conversation flows having nodes with frequency higher than a certain frequency threshold are included as stories in the Automatic Conversational AI Configuration. A story is defined as any linear subset of the graph and represents a specific manifestation of a unique conversation flow. For example, the conversation in FIG. **6** has the story: {cancel subscription, welcome message, ask cancel reason, affirm, cancel reason, offer plan, cancel subscription, cancellation confirmation, closing question}.

[0041] One of the last steps in the present invention before the ACAI system trains itself for deployment for live use, is configuring itself for fulfillment actions which are usually served via APIs connected to an entity's software infrastructure, and contextual slots and responses. A slot is a set of values extracted from a user utterance, or returned from a fulfillment action.

[0042] Consider a sample PII-removed conversation between a User and the Agent indicating the actor, turn and annotations at each turn in FIG. **7**. Model describes which model is responsible for prediction. Annotation Type indicates the type of annotation used for training the model and Annotation indicates the actual value. The User begins the conversation by wanting to cancel their subscription and rejects the Agent's counter-offer, prompting the Agent to actually cancel their subscription.

[0043] In the course of a conversation, an ACAI system needs to perform certain fulfillment actions. For example, loading a given user's account information from a database, or making an API call to get current weather information. In one embodiment of the present invention, we automate the task of identifying system actions, along with action implementations, without configuration help from a human in the entity. The objective is to identify where in a conversation actions need to be performed, which actions need to be performed and with what attributes. The Action Annotation module (**210**) uses both user messages and agent responses. A natural-language-to-SQL model is used to convert user messages such as “How much is due?” in an SQL query or an API call based on available schema information. Similarly, agent responses such as “Let me load your account information” are also translated into SQL or API action.

[0044] Turn 3 in FIG. **7** shows that the fulfillment action sql_load_account_information returns the name of the user. This is the slot that is used to build the response template “Hi $name, may I know the reason?”.

[0045] The present invention uses the construct of conversation context variables in addition to slots. The conversation context variables are used to keep track of the conversation state and previously mentioned slots.

[0046] In one embodiment of the present invention, the conversation context variables are discovered using a DNN that is trained on an existing dataset with <conversation state, conversation context variable> for every turn in the conversation and thus learns to predict the context variable at every turn.

[0047] In one embodiment of the present invention, the ACAI system configuration has now been created using the Configuration Generator (**211**) and it contains stories annotated with intents, responses, fulfillment actions and slots. The ACAI system then self-trains itself using this configuration and yields an NLU model, a DM model, an NLG model and a configured Action Server.

[0048] In one embodiment of the present invention, the ACAI system configuration can be exported to a different non-autonomous Conversational AI system built by a third party. Such a Conversational AI system could include but is not limited to Google DialogFlow CX or Twilio Auto-pilot, and such third party Conversational AI systems may have their own NLU model, DM model, NLG model and Action Server. Thus, the invention can either bootstrap its own native ACAI system or bootstrap a third party non-autonomous Conversational AI system, which was never intended to be autonomous, but this invention makes the third party system autonomously bootstrapped without configuration by a human. This bootstrapped third party Conversational AI system can then be trained and deployed for live operations.

[0049] In one embodiment of the invention, the deployed ACAI system is integrated with the Conversational AI Portal so that if ever an agent in the entity is needed to take over the conversation from the ACAI system, this Portal is where the hand-off occurs. Once an agent is involved, the ACAI system is no longer in the loop and the conversation is no longer being handled by the ACAI system.

[0050] The ACAI system is now ready to interact with the users and mimic the stories that it has learned from the conversation logs. In one embodiment of the present invention, the ACAI system can have either a successful conversation, or an unsuccessful conversation with the user. In a successful conversation, the ACAI system correctly acts and responds to the user throughout the conversation. In an unsuccessful conversation, the ACAI system interacts with the user until the user requests an agent handoff, or expresses frustration with the ACAI system. A user can request to speak to a human agent by saying “Transfer me to a human” or other linguistic variations with the same intent. Apart from the user's agent handoff request, other signals based on the user's sentiment, the user's feedback, and the number of times an intent has repeated occurrences are used to calculate a user's dissatisfaction threshold. Once the user requests an agent hand-off or the user dissatisfaction threshold is crossed, they are redirected to a human agent who takes over the conversation in the Conversational AI Portal. At this point the ACAI system is no longer in use because a human is involved.

[0051] The ACAI system will naturally make mistakes when it is first deployed for live operations in an entity. However, it is capable of learning from its mistakes and getting better and better over time until it makes very few mistakes, and it approximates the capability of a human agent. This is because of the new historical logs being created between a user and a human agent when a hand-off is done, and the conversation exits the ACAI system. A hand-off is requested because of two major reasons: (1) the ACAI system was not trained on a conversation flow pertaining to the user's request or (2) the ACAI system did not understand the user's request. Either way, the unsuccessful conversations now have become part of the conversational logs in the entity. The conversational log discovery process of the ACAI system can be repeated so that the ACAI system learns the new flow and/or understands the user request better by learning more paraphrases and intents for that request. To improve over time, the ACAI system periodically repeats the discovery process using the conversations between the user where the user requested an agent handoff and the agent has handled the user request, because these very conversations have now become historical user agent conversations. This periodic and autonomous discovery process adds new story configurations, retrains the ACAI system and thus enables the ACAI system to continually learn and improve itself without a human involved in configuration of the ACAI system.

DETAILED DESCRIPTION

[0052] Embodiments of the present invention provide a computer-implemented system that

bootstraps an entire ACAI System (**103**) from scratch just from data (conversational logs) and AI models unlike any other existing approach where humans manually annotate, configure or design conversational flows.

[0053] Referring to FIG. **1** those elements depicted within circles represent various personas of the system that access the system via various client interfaces or devices including but not limited to voice recognition apps or devices, web browsers and mobile apps. All other elements depicted in the figure are the computer-implemented components of the system according to one embodiment of the present invention.

[0054] Referring to FIG. **2** all elements depicted in the figure are the computer-implemented components of the system according to one embodiment of the present invention.

[0055] There are three main personas that are traditionally involved in the creation and functioning of an ACAI System (**103**). In the present invention, none of the human personas are involved in the creation of an ACAI System (**103**). User (**111**) and human agents (**112**) are only involved during inference time, after the ACAI System (**103**) has already been created and deployed. Users (**111**) are involved since the primary function of the ACAI System (**103**) is to interact with them. Human agents (**112**) are involved to continue conversation where the User (**111**) requests to speak to a human or is handed off based on a computed signal. These three personas are: [0056] 1. Users (**111**): They are the end users who interact with the ACAI System (**103**) to address their issues, requests or any other reason. [0057] 2. Agents: They are responsible for conversing with the users (**111**) to address their queries and fulfill their requests. There are two types of agents, namely the ACAI System (**103**) and the human agent (**112**).

[0058] In one embodiment of the present invention, the Conversation logs (**201**) are retrieved directly from a well-known location and passed through the PII Anonymizer (**202**).

[0059] In one embodiment of the present invention, the PII Anonymizer (**202**) identifies information like name, phone number, email address, address, bank account numbers, credit card information etc, and can either replace the existing information with fake values or with placeholder tokens to indicate that a value has been replaced.

[0060] In one embodiment of the present invention, once the PII-removed Conversation Logs (**203**) are generated, they are automatically validated, scrubbed and normalized according to the invention's data format by the Logs Preprocessor (**204**). In this step, the logs are enriched with a Universally Unique Identifier (UUID) to uniquely identify each message, turn numbers to record the sequence in which the messages were sent and the actor information, whether it was sent by the user (**111**) or the human agent (**112**). A turn number is the sequence number in which each reply in the conversation is sent irrespective of the sender of the reply. The first message in the conversation is viewed as the 0th turn, the second message is viewed as the 1st turn, and so on.

[0061] In one embodiment of the present invention, Logs Preprocessor (**204**) normalizes them into the invention's canonical representation. In one embodiment of the present invention, the system handles different modes of conversational logs including but not limited to chat logs, email logs, and voice differently. In chat logs, users tend to send multiple shorter texts and expect real time responses from the human agent. However, in emails, users tend to show much more variability in the length of the message. Since the response to an email is not immediate, the users tend to include more content into a single email. While chat logs and voice recordings are similar in nature, voice recordings are also prone to Automatic Speech Recognition (ASR) hypothesis errors like repetitions and filler words. The ASR hypothesis converts a voice recording into a text format identical to the text found in chat logs. An example of filler words are the user saying “ummm i want to cancel”. Here “umm” is a filler word typically found in voice recordings. An example of repetition is “okay okay let's cancel”. Here “okay” is repeated twice.

[0062] Since emails contain a lot of irrelevant information including but not limited to greetings and salutations, in one embodiment of the present invention, the present invention uses a DNN to extract the body of the email. This DNN is trained on a large corpora where the dataset contains

tuples of the format <raw email, body span>. The body spans are indexes in the raw email which contain the body of the email. After training, the DNN is able to extract the body from the emails. Similarly, the DNN is trained with the same dataset format for voice recordings, where it learns to remove filler words and repetitions from the ASR hypothesis using multiple spans. After invoking the DNN on emails and voice recordings, we obtain the extracted email body and ASR hypothesis which is referred to as the user utterance since it is identical to the user utterance found in chat logs.

[0063] In one embodiment of the present invention, the Logs Preprocessor module (**204**) uses a model or tool like Stanza (https://stanfordnlp.github.io/stanza/tokenize.html #tokenization-and-sentence-segmentation) or NLTK′ sentence tokenizer (https://www.nltk.org/_modules/nltk/tokenize/punkt.html #PunktSentenceTokenizer) to segment the user utterance into multiple sentences. For example, the user utterance “Hello there. I want to cancel my subscription. I don't really need it anymore.” is segmented into three sentences “Hello there.”, “I want to cancel my subscription.” and “I don't really need it anymore.”. Segmentation normalizes the conversation logs across multiple streams such that the conversation logs from all the modes are now framed as an identical problem in the invention's canonical representation.

[0064] In the process of discovery, a key step is to discover user intent and system response at each turn. To do this, in one embodiment, we use a Generative DNN model (**205**-*a*) fine-tuned on an existing dataset (not provided by a human in the entity) which contains data from a similar domain. Fine-tuning the model is a separate process and occurs before the discovery process begins. Unlike Discriminative DNN models that classify text into intents from a pre-decided, fixed sample space, the Generative DNN model (**205**-*a*) learns the distribution of intents in a particular domain, and generates intent labels for the utterances. Thus can be generalized across conversation logs (**201**) from a similar domain.

[0065] The invention leverages a particular domain's pre-built datasets to finetune the Generative DNN models (**205**-*a*), (**206**-*a*), and (**208**-*a*) before the discovery process. For example, entities in the Software-as-a-Service (SaaS) business domain typically have customers who pay on a monthly subscription basis. When this entity services its customers (user (**111**), their human agents frequently encounter intents like cancel subscription, refund request, update subscription, etc when interacting with the user (**111**). Furthermore, User (**111**) will typically use linguistic phrases like “I want to cancel”, “Please cancel my account”, “I don't want my subscription anymore”, to indicate an intent to cancel subscription. In this case, it will be fine-tuned on a dataset like {<“I want to cancel”, “cancel subscription”>, <“Please cancel my account”, “cancel subscription”>, <“I don't want my subscription anymore”, “cancel subscription”>, . . . }. These datasets are created by human experts that are skilled in the art for a variety of domains for all the annotation tasks described in the invention.

[0066] Since the fine-tuned Generative DNN models (**205**-*a*), (**206**-*a*), and (**208**-*a*) have already learnt to mimic the annotations made by human experts in the business domain of the entity, they accurately annotate the Conversation logs (**201**) as well.

[0067] In FIG. **6**, we see that the Generative DNN model (**206**-*a*) annotated the message in each turn of the conversation with a turn-level auto-intent and a turn-level auto-response (**205**). For example, turn 0 and sentence 0 shows a user message with the cancel subscription turn-level auto-intent.

[0068] In one embodiment of the present invention, each conversation is annotated with an auto-topic and an auto-subtopic. The turn-level auto-intent and auto-response annotations obtained from the previous step are used as the input to annotate the conversation-level auto-topics and auto-subtopics (**206**).

[0069] Similar to the previous annotation step, in one embodiment of the present invention, a Generative DNN model (**206**-*a*) fine-tuned on an existing dataset. The dataset contains data from a similar domain, is used to annotate auto-topic and auto-subtopic for each conversation, and is not provided by a human in the entity.

[0070] In one embodiment of the present invention, the Conversation Ranker (**207**) selects the n most representative conversations in each auto-topic and auto-subtopic pair for the top-k most frequent auto-topic and auto-subtopic pairs for maximum coverage. In this context, coverage refers to the measure of the proportion of unique user-human agent conversations identified by the system to the unique user-human agent conversations found in the conversation logs (**201**).
[0071] For example, FIG. **8** lists a sample conversation between a User and the Agent where the User begins the conversation by thinking about canceling their subscription but ends up accepting the Agent's counter-offer. The topic of this conversation is" User wanting to cancel their subscription" and the auto-subtopic of the conversation is "User accepts the offer". The Conversation Ranker (**207**) finds the most representative conversations within the same auto-topic and auto-subtopic.
[0072] To maximize coverage while using a small number of initial conversations, the Conversation Ranker (**207**) identifies the top k most frequent auto-topic and auto-subtopic pairs and selects the n most representative conversations for each auto-topic and auto-subtopic pair. It does so by calculating a normalized mean conversation embedding of all the conversations in each auto-topic and auto-subtopic pair, and choosing the n conversations closest to the normalized mean conversation embedding. A conversation embedding is a vector that represents the conversation in a multi-dimensional space. This technique is used in many NLP applications like sentiment analysis, machine translation, document summarization to represent words, sentences and any other text into a multi-dimensional space.
[0073] In one embodiment of the present invention, Sentence-level Auto-Intent annotation (**208**) enables the ACAI System (**103**) to respond to longer messages, typically found in emails. Since each sentence in these longer messages can have a different intent, we introduce the construct of a sentence-level auto-intent. In the Sentence-level Auto-Intent annotation (**208**), each of the split sentences from user utterance are annotated with sentence-level auto-intent using the Generative DNN model (**208**-*a*). Note that the human agent's (**112**) responses are not segmented into a list of constituent sentences.
[0074] In FIG. **6**, the user (**111**) replies with two sentences in turn 3. The user (**111**) says "Yes" in the first sentence and is thus annotated with an affirm sentence-level auto-intent. The second sentence is annotated with a cancel reason sentence-level auto-intent because the user (**111**) explains why they don't need the subscription by saying "I just dont need it anymore".
[0075] Similar to the turn-level auto-intent and auto-response annotation (**205**), in one embodiment of the present invention, a Generative DNN model (**208**-*a*) fine-tuned on an existing dataset (Not provided by a human in the entity) which contains data from a similar domain, is used to annotate the sentence-level auto-intent for each conversation.
[0076] In one embodiment of the present invention, the AutoFlows Generator (**209**) converts the conversations into a graph of sentence-level auto-intents and tum-level auto-responses. In this graph, a flow is defined as any subset of the graph. Similarly, a story is defined as any linear subset of the graph and represents a specific manifestation of a unique conversation flow. For example, the conversation in FIG. **6** has the story: {cancel subscription, welcome message, ask cancel reason, affirm, cancel reason, offer plan, cancel subscription, cancellation confirmation, closing question}. There is a key difference between stories and flows. While stories and flows both represent conversations, a story cannot have branching conditions because it is always linear whereas flows can have branching conditions according to business logic or other conditions.
[0077] After all the conversations have been converted into flows, the present invention uses a multi-sequence subsequence alignment algorithm to simplify the flows. Multi-sequence subsequence alignment is a set of techniques used in bioinformatics to detect alignment in genetic code and for other related use cases. In one embodiment of the present invention, we use this technique to align matching parts across the conversation flows. The flows in each auto-topic and auto-subtopic pair having user intents and system responses above a certain frequency threshold are

selected. The higher the frequency threshold, the simpler the flows.
[0078] If the topic User wanting to cancel their subscription and subtopic Agent cancels the subscription are selected, then all the conversations will become a part of Conversational AI System's (**103**) initial configuration thus, equipping the ACAI System (**103**) to converse with users (**111**) in situations where they express a desire to cancel their subscriptions. Note that while using more conversations might result in higher coverage, it also significantly increases the complexity of the flows used to configure the ACAI System (**103**) and thus, makes the ACAI System (**103**) susceptible to mimic complex flows rather than simple ones.
[0079] During the course of a conversation, an ACAI System (**103**) needs to perform certain actions. For example, loading a given user's account information from a database, or making an API call to get current weather information. In one embodiment of the present invention, we automate the task of identifying system actions, along with action implementations. The objective is to identify where in a conversation actions need to be performed, which actions need to be performed and with what attributes. The Action Annotation module (**210**) uses both user messages and agent responses. A “Natural Language to SQL” model is used to convert user messages such as “How much is my account due?” in an SQL query or an API call based on available schema information. Similarly, agent responses such as “Let me load your account information” are also translated into SQL or API action. Using a pre-trained slot-extractor model and tagging the words into slots, some of which can be the output of API or database action (requestable slots) while other words can be the placeholder for making the response more sensible. E.g. in the utterance “how much of my <amount due>”, “amount due” becomes requestable slots while other words are placeholders for grammar purposes. The present embodiment uses a state of the art “Natural Language to SQL” system called BERTrand-DR to generate SQL for database actions to obtain actionable requestable slots or select an API from the provided list of APIs.
[0080] After these steps, the Automatic Conversational AI Configuration (**101**) is created which contains stories annotated with intents, responses, fulfillment actions and slots. In one embodiment of the present invention, the stories annotated by the Action Annotation (**210**) are part of the initial Automatic Conversational AI Configuration (**101**). All the annotations from the previous steps associated with the selected stories are stored in the Automatic Conversational AI Configuration (**101**). Every discovered intent and response is pre-populated with paraphrases derived from the conversations after the Sentence-level Auto-intent annotation step (**208**). All the user utterances having the same sentence-level auto-intent are paraphrases of each other. Automatic Conversational AI Configuration (**101**) can be exported to existing non-autonomous Conversational AI system frameworks including but not limited to RASA [Reference **1**, Reference **2**], DialogFlow CX [Reference **2**], Microsoft Luis [Reference **2**].
[0081] In one embodiment of the present invention, after the Automatic Conversational AI Configuration (**101**) is created, the training of the ACAI System (**103**) is triggered.
[0082] Still referring to FIG. **1**, the ACAI System (**103**) uses the Automatic Conversational AI Configuration (**101**) containing stories annotated with intents, responses, fulfillment actions and entities to train and deploy the Natural Language Understanding (NLU) model (**104**), Dialog Management (DM) model (**105**), Natural Language Generation (NLG) model (**106**) and Action Server (**107**). The NLU model (**104**) is responsible for understanding the user's utterance. The DM model (**105**) is responsible for predicting the unique response template and/or actions to every user message. The NLG model (**106**) is responsible for generating the final response sent to the user (**111**). The Action Server (**107**) executes the predicted fulfillment actions on Third Party Systems and Databases (**108**). The NLU model (**104**) is a neural network that is trained on a dataset with tuples <user utterance, sentence-level auto-intent. Thus, when the ACAI System (**103**) receives the incoming messages, it is able to predict the sentence-level auto-intent for every message.
[0083] The DM model (**105**) is a neural network trained on a dataset<{user utterance, NLU predictions, conversation history}, {sequence of responses and/or actions}>. While training, the

DM model (**105**) takes the tuple containing a set of information: user utterance, NLU predictions like intent, entity, etc. and conversation history like the previous n user utterances and system responses. This set has a corresponding label: a sequence of responses and/or actions. Thus, once training is complete, the DM model (**105**) can predict a sequence of responses and/or actions for a user utterance.

[0084] In one embodiment, the NLG model (**106**) generates the response by filling the slot values in the response template predicted by the DM model (**105**). In another embodiment, it is a Generative DNN model that generates the response using the user utterance, conversation context variables, result of the NLU model (**104**) and the result of the DM model (**105**).

[0085] Still referring to FIG. **1**, the Conversational AI Portal (**109**) is a real-time component through which the agent communicates with the user. The agent connects to channel (**110**) through the Conversational AI Portal (**109**). In one embodiment of the present invention, the Conversational AI Portal (**109**) relays the message generated by the ACAI System (**103**) directly to the channels (**110**) without any interference by the human agent (**112**). The Conversational AI Portal (**109**) passes all the conversations between the agent and the user to the Conversational AI Inbox (**113**).

[0086] Once the ACAI System (**103**) is trained and deployed on the server, it is ready to interact with the users. In one embodiment of the present invention, the user (**111**) initiates a conversation with the ACAI System (**103**) on a particular channel. In one embodiment of the present invention, the channel sends the user utterance to the trained ACAI System (**103**). For instance, in conversation **1** shown in FIG. **7**, the user (**111**) initially says “Please cancel my account”. Since this is the first message in the conversation, it has the turn 0.

[0087] In one embodiment of the present invention, the user utterance is first passed through the NLU model (**104**) to obtain the intent and subsequently to the DM model (**105**) which predicts either a response or an action. If the prediction is a response template, the NLG model (**106**) fills the predicted response template with the appropriate slot information.

[0088] In one embodiment of the present invention, the fulfillment action predicted by the DM model (**105**) is executed on the Action Server (**107**). To illustrate the working of a fulfillment action, we consider the conversation in FIG. **7** which predicts a fulfillment action (sql_load_account_information) in turn 3. This is a SQL-based action and thus, is executed on a database (**108**) to retrieve the account information. This fulfillment action loads the account information of the user (**111**) and stores it in the conversation context variables.

[0089] In one embodiment of the present invention, the fulfillment action executes on the third party system (**108**) and returns the value to the Action Server (**107**).

[0090] In one embodiment of the present invention, the Action Server (**107**) sends the result of the fulfillment action to the Conversational AI Portal (**109**). The Conversational AI Portal (**109**) sets the results as entities in the conversation. These entities are used by the DM when it makes the next prediction.

[0091] In one embodiment of the present invention, after the response or action predicted by the ACAI System (**103**), it predicts another response or action until it predicts an action_listen. If the prediction is another fulfillment action, the fulfillment action is executed on the third party systems (**108**) and the ACAI System (**103**) makes another response or action prediction. If the prediction is a response, the predicted response template is filled with slot information and the ACAI System (**103**) makes another response or action prediction. Once the DM predicts an action_listen, the DM stops making any more predictions and the step comes to a halt.

[0092] In one embodiment of the present invention, the responses and actions are sent to channel (**110**) from the Conversational AI Portal (**109**).

[0093] In one embodiment of the present invention, channel (**110**) sends the responses and actions to the user (**111**).

[0094] In one embodiment of the present invention, after receiving each user's utterance, a check is made to identify if the user's dissatisfaction threshold is crossed. If the threshold is crossed, the

conversation is handed off to a human agent (**112**) immediately and the ACAI System (**103**) stops making further predictions. The user dissatisfaction threshold is computed by combining two types of signals (1) utterance-level and (2) conversation-level.

[0095] Utterance-level signals include whether the user (**111**) requested to speak to a human agent (**112**), the sentiment of the user (**111**) in each utterance. If the user (**111**) requests for a hand-off, then the conversation is immediately handed off to an available human agent (**112**) irrespective of any other signal. Additionally, the sentiment (positive, negative or neutral) of a user's utterance is a signal to the user dissatisfaction threshold.

[0096] Conversation-level signals include features such as: the sentiment of the entire conversation (positive, negative or neutral), the presence of multiple fallback intents. A fallback intent occurs when the NLU model (**104**) has low confidence in it's intent prediction. If the NLU model (**104**) predicts the fallback intent, the NLG model (**106**) will generate a response that asks the user (**111**) to rephrase their request. For example, it could say “I did not understand your request. Can you please rephrase it for me?”.

[0097] If the sentiment of the conversation is negative, it could indicate that the user (**111**) was not satisfied with the agent's responses. The presence of multiple fall back intents in a conversation suggests that the ACAI System (**103**) might not be trained to understand the user's (**111**) request. For example, if the ACAI System (**103**) was only trained on conversations under the topic User wanting to cancel their subscription, the NLU will predict a fall back intent when a user (**111**) interacts with the ACAI System (**103**) to get a new account since it does not have a request new account intent.

[0098] In one embodiment of the present invention, the Conversational AI Portal (**109**) uploads the conversations between the user (**111**) and the ACAI System (**103**) to the Conversational AI Inbox (**113**) in real time.

[0099] Over time, the ACAI System (**103**) needs to be re-trained for three major reasons: [0100] 1. To discover new auto-topics, auto-subtopics, auto-intents, fulfillment actions and flows that were absent in the previous Conversation logs (**201**) and thus, are not configured in the Automatic Conversational AI Configuration (**101**). [0101] 2. To improve the performance of DM model (**105**), and NLG model (**106**) by discovering new conversation flows within auto-topic and auto-subtopics that are already configured in the Automatic Conversational AI Configuration (**101**). [0102] 3. To improve the performance of the NLU model (**104**) by discovering new paraphrases for existing intents and response templates for existing responses.

[0103] In one embodiment, the present invention uses every conversation that was handed off to the human agent (**112**) since the last retraining. These conversations are populated in the Conversational AI Inbox (**113**). The conversations serve as the conversation logs (**201**) for the next discovery process.

[0104] The reason why these conversations are selected is because they were handed off to a human agent (**112**) midway during the conversation. This is a strong indication that the ACAI System (**103**) was unable to understand or service the user's (**111**) requests. Thus, the handed off conversations are included in the subsequent discovery process. When this process completes and the new ACAI System (**103**) is deployed, it can successfully respond to the flows, understand the user (**111**) and service their requests in cases where the previous ACAI System (**103**) failed. If the conversation is not handed off to a human agent, it means that the ACAI System (**103**) successfully serviced the user (**111**). In this case, the ACAI System (**103**) does not need to learn new flows.

[0105] In conversations where an agent hand-off was done, it is not desirable to include the entire conversation along with ACAI System (**103**) messages for the discovery process. Thus, the conversations are truncated to only include the conversation between a user (**111**) and a human agent (**112**). For example, if a conversation is handed off to an agent in turn 5, the Conversation logs (**201**) for the new discovery process only contains the conversation starting from turn 5 until the end of the conversation.

[0106] In one embodiment of the present invention, a retraining process is triggered periodically. The new Conversation logs (**201**) from the Conversational AI Inbox (**113**) are passed through the discovery process, leading to a new Automatic Conversational AI Configuration (**101**). This configuration file adds new stories to the existing configuration and then retrains the ACAI System (**103**). Now, the improved ACAI System (**103**) is ready to interact with the users (**111**).
[0107] In one embodiment, the present invention supports multilingual data and channels like chat, email and voice. It bootstraps an ACAI System (**103**) for any language (100+) by leveraging conversational logs from a single language like English. It does so by using state-of-the-art multilingual DNN that takes a user utterance from any language and generates an embedding which allows the NLU model (**104**) to understand multiple languages, and translating the response templates configured in the ACAI System (**103**) to the target language that the user wishes to communicate in.
[0108] In one embodiment, the present invention is encapsulated in a user interface that does not require any persona to write code in a programming language, and connects to the channels (**110**) and backend systems via connectors. Backend systems include but are not limited to Databases, CRM (Customer Relationship Management) softwares, Ticketing systems, other non-autonomous Conversational AI system frameworks and Knowledge Bases. Thus, the system can read from or write to backend systems that the user wants to connect to, allowing the invention to converse with a customer in real-time with access to up-to-date information and the ability to fulfill customer requests including but not limited to canceling a subscription, processing a refund, and redirecting to a concierge service.
[0109] The foregoing descriptions, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that specific details are not required in order to practice the invention. Thus, the foregoing descriptions of specific embodiments of the invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed; obviously, many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, they thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the following claims and their equivalents define the scope of the invention.

Claims

1. A computer-implemented method for executing system actions during a conversation between a human user and an autonomous conversational system, the method comprising receiving a user message from the human user via a communication channel; providing the user message as input to a first generative language model and receiving as output an indication of a user intent; determining, by a dialog management model based on the user intent and conversation context, that a system action is required; executing the system action to obtain response data, wherein executing comprises: retrieving required parameters from the conversation context, performing the system action using the required parameters, and storing results of the system action in conversation context variables; generating, by a second generative language model using the response data, a response message to the human user; and transmitting the response message to the human user via the communication channel.
2. The method of claim 1, wherein executing the system action comprises: retrieving a database query configuration from stored system actions; executing the database query on a connected database system using the required parameters to obtain the response data; validating the response data received from the database system; and storing the validated response data in the conversation

context variables.
3. The method of claim 1, wherein executing the system action comprises: retrieving an API call configuration from stored system actions; executing the API call to a third-party system using the required parameters to obtain the response data; validating the response data received from the third-party system; and storing the validated response data in the conversation context variables.
4. The method of claim 1, wherein the conversation context comprises: previously exchanged messages between the human user and the autonomous conversational system; results from previously executed system actions; extracted parameter values from the user message; and conversation state information for use by one or more of a natural language understanding model, dialog management model, and generative language model.
5. The method of claim 1, wherein generating the response message comprises: selecting a response template based on the system action results; filling placeholder parameters in the template with values from the response data; and generating a natural language response with the filled template using the second generative language model.
6. The method of claim 1, further comprising: detecting when execution of the system action fails; generating an alternative response without using the system action; storing information about the failed system action attempt; and initiating a handoff to a human agent when system action failures exceed a threshold.
7. The method of claim 1, wherein the system action comprises retrieving user account information, and wherein executing the system action comprises: accessing a database containing user account records; querying the database using user identification parameters extracted from the conversation context; and retrieving account details required for generating the response message.
8. The method of claim 1, wherein the system action comprises processing a service request, and wherein executing the system action comprises: calling a third-party API endpoint with required service parameters; receiving confirmation of the service request processing; and storing the confirmation details in the conversation context variables.
9. The method of claim 1, wherein determining that a system action is required comprises: analyzing the user intent and conversation context to identify required system actions; retrieving a stored action configuration corresponding to the identified required system actions; validating that all required parameters are available in the conversation context; and selecting between multiple available system actions based on conversation state and parameter availability.
10. The method of claim 1, wherein the conversation context variables comprise: a conversation identifier uniquely identifying the conversation; turn numbers tracking a sequence of message exchanges; system action results organized by turn number; user intent information from the first generative language model; and parameter values extracted from user messages and system action results.
11. The method of claim 1, wherein the first generative language model and the second generative language model are either: (i) a single generative language model trained to perform both user intent determination and response message generation tasks, or (ii) two separate generative language models, wherein: the first generative language model is specifically trained to determine user intent from received user messages, and the second generative language model is specifically trained to generate response messages using response data from executed system actions.
12. A system for executing system actions during a conversation between a human user and an autonomous conversational system, the system comprising: a communication interface configured to receive user messages from the human user; one or more processors; and memory storing instructions that, when executed by the one or more processors, cause the system to: receive a user message via the communication interface; provide the user message as input to a first generative language model and receive as output an indication of a user intent; determine, by a dialog management model based on the user intent and conversation context, that a system action is required; execute the system action to obtain response data by: retrieving required parameters from

the conversation context, performing the system action using the required parameters, and storing results of the system action in conversation context variables; generate, by a second generative language model using the response data, a response message; and transmit the response message via the communication interface.

13. The system of claim 12, wherein executing the system action comprises: retrieving a database query configuration from stored system actions; executing the database query on a connected database system using the required parameters to obtain the response data; validating the response data received from the database system; and storing the validated response data in the conversation context variables.

14. The system of claim 12, wherein executing the system action comprises: retrieving an API call configuration from stored system actions; executing the API call to a third-party system using the required parameters to obtain the response data; validating the response data received from the third-party system; and storing the validated response data in the conversation context variables.

15. The system of claim 12, wherein the conversation context comprises: previously exchanged messages between the human user and the autonomous conversational system; results from previously executed system actions; extracted parameter values from the user message; and conversation state information for use by one or more of a natural language understanding model, dialog management model, and generative language model.

16. The system of claim 12, wherein generating the response message comprises: selecting a response template based on the system action results; filling placeholder parameters in the template with values from the response data; and generating a natural language response with the filled template using the second generative language model.

17. The system of claim 12, wherein the instructions further cause the system to: detect when execution of the system action fails; generate an alternative response without using the system action; store information about the failed system action attempt; and initiate a handoff to a human agent when system action failures exceed a threshold.

18. The system of claim 12, wherein the system action comprises retrieving user account information, and wherein executing the system action comprises: accessing a database containing user account records; querying the database using user identification parameters extracted from the conversation context; and retrieving account details required for generating the response message.

19. The system of claim 12, wherein determining that a system action is required comprises: analyzing the user intent and conversation context to identify required system actions; retrieving a stored action configuration corresponding to the identified required system actions; validating that all required parameters are available in the conversation context; and selecting between multiple available system actions based on conversation state and parameter availability.

20. The system of claim 12, wherein the conversation context variables comprise one or more of: a conversation identifier uniquely identifying the conversation; turn numbers tracking a sequence of message exchanges; system action results organized by turn number; user intent information from the first generative language model; and parameter values extracted from user messages and system action results.