(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
04 May 2023 (04.05.2023)

WIPO | PCT

(10) International Publication Number
WO 2023/076754 A1

(51) International Patent Classification:
G06F 16/332 (2019.01) *H04L 51/02* (2022.01)
G06F 16/33 (2019.01) *G06N 3/08* (2006.01)
G06F 16/35 (2019.01)

(21) International Application Number:
PCT/US2022/074974

(22) International Filing Date:
15 August 2022 (15.08.2022)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/273,761 29 October 2021 (29.10.2021) US
17/819,445 12 August 2022 (12.08.2022) US

(71) Applicant: **ORACLE INTERNATIONAL CORPORATION** [US/US]; 500 Oracle Parkway M/S 5OP7, Redwood Shores, California 94065 (US).

(72) Inventors: **ZHONG, Xu**; c/o Oracle International Corporation, 500 Oracle Parkway M/S 5OP7, Redwood Shores, California 94065 (US). **DHARMASIRI, Yakupitiyage Don Thanuja Samodhye**; c/o Oracle International Corporation, 500 Oracle Parkway M/S 5OP7, Redwood Shores, California 94065 (US). **DUONG, Thanh Long**; c/o Oracle International Corporation, 500 Oracle Parkway M/S 5OP7, Redwood Shores, California 94065 (US). **JOHNSON, Mark Edward**; c/o Oracle International Corporation, 500 Oracle Parkway M/S 5OP7, Redwood Shores, California 94065 (US).

(74) Agent: **ASSESSOR, Brian J.** et al.; Kilpatrick Townsend & Stockton LLP, 1100 Peachtree Street NE, Suite 2800, Mailstop: IP Docketing - 22, Atlanta, Georgia 30309 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ,

(54) Title: DEEP LEARNING TECHNIQUES FOR EXTRACTION OF EMBEDDED DATA FROM DOCUMENTS

(57) Abstract: Deep learning techniques are disclosed for extraction of embedded data from documents. In an exemplary technique, a set of unstructured text data is received. One or more text groupings are generated by processing the set of unstructured text data. One or more text grouping embeddings are generated in a format for input to a machine learning model based on the one or more generated text groupings. One or more output predictions are generated by inputting the one or more text grouping embeddings into the machine learning model. Each output prediction of the one or more output predictions correspond to a predicted aspect of a text grouping of the one or more text groupings.

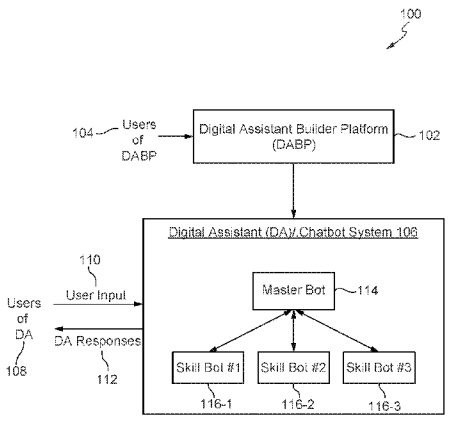


FIG. 1

[Continued on next page]

WO 2023/076754 A1

TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

DEEP LEARNING TECHNIQUES FOR EXTRACTION OF EMBEDDED DATA FROM DOCUMENTS

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of priority of U.S. Patent Application No. 17/819,445, filed on August 12, 2022, which claims the benefit of U.S. Provisional Application No. 63/273,761, filed on October 29, 2021, the specifications of each of which are incorporated herein by reference in their entirety.

FIELD

[0002] The present disclosure relates generally to chatbot systems, and more particularly, to deep learning techniques for extraction of question and answer pairs from data.

BACKGROUND

[0003] Instant messaging functionality and automated chat platforms are efficient solutions to modern customer service issues. Organizations can leverage these practices to provide timely and responsive service to their customers without committing valuable human capital to individual user inquiries. Modern automated chat platforms may utilize a "chatbot" to handle customer service requests or other interactions with humans. Some chatbots may be designed and trained to handle specific requests, such as answering inquiries posed by a human.

[0004] Training a chatbot to interact effectively with a human is a time and resource intensive task. A large volume of training data is often required during the training process, and the chatbot model being trained may be designated for a certain task for which an existing set of training data is not appropriate. Obtaining sufficient training data to train these chatbots often requires a user to manually create training data or parse a document to extract and label data in a manner that a chatbot model can interact with. This is a very time and resource intensive task for the human and delays the deployment of the chatbot. For example, to train a chatbot to answer questions posed by a human, the chatbot may be required to parse thousands of examples of sample questions and corresponding answers, that must be written or modified manually by a human.

[0005] A great number of documents are available digitally that include questions and answers. For example, a customer training a new chatbot may utilize existing documents on

the customer's website, such as frequently asked questions (FAQ) to generate a set of training data. These existing documents are often in unstructured formats, such as portable document format files (PDF files) in which a chatbot cannot be trained, as the chatbot will not accept the file as a training input. Furthermore, a human user must personally parse the unstructured documents and manually modify or regenerate the data in an appropriate embedded format compatible the chatbot before training may commence.

BRIEF SUMMARY

[0006] Deep learning techniques are disclosed for extraction of embedded data from documents.

[0007] In various embodiments, a method includes receiving, at a data processing system, a set of unstructured text data; generating, by the data processing system, one or more text groupings by processing the set of unstructured text data; generating, by the data processing system and based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model; and generating, by the data processing system, one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model, each output prediction of the one or more output predictions corresponding to a predicted aspect of a text grouping of the one or more text groupings.

[0008] In various embodiments, a system can extract embedded data from documents. The system can include one or more processors and a non-transitory computer-readable medium coupled to the one or more processors. The non-transitory computer-readable medium can store instructions executable by the one or more processors to cause the one or more processors to perform various operations. The system can receive a set of unstructured text data. The system can generate one or more text groupings by processing the set of unstructured text data. The system can generate, based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model. The system can generate one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model. Each output prediction of the one or more output predictions can correspond to a predicted aspect of a text grouping of the one or more text groupings.

[0009] In various embodiments, a non-transitory computer-readable medium can store instructions executable by one or more processors for causing the one or more processors to

perform various operations relating to extracting embedded data from documents. The operations can involve receiving a set of unstructured text data. The operations can involve generating one or more text groupings by processing the set of unstructured text data. The operations can involve generating, based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model. The operations can involve generating one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model. Each output prediction of the one or more output predictions can correspond to a predicted aspect of a text grouping of the one or more text groupings.

[0010] In some further embodiments, the set of unstructured text data is one or more portable document format (PDF) text files. In some embodiments, processing the set of unstructured text data comprises extracting, from the set of unstructured text data, one or more sets of character and generating the one or more text groupings comprises grouping the one or more sets of characters according to a relative position of each character in the set of unstructured text data.

[0011] In some embodiments, generating the one or more text grouping embeddings comprises generating a plurality of sub-embeddings based on the set of unstructured text data or the one or more text groupings and generating the one or more text groupings based on the plurality of sub-embeddings. In some further embodiments, a sub-embedding of the plurality of sub-embeddings is a text sub-embedding generated based on one or more semantic aspects of the one or more text groupings. In some further embodiments a sub-embedding of the plurality of sub-embeddings is a bounding sub-embedding generated based on one or more extracted spatial bounds of characters in the set of unstructured text data. In some further embodiments a sub-embedding of the plurality of sub-embeddings is a visual sub-embedding generated based on one or more extracted image-based aspects of the set of unstructured text data. In some further embodiments a sub-embedding of the plurality of sub-embeddings is a relative font sub-embedding generated based on one or more different visual fonts of text in the set of unstructured text data.

[0012] In some embodiments, the one or more text groupings are one or more sentences of structured characters extracted by processing the set of unstructured text data and the one or more output predictions are one or more sentence labels, each sentence label corresponding to predicted relative order of a sentence in a group of related sentences. In

some further embodiments, the method further comprises determining, by the data processing system, a set of ground-truth training data, the set of ground-truth training data comprising at least a known label corresponding to a sentence label of the one or more sentence labels and training the machine learning model by comparing the sentence label of the one or more sentence labels to the a corresponding known label to determine an objective value and modifying a configuration of the machine learning model based on the objective value. In some embodiments, the method further comprises processing the one or more sentences and the one or more sentence labels to generate one or more question-and-answer pairs, each of the one or more question-and-answer pairs associated with at least one sentence as a textual question and at least one corresponding sentence as a textual answer to the textual question.

[0013] The techniques described above and below may be implemented in a number of ways and in a number of contexts. Several example implementations and contexts are provided with reference to the following figures, as described below in more detail. However, the following implementations and contexts are but a few of many.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] FIG. 1 is a simplified block diagram of a distributed environment incorporating an exemplary embodiment.

[0015] FIG. 2 is a simplified block diagram of a master bot (MB) system according to certain embodiments.

[0016] FIG. 3 is a simplified block diagram of a skill bot system according to certain embodiments.

[0017] FIG. 4 is a simplified block diagram of a computing system implementing a data processing system for text embedding generation according to various embodiments.

[0018] FIG. 5 is a simplified block diagram of a computing system for training a machine learning model according to various embodiments.

[0019] FIG. 6 is a simplified block diagram of a computing system for generating inferences by a machine learning model according to various embodiments.

[0020] FIG. 7 is a simplified block diagram of a computing system for feature extractions of text groupings according to various embodiments.

[0021] FIG. 8 illustrates a process flow for generating text grouping embeddings for machine learning processes according to various embodiments.

[0022] FIG. 9 depicts a simplified diagram of a distributed system for implementing various embodiments.

[0023] FIG. 10 is a simplified block diagram of one or more components of a system environment by which services provided by one or more components of an embodiment system may be offered as cloud services, in accordance with various embodiments.

[0024] FIG. 11 illustrates an example computer system that may be used to implement various embodiments.

DETAILED DESCRIPTION

[0025] In the following description, for the purposes of explanation, specific details are set forth in order to provide a thorough understanding of certain embodiments. However, it will be apparent that various embodiments may be practiced without these specific details. The figures and description are not intended to be restrictive. The word "exemplary" is used herein to mean "serving as an example, instance, or illustration." Any embodiment or design described herein as "exemplary" is not necessarily to be construed as preferred or advantageous over other embodiments or designs.

[0026] Training a chatbot to interact effectively with a human is a time and resource intensive task. An untrained chatbot will not be able to process new or complex data properly and may return no answer or an incorrect answer in response to various utterances such as questions posed to a chatbot. A large volume of data is often required to train a chatbot model to a point that it can effectively answer human utterances most of the time. This data includes labelled training data where a chatbot model, such as a neural network, may be trained by generating predicted data for an utterance and then comparing the predicted data to ground-truth data. The difference between the predicted data and the ground-truth data can be measured using an objective function and used to refine the model (e.g., learn model parameters via back propagation).

[0027] The labelled training data is typically generated manually by reviewing documents with data sought to be included as training data. A human may personally parse the document for question-and-answer pairs that can be transformed into training data for a chatbot model. The human will copy the question and the answer for a question that the chatbot should

"learn" and use the pair to generate a training dataset that will be used to train the chatbot. This requires that the human user manually review documents for question and answer pairs, often unaware of where they may be or missing them due to human error. This contributes to inefficient utilization of resources and a delay in training and deployment of chatbot models.

[0028] In order to overcome these challenges and other, described herein are deep learning techniques for extraction of embedded data from documents that may be used to train a chatbot model. The deep learning techniques described herein allow for automatic and dynamic extraction of embedded data that may be used to train a machine learning model without requiring human action or intervention. Automated and dynamic extraction will allow a chatbot model to be trained more quickly, comprehensively, and thoroughly due to the elimination of human error and inefficiency in the training data generation processes. For example, deep learning techniques may not just extract question and answer pairs from digital documents faster and more accurately than humans but may also avoid human biases and include contextually relevant embeddings in the training data that humans may not be aware of or would otherwise not include when manually generating training data.

Example Chatbot System

[0029] A bot (also referred to as a skill, chatbot, chatterbot, or talkbot) is a computer program that can perform conversations with end users. The bot can generally respond to natural-language messages (e.g., questions or comments) through a messaging application that uses natural-language messages. Enterprises may use one or more bot systems to communicate with end users through a messaging application. The messaging application, which may be referred to as a channel, may be an end user preferred messaging application that the end user has already installed and familiar with. Thus, the end user does not need to download and install new applications in order to chat with the bot system. The messaging application may include, for example, over-the-top (OTT) messaging channels (such as Facebook Messenger, Facebook WhatsApp, WeChat, Line, Kik, Telegram, Talk, Skype, Slack, or SMS), virtual private assistants (such as Amazon Dot, Echo, or Show, Google Home, Apple HomePod, etc.), mobile and web app extensions that extend native or hybrid/responsive mobile apps or web applications with chat capabilities, or voice based input (such as devices or apps with interfaces that use Siri, Cortana, Google Voice, or other speech input for interaction).

[0030] In some examples, a bot system may be associated with a Uniform Resource Identifier (URI). The URI may identify the bot system using a string of characters. The URI may be used as a webhook for one or more messaging application systems. The URI may include, for example, a Uniform Resource Locator (URL) or a Uniform Resource Name (URN). The bot system may be designed to receive a message (e.g., a hypertext transfer protocol (HTTP) post call message) from a messaging application system. The HTTP post call message may be directed to the URI from the messaging application system. In some embodiments, the message may be different from a HTTP post call message. For example, the bot system may receive a message from a Short Message Service (SMS). While discussion herein may refer to communications that the bot system receives as a message, it should be understood that the message may be an HTTP post call message, a SMS message, or any other type of communication between two systems.

[0031] End users may interact with the bot system through a conversational interaction (sometimes referred to as a conversational user interface (UI)), just as interactions between people. In some cases, the interaction may include the end user saying "Hello" to the bot and the bot responding with a "Hi" and asking the end user how it can help. In some cases, the interaction may also be a transactional interaction with, for example, a banking bot, such as transferring money from one account to another; an informational interaction with, for example, a HR bot, such as checking for vacation balance; or an interaction with, for example, a retail bot, such as discussing returning purchased goods or seeking technical support.

[0032] In some embodiments, the bot system may intelligently handle end user interactions without interaction with an administrator or developer of the bot system. For example, an end user may send one or more messages to the bot system in order to achieve a desired goal. A message may include certain content, such as text, emojis, audio, image, video, or other method of conveying a message. In some embodiments, the bot system may convert the content into a standardized form (e.g., a representational state transfer (REST) call against enterprise services with the proper parameters) and generate a natural language response. The bot system may also prompt the end user for additional input parameters or request other additional information. In some embodiments, the bot system may also initiate communication with the end user, rather than passively responding to end user utterances. Described herein are various techniques for identifying an explicit invocation of a bot system and determining an input for the bot system being invoked. In certain embodiments, explicit

invocation analysis is performed by a master bot based on detecting an invocation name in an utterance. In response to detection of the invocation name, the utterance may be refined for input to a skill bot associated with the invocation name.

[0033] A conversation with a bot may follow a specific conversation flow including multiple states. The flow may define what would happen next based on an input. In some embodiments, a state machine that includes user defined states (e.g., end user intents) and actions to take in the states or from state to state may be used to implement the bot system. A conversation may take different paths based on the end user input, which may impact the decision the bot makes for the flow. For example, at each state, based on the end user input or utterances, the bot may determine the end user's intent in order to determine the appropriate next action to take. As used herein and in the context of an utterance, the term "intent" refers to an intent of the user who provided the utterance. For example, the user may intend to engage a bot in conversation for ordering pizza, so that the user's intent could be represented through the utterance "Order pizza." A user intent can be directed to a particular task that the user wishes a chatbot to perform on behalf of the user. Therefore, utterances can be phrased as questions, commands, requests, and the like, that reflect the user's intent. An intent may include a goal that the end user would like to accomplish.

[0034] In the context of the configuration of a chatbot, the term "intent" is used herein to refer to configuration information for mapping a user's utterance to a specific task/action or category of task/action that the chatbot can perform. In order to distinguish between the intent of an utterance (i.e., a user intent) and the intent of a chatbot, the latter is sometimes referred to herein as a "bot intent." A bot intent may comprise a set of one or more utterances associated with the intent. For instance, an intent for ordering pizza can have various permutations of utterances that express a desire to place an order for pizza. These associated utterances can be used to train an intent classifier of the chatbot to enable the intent classifier to subsequently determine whether an input utterance from a user matches the order pizza intent. A bot intent may be associated with one or more dialog flows for starting a conversation with the user and in a certain state. For example, the first message for the order pizza intent could be the question "What kind of pizza would you like?" In addition to associated utterances, a bot intent may further comprise named entities that relate to the intent. For example, the order pizza intent could include variables or parameters used to perform the task of ordering pizza, e.g., topping 1, topping 2, pizza type, pizza size, pizza

quantity, and the like. The value of an entity is typically obtained through conversing with the user.

[0035] FIG. 1 is a simplified block diagram of an environment 100 incorporating a chatbot system according to certain embodiments. Environment 100 comprises a digital assistant builder platform (DABP) 102 that enables users of DABP 102 to create and deploy digital assistants or chatbot systems. DABP 102 can be used to create one or more digital assistants (or DAs) or chatbot systems. For example, as shown in Fig. 1, user 104 representing a particular enterprise can use DABP 102 to create and deploy a digital assistant 106 for users of the particular enterprise. For example, DABP 102 can be used by a bank to create one or more digital assistants for use by the bank's customers. The same DABP 102 platform can be used by multiple enterprises to create digital assistants. As another example, an owner of a restaurant (e.g., a pizza shop) may use DABP 102 to create and deploy a digital assistant that enables customers of the restaurant to order food (e.g., order pizza).

[0036] For purposes of this disclosure, a "digital assistant" is an entity that helps users of the digital assistant accomplish various tasks through natural language conversations. A digital assistant can be implemented using software only (e.g., the digital assistant is a digital entity implemented using programs, code, or instructions executable by one or more processors), using hardware, or using a combination of hardware and software. A digital assistant can be embodied or implemented in various physical systems or devices, such as in a computer, a mobile phone, a watch, an appliance, a vehicle, and the like. A digital assistant is also sometimes referred to as a chatbot system. Accordingly, for purposes of this disclosure, the terms digital assistant and chatbot system are interchangeable.

[0037] A digital assistant, such as digital assistant 106 built using DABP 102, can be used to perform various tasks via natural language-based conversations between the digital assistant and its users 108. As part of a conversation, a user may provide one or more user inputs 110 to digital assistant 106 and get responses 112 back from digital assistant 106. A conversation can include one or more of inputs 110 and responses 112. Via these conversations, a user can request one or more tasks to be performed by the digital assistant and, in response, the digital assistant is configured to perform the user-requested tasks and respond with appropriate responses to the user.

[0038] User inputs 110 are generally in a natural language form and are referred to as utterances. A user utterance 110 can be in text form, such as when a user types in a sentence,

a question, a text fragment, or even a single word and provides it as input to digital assistant 106. In some embodiments, a user utterance 110 can be in audio input or speech form, such as when a user says or speaks something that is provided as input to digital assistant 106. The utterances are typically in a language spoken by the user 108. For example, the utterances may be in English, or some other language. When an utterance is in speech form, the speech input is converted to text form utterances in that particular language and the text utterances are then processed by digital assistant 106. Various speech-to-text processing techniques may be used to convert a speech or audio input to a text utterance, which is then processed by digital assistant 106. In some embodiments, the speech-to-text conversion may be done by digital assistant 106 itself.

[0039] An utterance, which may be a text utterance or a speech utterance, can be a fragment, a sentence, multiple sentences, one or more words, one or more questions, combinations of the aforementioned types, and the like. Digital assistant 106 is configured to apply natural language understanding (NLU) techniques to the utterance to understand the meaning of the user input. As part of the NLU processing for a utterance, digital assistant 106 is configured to perform processing to understand the meaning of the utterance, which involves identifying one or more intents and one or more entities corresponding to the utterance. Upon understanding the meaning of an utterance, digital assistant 106 may perform one or more actions or operations responsive to the understood meaning or intents. For purposes of this disclosure, it is assumed that the utterances are text utterances that have been provided directly by a user 108 of digital assistant 106 or are the results of conversion of input speech utterances to text form. This however is not intended to be limiting or restrictive in any manner.

[0040] For example, a user 108 input may request a pizza to be ordered by providing an utterance such as "I want to order a pizza." Upon receiving such an utterance, digital assistant 106 is configured to understand the meaning of the utterance and take appropriate actions. The appropriate actions may involve, for example, responding to the user with questions requesting user input on the type of pizza the user desires to order, the size of the pizza, any toppings for the pizza, and the like. The responses provided by digital assistant 106 may also be in natural language form and typically in the same language as the input utterance. As part of generating these responses, digital assistant 106 may perform natural language generation (NLG). For the user ordering a pizza, via the conversation between the user and digital assistant 106, the digital assistant may guide the user to provide all the

requisite information for the pizza order, and then at the end of the conversation cause the pizza to be ordered. Digital assistant 106 may end the conversation by outputting information to the user indicating that the pizza has been ordered.

[0041] At a conceptual level, digital assistant 106 performs various processing in response to an utterance received from a user. In some embodiments, this processing involves a series or pipeline of processing steps including, for example, understanding the meaning of the input utterance (sometimes referred to as Natural Language Understanding (NLU), determining an action to be performed in response to the utterance, where appropriate causing the action to be performed, generating a response to be output to the user responsive to the user utterance, outputting the response to the user, and the like. The NLU processing can include parsing the received input utterance to understand the structure and meaning of the utterance, refining and reforming the utterance to develop a better understandable form (e.g., logical form) or structure for the utterance. Generating a response may include using NLG techniques.

[0042] The NLU processing performed by a digital assistant, such as digital assistant 106, can include various NLP related processing such as sentence parsing (e.g., tokenizing, lemmatizing, identifying part-of-speech tags for the sentence, identifying named entities in the sentence, generating dependency trees to represent the sentence structure, splitting a sentence into clauses, analyzing individual clauses, resolving anaphoras, performing chunking, and the like). In certain embodiments, the NLU processing or portions thereof is performed by digital assistant 106 itself. In some other embodiments, digital assistant 106 may use other resources to perform portions of the NLU processing. For example, the syntax and structure of an input utterance sentence may be identified by processing the sentence using a parser, a part-of-speech tagger, and/or a named entity recognizer. In one implementation, for the English language, a parser, a part-of-speech tagger, and a named entity recognizer such as ones provided by the Stanford Natural Language Processing (NLP) Group are used for analyzing the sentence structure and syntax. These are provided as part of the Stanford CoreNLP toolkit.

[0043] While the various examples provided in this disclosure show utterances in the English language, this is meant only as an example. In certain embodiments, digital assistant 106 is also capable of handling utterances in languages other than English. Digital assistant 106 may provide subsystems (e.g., components implementing NLU functionality) that are

configured for performing processing for different languages. These subsystems may be implemented as pluggable units that can be called using service calls from an NLU core server. This makes the NLU processing flexible and extensible for each language, including allowing different orders of processing. A language pack may be provided for individual languages, where a language pack can register a list of subsystems that can be served from the NLU core server.

[0044] A digital assistant, such as digital assistant 106 depicted in FIG. 1, can be made available or accessible to its users 108 through a variety of different channels, such as but not limited to, via certain applications, via social media platforms, via various messaging services and applications, and other applications or channels. A single digital assistant can have several channels configured for it so that it can be run on and be accessed by different services simultaneously.

[0045] A digital assistant or chatbot system generally contains or is associated with one or more skills. In certain embodiments, these skills are individual chatbots (referred to as skill bots) that are configured to interact with users and fulfill specific types of tasks, such as tracking inventory, submitting timecards, creating expense reports, ordering food, checking a bank account, making reservations, buying a widget, and the like. For example, for the embodiment depicted in Fig. 1, digital assistant or chatbot system 106 includes skills 116-1, 116-2, and so on. For purposes of this disclosure, the terms "skill" and "skills" are used synonymously with the terms "skill bot" and "skill bots," respectively.

[0046] Each skill associated with a digital assistant helps a user of the digital assistant complete a task through a conversation with the user, where the conversation can include a combination of text or audio inputs provided by the user and responses provided by the skill bots. These responses may be in the form of text or audio messages to the user and/or using simple user interface elements (e.g., select lists) that are presented to the user for the user to make selections.

[0047] There are various ways in which a skill or skill bot can be associated or added to a digital assistant. In some instances, a skill bot can be developed by an enterprise and then added to a digital assistant using DABP 102. In other instances, a skill bot can be developed and created using DABP 102 and then added to a digital assistant created using DABP 102. In yet other instances, DABP 102 provides an online digital store (referred to as a "skills store") that offers multiple skills directed to a wide range of tasks. The skills offered through

the skills store may also expose various cloud services. In order to add a skill to a digital assistant being generated using DABP 102, a user of DABP 102 can access the skills store via DABP 102, select a desired skill, and indicate that the selected skill is to be added to the digital assistant created using DABP 102. A skill from the skills store can be added to a digital assistant as is or in a modified form (for example, a user of DABP 102 may select and clone a particular skill bot provided by the skills store, make customizations or modifications to the selected skill bot, and then add the modified skill bot to a digital assistant created using DABP 102).

[0048] Various different architectures may be used to implement a digital assistant or chatbot system. For example, in certain embodiments, the digital assistants created and deployed using DABP 102 may be implemented using a master bot/child(or sub) bot paradigm or architecture. According to this paradigm, a digital assistant is implemented as a master bot that interacts with one or more child bots that are skill bots. For example, in the embodiment depicted in FIG. 1, digital assistant 106 comprises a master bot 114 and skill bots 116–1, 116-2, etc. that are child bots of master bot 114. In certain embodiments, digital assistant 106 is itself considered to act as the master bot.

[0049] A digital assistant implemented according to the master-child bot architecture enables users of the digital assistant to interact with multiple skills through a unified user interface, namely via the master bot. When a user engages with a digital assistant, the user input is received by the master bot. The master bot then performs processing to determine the meaning of the user input utterance. The master bot then determines whether the task requested by the user in the utterance can be handled by the master bot itself, else the master bot selects an appropriate skill bot for handling the user request and routes the conversation to the selected skill bot. This enables a user to converse with the digital assistant through a common single interface and still provide the capability to use several skill bots configured to perform specific tasks. For example, for a digital assistance developed for an enterprise, the master bot of the digital assistant may interface with skill bots with specific functionalities, such as a CRM bot for performing functions related to customer relationship management (CRM), an ERP bot for performing functions related to enterprise resource planning (ERP), an HCM bot for performing functions related to human capital management (HCM), etc. This way the end user or consumer of the digital assistant need only know how to access the digital assistant through the common master bot interface and behind the scenes multiple skill bots are provided for handling the user request.

[0050] In certain embodiments, in a master bot/child bots infrastructure, the master bot is configured to be aware of the available list of skill bots. The master bot may have access to metadata that identifies the various available skill bots, and for each skill bot, the capabilities of the skill bot including the tasks that can be performed by the skill bot. Upon receiving a user request in the form of an utterance, the master bot is configured to, from the multiple available skill bots, identify or predict a specific skill bot that can best serve or handle the user request. The master bot then routes the utterance (or a portion of the utterance) to that specific skill bot for further handling. Control thus flows from the master bot to the skill bots. The master bot can support multiple input and output channels.

[0051] While the embodiment in FIG. 1 shows digital assistant 106 comprising a master bot 114 and skill bots 116-1, 116-2, and 116-3, this is not intended to be limiting. A digital assistant can include various other components (e.g., other systems and subsystems) that provide the functionalities of the digital assistant. These systems and subsystems may be implemented only in software (e.g., code, instructions stored on a computer-readable medium and executable by one or more processors), in hardware only, or in implementations that use a combination of software and hardware.

[0052] DABP 102 provides an infrastructure and various services and features that enable a user of DABP 102 to create a digital assistant including one or more skill bots associated with the digital assistant. In some instances, a skill bot can be created by cloning an existing skill bot, for example, cloning a skill bot provided by the skills store. As previously indicated, DABP 102 provides a skills store or skills catalog that offers multiple skill bots for performing various tasks. A user of DABP 102 can clone a skill bot from the skills store. As needed, modifications or customizations may be made to the cloned skill bot. In some other instances, a user of DABP 102 created a skill bot from scratch using tools and services offered by DABP 102. As previously indicated, the skills store or skills catalog provided by DABP 102 may offer multiple skill bots for performing various tasks.

[0053] In certain embodiments, at a high level, creating or customizing a skill bot involves the following steps:

(1) Configuring settings for a new skill bot

(2) Configuring one or more intents for the skill bot

(3) Configuring one or more entities for one or more intents

(4) Training the skill bot

(5) Creating a dialog flow for the skill bot

(6) Adding custom components to the skill bot as needed

(7) Testing and deploying the skill bot

Each of the above steps is briefly described below.

[0054] (1) Configuring settings for a new skill bot – Various settings may be configured for the skill bot. For example, a skill bot designer can specify one or more invocation names for the skill bot being created. These invocation names can then be used by users of a digital assistant to explicitly invoke the skill bot. For example, a user can input an invocation name in the user's utterance to explicitly invoke the corresponding skill bot.

[0055] (2) Configuring one or more intents and associated example utterances for the skill bot– The skill bot designer specifies one or more intents (also referred to as bot intents) for a skill bot being created. The skill bot is then trained based upon these specified intents. These intents represent categories or classes that the skill bot is trained to infer for input utterances. Upon receiving an utterance, a trained skill bot infers an intent for the utterance, where the inferred intent is selected from the predefined set of intents used to train the skill bot. The skill bot then takes an appropriate action responsive to an utterance based upon the intent inferred for that utterance. In some instances, the intents for a skill bot represent tasks that the skill bot can perform for users of the digital assistant. Each intent is given an intent identifier or intent name. For example, for a skill bot trained for a bank, the intents specified for the skill bot may include "CheckBalance," "TransferMoney," "DepositCheck," and the like.

[0056] For each intent defined for a skill bot, the skill bot designer may also provide one or more example utterances that are representative of and illustrate the intent. These example utterances are meant to represent utterances that a user may input to the skill bot for that intent. For example, for the CheckBalance intent, example utterances may include "What's my savings account balance?", "How much is in my checking account?", "How much money do I have in my account," and the like. Accordingly, various permutations of typical user utterances may be specified as example utterances for an intent.

[0057] The intents and the their associated example utterances are used as training data to train the skill bot. Various different training techniques may be used. As a result of this training, a predictive model is generated that is configured to take an utterance as input and output an intent inferred for the utterance by the predictive model. In some instances, input utterances are provided to an intent analysis engine, which is configured to use the trained model to predict or infer an intent for the input utterance. The skill bot may then take one or more actions based upon the inferred intent.

[0058] (3) Configuring entities for one or more intents of the skill bot – In some instances, additional context may be needed to enable the skill bot to properly respond to a user utterance. For example, there may be situations where a user input utterance resolves to the same intent in a skill bot. For instance, in the above example, utterances "What's my savings account balance?" and "How much is in my checking account?" both resolve to the same CheckBalance intent, but these utterances are different requests asking for different things. To clarify such requests, one or more entities are added to an intent. Using the banking skill bot example, an entity called AccountType, which defines values called "checking" and "saving" may enable the skill bot to parse the user request and respond appropriately. In the above example, while the utterances resolve to the same intent, the value associated with the AccountType entity is different for the two utterances. This enables the skill bot to perform possibly different actions for the two utterances in spite of them resolving to the same intent. One or more entities can be specified for certain intents configured for the skill bot. Entities are thus used to add context to the intent itself. Entities help describe an intent more fully and enable the skill bot to complete a user request.

[0059] In certain embodiments, there are two types of entities: (a) built-in entities provided by DABP 102, and (2) custom entities that can be specified by a skill bot designer. Built-in entities are generic entities that can be used with a wide variety of bots. Examples of built-in entities include, without limitation, entities related to time, date, addresses, numbers, email addresses, duration, recurring time periods, currencies, phone numbers, URLs, and the like. Custom entities are used for more customized applications. For example, for a banking skill, an AccountType entity may be defined by the skill bot designer that enables various banking transactions by checking the user input for keywords like checking, savings, and credit cards, etc.

[0060] (4) Training the skill bot – A skill bot is configured to receive user input in the form of utterances parse or otherwise process the received input, and identify or select an intent that is relevant to the received user input. As indicated above, the skill bot has to be trained for this. In certain embodiments, a skill bot is trained based upon the intents configured for the skill bot and the example utterances associated with the intents (collectively, the training data), so that the skill bot can resolve user input utterances to one of its configured intents. In certain embodiments, the skill bot uses a predictive model that is trained using the training data and allows the skill bot to discern what users say (or in some cases, are trying to say). DABP 102 provides various different training techniques that can be used by a skill bot designer to train a skill bot, including various machine-learning based training techniques, rules-based training techniques, and/or combinations thereof. In certain embodiments, a portion (e.g., 80%) of the training data is used to train a skill bot model and another portion (e.g., the remaining 20%) is used to test or verify the model. Once trained, the trained model (also sometimes referred to as the trained skill bot) can then be used to handle and respond to user utterances. In certain cases, a user's utterance may be a question that requires only a single answer and no further conversation. In order to handle such situations, a Q&A (question-and-answer) intent may be defined for a skill bot. This enables a skill bot to output replies to user requests without having to update the dialog definition. Q&A intents are created in a similar manner as regular intents. The dialog flow for Q&A intents can be different from that for regular intents.

[0061] (5) Creating a dialog flow for the skill bot -- A dialog flow specified for a skill bot describes how the skill bot reacts as different intents for the skill bot are resolved responsive to received user input. The dialog flow defines operations or actions that a skill bot will take, e.g., how the skill bot responds to user utterances, how the skill bot prompts users for input, how the skill bot returns data. A dialog flow is like a flowchart that is followed by the skill bot. The skill bot designer specifies a dialog flow using a language, such as markdown language. In certain embodiments, a version of YAML called OBotML may be used to specify a dialog flow for a skill bot. The dialog flow definition for a skill bot acts as a model for the conversation itself, one that lets the skill bot designer choreograph the interactions between a skill bot and the users that the skill bot services.

[0062] In certain embodiments, the dialog flow definition for a skill bot contains three sections:

(a) a context section

(b) a default transitions section

(c) a states section

[0063] Context section -- The skill bot designer can define variables that are used in a conversation flow in the context section. Other variables that may be named in the context section include, without limitation: variables for error handling, variables for built-in or custom entities, user variables that enable the skill bot to recognize and persist user preferences, and the like.

[0064] Default transitions section -- Transitions for a skill bot can be defined in the dialog flow states section or in the default transitions section. The transitions defined in the default transition section act as a fallback and get triggered when there are no applicable transitions defined within a state, or the conditions required to trigger a state transition cannot be met. The default transitions section can be used to define routing that allows the skill bot to gracefully handle unexpected user actions.

[0065] States section – A dialog flow and its related operations are defined as a sequence of transitory states, which manage the logic within the dialog flow. Each state node within a dialog flow definition names a component that provides the functionality needed at that point in the dialog. States are thus built around the components. A state contains component-specific properties and defines the transitions to other states that get triggered after the component executes.

[0066] Special case scenarios may be handled using the states sections. For example, there might be times when you want to provide users the option to temporarily leave a first skill they are engaged with to do something in a second skill within the digital assistant. For example, if a user is engaged in a conversation with a shopping skill (e.g., the user has made some selections for purchase), the user may want to jump to a banking skill (e.g., the user may want to ensure that he/she has enough money for the purchase), and then return to the shopping skill to complete the user's order. To address this, an action in the first skill can be configured to initiate an interaction with the second different skill in the same digital assistant and then return to the original flow.

[0067] (6) Adding custom components to the skill bot – As described above, states specified in a dialog flow for a skill bot name components that provide the functionality

needed corresponding to the states. Components enable a skill bot to perform functions. In certain embodiments, DABP 102 provides a set of preconfigured components for performing a wide range of functions. A skill bot designer can select one of more of these preconfigured components and associate them with states in the dialog flow for a skill bot. The skill bot designer can also create custom or new components using tools provided by DABP 102 and associate the custom components with one or more states in the dialog flow for a skill bot.

[0068] (7) Testing and deploying the skill bot – DABP 102 provides several features that enable the skill bot designer to test a skill bot being developed. The skill bot can then be deployed and included in a digital assistant.

[0069] While the description above describes how to create a skill bot, similar techniques may also be used to create a digital assistant (or the master bot). At the master bot or digital assistant level, built-in system intents may be configured for the digital assistant. These built-in system intents are used to identify general tasks that the digital assistant itself (i.e., the master bot) can handle without invoking a skill bot associated with the digital assistant. Examples of system intents defined for a master bot include: (1) Exit: applies when the user signals the desire to exit the current conversation or context in the digital assistant; (2) Help: applies when the user asks for help or orientation; and (3) UnresolvedIntent: applies to user input that doesn't match well with the exit and help intents. The digital assistant also stores information about the one or more skill bots associated with the digital assistant. This information enables the master bot to select a particular skill bot for handling an utterance.

[0070] At the master bot or digital assistant level, when a user inputs a phrase or utterance to the digital assistant, the digital assistant is configured to perform processing to determine how to route the utterance and the related conversation. The digital assistant determines this using a routing model, which can be rules-based, AI-based, or a combination thereof. The digital assistant uses the routing model to determine whether the conversation corresponding to the user input utterance is to be routed to a particular skill for handling, is to be handled by the digital assistant or master bot itself per a built-in system intent, or is to be handled as a different state in a current conversation flow.

[0071] In certain embodiments, as part of this processing, the digital assistant determines if the user input utterance explicitly identifies a skill bot using its invocation name. If an invocation name is present in the user input, then it is treated as explicit invocation of the skill bot corresponding to the invocation name. In such a scenario, the digital assistant may

route the user input to the explicitly invoked skill bot for further handling. If there is no specific or explicit invocation, in certain embodiments, the digital assistant evaluates the received user input utterance and computes confidence scores for the system intents and the skill bots associated with the digital assistant. The score computed for a skill bot or system intent represents how likely the user input is representative of a task that the skill bot is configured to perform or is representative of a system intent. Any system intent or skill bot with an associated computed confidence score exceeding a threshold value (e.g., a Confidence Threshold routing parameter) is selected as a candidate for further evaluation. The digital assistant then selects, from the identified candidates, a particular system intent or a skill bot for further handling of the user input utterance. In certain embodiments, after one or more skill bots are identified as candidates, the intents associated with those candidate skills are evaluated (according to the intent model for each skill) and confidence scores are determined for each intent. In general, any intent that has a confidence score exceeding a threshold value (e.g., 70%) is treated as a candidate intent. If a particular skill bot is selected, then the user utterance is routed to that skill bot for further processing. If a system intent is selected, then one or more actions are performed by the master bot itself according to the selected system intent.

[0072] FIG. 2 is a simplified block diagram of a master bot (MB) system 200 according to certain embodiments. MB system 200 can be implemented in software only, hardware only, or a combination of hardware and software. MB system 200 includes a pre-processing subsystem 210, a multiple intent subsystem (MIS) 220, an explicit invocation subsystem (EIS) 230, a skill bot invoker 240, and a data store 250. MB system 200 depicted in FIG. 2 is merely an example of an arrangement of components in a master bot. One of ordinary skill in the art would recognize many possible variations, alternatives, and modifications. For example, in some implementations, MB system 200 may have more or fewer systems or components than those shown in FIG. 2, may combine two or more subsystems, or may have a different configuration or arrangement of subsystems.

[0073] Pre-processing subsystem 210 receives an utterance "A" 202 from a user and processes the utterance through a language detector 212 and a language parser 214. As indicated above, an utterance can be provided in various ways including audio or text. The utterance 202 can be a sentence fragment, a complete sentence, multiple sentences, and the like. Utterance 202 can include punctuation. For example, if the utterance 202 is provided as audio, the pre-processing subsystem 210 may convert the audio to text using a speech-to-text

converter (not shown) that inserts punctuation marks into the resulting text, e.g., commas, semicolons, periods, etc.

[0074] Language detector 212 detects the language of the utterance 202 based on the text of the utterance 202. The manner in which the utterance 202 is handled depends on the language since each language has its own grammar and semantics. Differences between languages are taken into consideration when analyzing the syntax and structure of an utterance.

[0075] Language parser 214 parses the utterance 202 to extract part of speech (POS) tags for individual linguistic units (e.g., words) in the utterance 202. POS tags include, for example, noun (NN), pronoun (PN), verb (VB), and the like. Language parser 214 may also tokenize the linguistic units of the utterance 202 (e.g., to convert each word into a separate token) and lemmatize words. A lemma is the main form of a set of words as represented in a dictionary (e.g., "run" is the lemma for run, runs, ran, running, etc.). Other types of pre-processing that the language parser 214 can perform include chunking of compound expressions, e.g., combining "credit" and "card" into a single expression "credit_card." Language parser 214 may also identify relationships between the words in the utterance 202. For example, in some embodiments, the language parser 214 generates a dependency tree that indicates which part of the utterance (e.g. a particular noun) is a direct object, which part of the utterance is a preposition, and so on. The results of the processing performed by the language parser 214 form extracted information 205 and are provided as input to MIS 220 together with the utterance 202 itself.

[0076] As indicated above, the utterance 202 can include more than one sentence. For purposes of detecting multiple intents and explicit invocation, the utterance 202 can be treated as a single unit even if it includes multiple sentences. However, in certain embodiments, pre-processing can be performed, e.g., by the pre-processing subsystem 210, to identify a single sentence among multiple sentences for multiple intents analysis and explicit invocation analysis. In general, the results produced by MIS 220 and EIS 230 are substantially the same regardless of whether the utterance 202 is processed at the level of an individual sentence or as a single unit comprising multiple sentences.

[0077] MIS 220 determines whether the utterance 202 represents multiple intents. Although MIS 220 can detect the presence of multiple intents in the utterance 202, the processing performed by MIS 220 does not involve determining whether the intents of the

utterance 202 match to any intents that have been configured for a bot. Instead, processing to determine whether an intent of the utterance 202 matches a bot intent can be performed by an intent classifier 242 of the MB system 200 or by an intent classifier of a skill bot (e.g., as shown in the embodiment of FIG. 3). The processing performed by MIS 220 assumes that there exists a bot (e.g., a particular skill bot or the master bot itself) that can handle the utterance 202. Therefore, the processing performed by MIS 220 does not require knowledge of what bots are in the chatbot system (e.g., the identities of skill bots registered with the master bot) or knowledge of what intents have been configured for a particular bot.

[0078] To determine that the utterance 202 includes multiple intents, the MIS 220 applies one or more rules from a set of rules 252 in the data store 250. The rules applied to the utterance 202 depend on the language of the utterance 202 and may include sentence patterns that indicate the presence of multiple intents. For example, a sentence pattern may include a coordinating conjunction that joins two parts (e.g., conjuncts) of a sentence, where both parts correspond to a separate intent. If the utterance 202 matches the sentence pattern, it can be inferred that the utterance 202 represents multiple intents. It should be noted that an utterance with multiple intents does not necessarily have different intents (e.g., intents directed to different bots or to different intents within the same bot). Instead, the utterance could have separate instances of the same intent, e.g. "Place a pizza order using payment account X, then place a pizza order using payment account Y."

[0079] As part of determining that the utterance 202 represents multiple intents, the MIS 220 also determines what portions of the utterance 202 are associated with each intent. MIS 220 constructs, for each intent represented in an utterance containing multiple intents, a new utterance for separate processing in place of the original utterance, e.g., an utterance "B" 206 and an utterance "C" 208, as depicted in FIG. 2. Thus, the original utterance 202 can be split into two or more separate utterances that are handled one at a time. MIS 220 determines, using the extracted information 205 and/or from analysis of the utterance 202 itself, which of the two or more utterances should be handled first. For example, MIS 220 may determine that the utterance 202 contains a marker word indicating that a particular intent should be handled first. The newly formed utterance corresponding to this particular intent (e.g., one of utterance 206 or utterance 208) will be the first to be sent for further processing by EIS 230. After a conversation triggered by the first utterance has ended (or has been temporarily suspended), the next highest priority utterance (e.g., the other one of utterance 206 or utterance 208) can then be sent to the EIS 230 for processing.

[0080] EIS 230 determines whether the utterance that it receives (e.g., utterance 206 or utterance 208) contains an invocation name of a skill bot. In certain embodiments, each skill bot in a chatbot system is assigned a unique invocation name that distinguishes the skill bot from other skill bots in the chatbot system. A list of invocation names can be maintained as part of skill bot information 254 in data store 250. An utterance is deemed to be an explicit invocation when the utterance contains a word match to an invocation name. If a bot is not explicitly invoked, then the utterance received by the EIS 230 is deemed a non-explicitly invoking utterance 234 and is input to an intent classifier (e.g., intent classifier 242) of the master bot to determine which bot to use for handling the utterance. In some instances, the intent classifier 242 will determine that the master bot should handle a non-explicitly invoking utterance. In other instances, the intent classifier 242 will determine a skill bot to route the utterance to for handling.

[0081] The explicit invocation functionality provided by the EIS 230 has several advantages. It can reduce the amount of processing that the master bot has to perform. For example, when there is an explicit invocation, the master bot may not have to do any intent classification analysis (e.g., using the intent classifier 242), or may have to do reduced intent classification analysis for selecting a skill bot. Thus, explicit invocation analysis may enable selection of a particular skill bot without resorting to intent classification analysis.

[0082] Also, there may be situations where there is an overlap in functionalities between multiple skill bots. This may happen, for example, if the intents handled by the two skill bots overlap or are very close to each other. In such a situation, it may be difficult for the master bot to identify which of the multiple skill bots to select based upon intent classification analysis alone. In such scenarios, the explicit invocation disambiguates the particular skill bot to be used

[0083] In addition to determining that an utterance is an explicit invocation, the EIS 230 is responsible for determining whether any portion of the utterance should be used as input to the skill bot being explicitly invoked. In particular, EIS 230 can determine whether part of the utterance is not associated with the invocation. The EIS 230 can perform this determination through analysis of the utterance and/or analysis of the extracted information 205. EIS 230 can send the part of the utterance not associated with the invocation to the invoked skill bot in lieu of sending the entire utterance that was received by the EIS 230. In some instances, the input to the invoked skill bot is formed simply by removing any portion of the utterance

associated with the invocation. For example, "I want to order pizza using Pizza Bot" can be shortened to "I want to order pizza" since "using Pizza Bot" is relevant to the invocation of the pizza bot, but irrelevant to any processing to be performed by the pizza bot. In some instances, EIS 230 may reformat the part to be sent to the invoked bot, e.g., to form a complete sentence. Thus, the EIS 230 determines not only that there is an explicit invocation, but also what to send to the skill bot when there is an explicit invocation. In some instances, there may not be any text to input to the bot being invoked. For example, if the utterance was "Pizza Bot", then the EIS 230 could determine that the pizza bot is being invoked, but there is no text to be processed by the pizza bot. In such scenarios, the EIS 230 may indicate to the skill bot invoker 240 that there is nothing to send.

[0084] Skill bot invoker 240 invokes a skill bot in various ways. For instance, skill bot invoker 240 can invoke a bot in response to receiving an indication 235 that a particular skill bot has been selected as a result of an explicit invocation. The indication 235 can be sent by the EIS 230 together with the input for the explicitly invoked skill bot. In this scenario, the skill bot invoker 240 will turn control of the conversation over to the explicitly invoked skill bot. The explicitly invoked skill bot will determine an appropriate response to the input from the EIS 230 by treating the input as a stand-alone utterance. For example, the response could be to perform a specific action or to start a new conversation in a particular state, where the initial state of the new conversation depends on the input sent from the EIS 230.

[0085] Another way in which skill bot invoker 240 can invoke a skill bot is through implicit invocation using the intent classifier 242. The intent classifier 242 can be trained, using machine-learning and/or rules-based training techniques, to determine a likelihood that an utterance is representative of a task that a particular skill bot is configured to perform. The intent classifier 242 is trained on different classes, one class for each skill bot. For instance, whenever a new skill bot is registered with the master bot, a list of example utterances associated with the new skill bot can be used to train the intent classifier 242 to determine a likelihood that a particular utterance is representative of a task that the new skill bot can perform. The parameters produced as result of this training (e.g., a set of values for parameters of a machine-learning model) can be stored as part of skill bot information 254.

[0086] In certain embodiments, the intent classifier 242 is implemented using a machine-learning model, as described in further detail herein. Training of the machine-learning model may involve inputting at least a subset of utterances from the example utterances associated

with various skill bots to generate, as an output of the machine-learning model, inferences as to which bot is the correct bot for handling any particular training utterance. For each training utterance, an indication of the correct bot to use for the training utterance may be provided as ground truth information. The behavior of the machine-learning model can then be adapted (e.g., through back-propagation) to minimize the difference between the generated inferences and the ground truth information.

[0087] In certain embodiments, the intent classifier 242 determines, for each skill bot registered with the master bot, a confidence score indicating a likelihood that the skill bot can handle an utterance (e.g., the non-explicitly invoking utterance 234 received from EIS 230). The intent classifier 242 may also determine a confidence score for each system level intent (e.g., help, exit) that has been configured. If a particular confidence score meets one or more conditions, then the skill bot invoker 240 will invoke the bot associated with the particular confidence score. For example, a threshold confidence score value may need to be met. Thus, an output 245 of the intent classifier 242 is either an identification of a system intent or an identification of a particular skill bot. In some embodiments, in addition to meeting a threshold confidence score value, the confidence score must exceed the next highest confidence score by a certain win margin. Imposing such a condition would enable routing to a particular skill bot when the confidence scores of multiple skill bots each exceed the threshold confidence score value.

[0088] After identifying a bot based on evaluation of confidence scores, the skill bot invoker 240 hands over processing to the identified bot. In the case of a system intent, the identified bot is the master bot. Otherwise, the identified bot is a skill bot. Further, the skill bot invoker 240 will determine what to provide as input 247 for the identified bot. As indicated above, in the case of an explicit invocation, the input 247 can be based on a part of an utterance that is not associated with the invocation, or the input 247 can be nothing (e.g., an empty string). In the case of an implicit invocation, the input 247 can be the entire utterance.

[0089] Data store 250 comprises one or more computing devices that store data used by the various subsystems of the master bot system 200. As explained above, the data store 250 includes rules 252 and skill bot information 254. The rules 252 include, for example, rules for determining, by MIS 220, when an utterance represents multiple intents and how to split an utterance that represents multiple intents. The rules 252 further include rules for determining,

by EIS 230, which parts of an utterance that explicitly invokes a skill bot to send to the skill bot. The skill bot information 254 includes invocation names of skill bots in the chatbot system, e.g., a list of the invocation names of all skill bots registered with a particular master bot. The skill bot information 254 can also include information used by intent classifier 242 to determine a confidence score for each skill bot in the chatbot system, e.g., parameters of a machine-learning model.

[0090] FIG. 3 is a simplified block diagram of a skill bot system 300 according to certain embodiments. Skill bot system 300 is a computing system that can be implemented in software only, hardware only, or a combination of hardware and software. In certain embodiments such as the embodiment depicted in FIG. 1, skill bot system 300 can be used to implement one or more skill bots within a digital assistant.

[0091] Skill bot system 300 includes an MIS 310, an intent classifier 320, and a conversation manager 330. The MIS 310 is analogous to the MIS 220 in FIG. 2 and provides similar functionality, including being operable to determine, using rules 352 in a data store 350: (1) whether an utterance represents multiple intents and, if so, (2) how to split the utterance into a separate utterance for each intent of the multiple intents. In certain embodiments, the rules applied by MIS 310 for detecting multiple intents and for splitting an utterance are the same as those applied by MIS 220. The MIS 310 receives an utterance 302 and extracted information 304. The extracted information 304 is analogous to the extracted information 205 in FIG. 1 and can be generated using the language parser 214 or a language parser local to the skill bot system 300.

[0092] Intent classifier 320 can be trained in a similar manner to the intent classifier 242 discussed above in connection with the embodiment of FIG. 2 and as described in further detail herein. For instance, in certain embodiments, the intent classifier 320 is implemented using a machine-learning model. The machine-learning model of the intent classifier 320 is trained for a particular skill bot, using at least a subset of example utterances associated with that particular skill bot as training utterances. The ground truth for each training utterance would be the particular bot intent associated with the training utterance.

[0093] The utterance 302 can be received directly from the user or supplied through a master bot. When the utterance 302 is supplied through a master bot, e.g., as a result of processing through MIS 220 and EIS 230 in the embodiment depicted in FIG. 2, the MIS 310 can be bypassed so as to avoid repeating processing already performed by MIS 220.

However, if the utterance 302 is received directly from the user, e.g., during a conversation that occurs after routing to a skill bot, then MIS 310 can process the utterance 302 to determine whether the utterance 302 represents multiple intents. If so, then MIS 310 applies one or more rules to split the utterance 302 into a separate utterance for each intent, e.g., an utterance "D" 306 and an utterance "E" 308. If utterance 302 does not represent multiple intents, then MIS 310 forwards the utterance 302 to intent classifier 320 for intent classification and without splitting the utterance 302.

[0094] Intent classifier 320 is configured to match a received utterance (e.g., utterance 306 or 308) to an intent associated with skill bot system 300. As explained above, a skill bot can be configured with one or more intents, each intent including at least one example utterance that is associated with the intent and used for training a classifier. In the embodiment of FIG. 2, the intent classifier 242 of the master bot system 200 is trained to determine confidence scores for individual skill bots and confidence scores for system intents. Similarly, intent classifier 320 can be trained to determine a confidence score for each intent associated with the skill bot system 300. Whereas the classification performed by intent classifier 242 is at the bot level, the classification performed by intent classifier 320 is at the intent level and therefore finer grained. The intent classifier 320 has access to intents information 354. The intents information 354 includes, for each intent associated with the skill bot system 300, a list of utterances that are representative of and illustrate the meaning of the intent and are typically associated with a task performable by that intent. The intents information 354 can further include parameters produced as a result of training on this list of utterances.

[0095] Conversation manager 330 receives, as an output of intent classifier 320, an indication 322 of a particular intent, identified by the intent classifier 320, as best matching the utterance that was input to the intent classifier 320. In some instances, the intent classifier 320 is unable to determine any match. For example, the confidence scores computed by the intent classifier 320 could fall below a threshold confidence score value if the utterance is directed to a system intent or an intent of a different skill bot. When this occurs, the skill bot system 300 may refer the utterance to the master bot for handling, e.g., to route to a different skill bot. However, if the intent classifier 320 is successful in identifying an intent within the skill bot, then the conversation manager 330 will initiate a conversation with the user.

[0096] The conversation initiated by the conversation manager 330 is a conversation specific to the intent identified by the intent classifier 320. For instance, the conversation manager 330 may be implemented using a state machine configured to execute a dialog flow for the identified intent. The state machine can include a default starting state (e.g., for when the intent is invoked without any additional input) and one or more additional states, where each state has associated with it actions to be performed by the skill bot (e.g., executing a purchase transaction) and/or dialog (e.g., questions, responses) to be presented to the user. Thus, the conversation manager 330 can determine an action/dialog 335 upon receiving the indication 322 identifying the intent, and can determine additional actions or dialog in response to subsequent utterances received during the conversation.

[0097] Data store 350 comprises one or more computing devices that store data used by the various subsystems of the skill bot system 300. As depicted in FIG. 3, the data store 350 includes the rules 352 and the intents information 354. In certain embodiments, data store 350 can be integrated into a data store of a master bot or digital assistant, e.g., the data store 250 in FIG. 2.

Example Data Processing System

[0098] FIG. 4 is a simplified block diagram of a computing system implementing a data processing system for text embedding generation according to various embodiments. The arrangement shown in FIG. 4 is an example embodiment of a data processing system which may be utilized to perform all or part of the methods described herein, though the system shown in FIG. 4 is not the only system that may perform those methods. Data processing system 400 depicted in FIG. 4 can be implemented in software only, hardware only, or a combination of hardware and software.

[0099] As depicted in FIG. 4, data processing system 400 may be a system for training and deploying machine learning models for chatbot applications. Data processing system 400 may include various components and subsystems for receiving textual data and generating textual embeddings for training chatbot-based machine learning models. For example, data processing system 400 may include a repository of deployable chatbot models that may be trained using data from text-based documents. Subsystems of data processing system 400 may first be trained to extract training data from these text documents. Subsequent to training, different subsystems of data processing system 400 may use trained extraction

models to extract training data from text-based documents and send the training data to chatbot-based subsystems for training the chatbots to interact with humans.

[0100] As depicted in FIG. 4, data processing system 400 includes training subsystem 410, inference subsystem 420 and chatbot subsystem 430. Training subsystem 410 may be, for example, a subsystem of data processing system 400 configured to receive training data for training a deep learning model to extract textual embeddings from unstructured document data. For example, training subsystem 410 may be configured to receive unstructured text data and ground-truth data. The deep learning model may be utilized to generate output predictions, such as classification labels for sentences in the unstructured text data, and compare the output predictions to the ground-truth data. The difference between the output predictions and the ground-truth data may be used to alter the configuration of the machine learning model, for example by changing neural network model parameters in the model according to an objective function based on the difference between the output predictions and the ground-truth data.

[0101] The deep learning model may be utilized by an inference subsystem 420 to generate output predictions for a set of unstructured text data sometime after training has occurred at the training subsystem 410. For example, inference subsystem 420 may be engaged to generate output prediction classification labels for sentences in the unstructured text data as part of a service offered to customers utilizing chatbots. The inference subsystem 420 may generate the classification labels for sentences in the unstructured text data and generate corresponding training data to be utilized by a chatbot system to train chatbots. For example, a customer may send unstructured text data to the inference subsystem 420 to cause generation of training data that the customer may use to train their own chatbots.

[0102] In some embodiments, training data output by the inference subsystem 420 may be utilized by a chatbot subsystem 430 of data processing system 400 to train chatbots or otherwise store the resulting training data. For example, a chatbot subsystem 430 may receive training data generated by inference subsystem 420 and utilize the training data to train one or more chatbots associated with the chatbot subsystem 430. The use of deep learning model to generate this training data for training chatbots from unstructured text data removes the need for costly and time inefficient manual generation of training data by humans.

[0103] In an example embodiment, a client of data processing system 400 utilizes a service including a question-and-answer (QA) chatbot configured to receive questions from

humans in the form of text and send a predicted answer back to the human. The client may utilize this service and the QA chatbot as part of its own service for its own customers. The client may send unstructured text data, for example a digital PDF document including frequently asked questions (FAQs) and corresponding answers, to the data processing system in order to train the QA chatbot. A deep-learning model for extracting QA training data may have already been trained by the training subsystem 410. The trained deep-learning model may receive the unstructured text data, or a preprocessed version thereon, and generate prediction labels for the sentences therein. The prediction labels may be used to generate QA training data to train the QA chatbot, which may be stored in chatbot subsystem 430, with the clients FAQ data. As a result, the next time the QA chatbot is deployed, it is better configured to answer questions similar to those included in the FAQ document sent by the client.

[0104] Returning to FIG. 4, data processing system 400 may include communication portal 402. Communication portal 402 may be a subsystem of data processing system 400 configured to intake communications and data received from outside systems and perform the embodiment described herein. In various embodiments, communication portal 402 is configured to determine, based on some incoming document data 406, which subsystem of data processing system 400 to route the data to for further processing. For example, communication portal 402 may include chatbot routing instructions 404, which may be a set of instructions for determining how to process incoming document data 406. Document data may include, for example, metadata indicating whether the incoming data is training data to be used to train a deep learning model in training subsystem 410 or inference data to be used to generate a set of training data for a chatbot in inference subsystem 420. For example, in response to receiving incoming document data 406, the communication portal 402 may extract metadata from the incoming document data 406. The extracted metadata may be processed by chatbot routing instructions to determine if the incoming document data 406 is a training dataset to be sent to training subsystem 410 and used for training the deep learning model or an inference dataset to be sent to inference subsystem 420 and input to the deep learning model to cause generation of chatbot training data.

[0105] Training subsystem 410 may include separate subsystems for extracting classification labels by the deep learning model and training the deep learning model using the extracted classification labels and ground-truth labelled data. For example, training subsystem 410 may include an extraction subsystem 412 as a first domain in which extraction of the classification labels from unstructured text data occurs. The deep learning model may

be part of this extraction domain, as it is used to extract data and generate predicted classification labels as part of the training process for the deep learning model. The classification labels may be sent to components of a retraining subsystem 414 to generate an objective function using ground-truth labelled data. The objection function may then be utilized to retrain the deep learning model in the extraction domain. The training process of the deep learning model and domains are discussed further below with reference to FIG. 5.

[0106] Inference subsystem 420 may include separate subsystems for extracting classification labels by the deep learning model and generating training data for a chatbot model using the extracted classification labels. For example, inference subsystem 420 may include an extraction subsystem 422 as a first domain in which extraction of the classification labels from unstructured text data occurs. The extraction subsystem 422 of the inference subsystem 420 may be similar to the extraction subsystem 412 of training subsystem 410. For example, both extraction subsystem 412 and extraction subsystem 422 are domains in which a deep learning model may extract data from unstructured text data to generate classification labels for textual groupings of the unstructured text data, though those classification labels will be used to retrain the model or generate chatbot training data respectively. In the inference subsystem 420, the classification labels may be sent to components of a data generation subsystem 424 to further process the classification labels. For example, the data generation subsystem may utilize the classification labels to generate training data for one or more chatbot models. The inference and data generation process of the deep learning model and domains are discussed in greater detail below.

[0107] Chatbot subsystem 430 may be a subsystem of data processing system 400 configured to perform processing or storage functions related to the chatbots. For example, chatbot subsystem 430 may include chatbot repository 432, which may be a repository of chatbot models that may be trained and deployed for use by a service or a client. Chatbot training subsystem 434 may be a subsystem of chatbot subsystem 430 configured to utilize training data to train machine learning models of chatbots. For example, chatbot training subsystem 434 may receive training data generated by inference subsystem 420 to cause training of a chatbot model stored in chatbot repository 432. Training data repository 436 may be a repository for storing training data generated by the inference subsystem 420 for later training chatbot models. For example, training datasets associated with clients utilizing chatbots may be stored in training data repository so that training data is not required to be regenerated for each new chatbot generated for a particular client. Extraction model

repository may be a repository of deep learning model utilized as part of extraction subsystems and processes, such as extraction subsystem 412 or extraction subsystem 422. Chatbot subsystem 430 may be communicatively coupled to training subsystem 410 and inference subsystem 420 to facilitate the embodiments described herein.

Example Machine Learning Embodiments

[0108] FIG. 5 is a simplified block diagram of a computing system for training a machine learning model according to various embodiments. The block diagram depicted in FIG. 5 may be implemented in software (e.g., code, instructions, program) executed by one or more processing units (e.g., processors, cores) of the respective systems, hardware, or combinations thereof. The software may be stored on a non-transitory storage medium (e.g., on a memory device). The flow of data and components presented in FIG. 5 and described below is intended to be illustrative and non-limiting. Although FIG. 5 depicts the various steps occurring in a particular sequence or order, this is not intended to be limiting. In certain alternative embodiments, the steps may be performed in some different order or some steps may also be performed in parallel.

[0109] As depicted in FIG. 5, system 500 is a system for utilizing raw text document data to retrain one or more deep learning models to generate sentence label data. System 500 may include an extraction layer 501 and a training layer 502. In various embodiments, system 500 is a training subsystem 410, extraction layer 501 is part of extraction subsystem 412, and training layer 502 is part of retraining subsystem 414. In various embodiments, system 500 operates as part of a sentence labelling model configured to generate one or more labels corresponding to sentences extracted from a digital document containing text.

[0110] Extraction layer 501 may receive, or retrieve from a storage, raw text document data 505. Raw text document data 505 may be one or more sets of unstructured text data, such as a document, file, or other textual data. For example, raw text document data may be one or more digital files in formats such as PDF, spreadsheet, slide show, text, image, or other formats. The raw text document data 505 may include one or more visual representations of text characters representing a grammar-structured language. For example, a PDF file may contain a number of characters in a format and order that represents a set of sentences structured according to English-language grammar.

[0111] Sentence data extractor 510 may receive the raw text document data 505 in order to extract data related to one or more sentences. For example, sentence data extractor 510

may parse the raw text document data 505 in order to determine one or more aspects of the raw text document data that may be used to identify and group sentences structured according to a language grammar. In some embodiments, the sentence data extracted from the raw text document data 505 includes text data representing semantic aspects of one or more textual sentences represented in the raw text document data 505. For example, text data extracted may include a set of ASCII characters determined to be represented in the raw text document data 505. In another example, the text data includes indications of one or more semantic aspects of sentences, such as whether the sentence is compound, contains punctuation, contains capital letters, etc.

[0112] In some embodiments, the sentence data extracted from the raw text document data 505 includes box data representing bounding boxes of characters in raw text document data 505. For example, box data extracted may include a coordinate set of bounding boxes for characters in the raw text document data 505. A PDF file, for example, often inherently contains metadata corresponding to the bounding boxes of visually represented characters in the PDF document that can be extracted. In another example, the box data is generated by sentence data extractor 510 based on pixel-based representations of the characters in the document, for example, by using an artificial neural network to determine and measure bounding boxes around characters represented by the raw text document data 505.

[0113] In some embodiments, the sentence data extracted from the raw text document data 505 includes font data corresponding to font or styles of characters in raw text document data 505. For example, font data extracted may include one or more indications of a known or unknown font style for representing characters in the raw text document data 505. The font data may be generated by sentence data extractor 510 based on pixel-based representations of the characters in the document, for example, by using an artificial neural network to determine and measure relative pixel colorations for each predicted characters in the raw text document data 505. In some embodiments, the font data includes relative font data, such as data relating to difference between characters of different fonts, colors, sizes, styles, etc. in the represented by the raw text document data 505. One example of a sentence data extractor 510 is the PDFMiner text extraction toll for PDF documents available at https://pypi.org/project/pdfminer/.

[0114] System 500 may include a postprocessor 515 or similar component to further process the data extracted and generated by sentence data extractor 510. postprocessor 515

may be configured to transform raw sentence data extracted and generated by sentence data extractor 510 into a new format of sentence data that is accepted by a model. For example, postprocessor 515 may transform raw data including many sets of coordinates of bounding boxes extracted from the raw text document data into a linked list of coordinate data that is parsable by another model in system 500. The output of postprocessor 515 is sentence data 520.

[0115] Sentence data 520 may then be sent to a feature extractor 525 component of extraction layer 501. Feature extractor 525 may be a component or subsystem configured to receive sentence data as input and output embedded version of the sentence data. The output embedded version of the sentence data may be acceptable as input to one or more deep learning models configured to generate classification labels for the one or more sentences represented in the raw text document data 505. Feature extractor 525 is discussed below in more detail with regard to FIG. 5.

[0116] The feature extractor 525 will output sentence embedding data 530. Sentence embedding data 530 may be an embedded version of sentence data 520 in a format that can be accepted as input by one or more deep learning models. For example, extraction layer 501 includes contextual model 535. Contextual model 535 may be a deep learning machine learning model configured to received sentence embedding data 530 as input to output one or more predicted context values. The one or more predicted context values may be related to one or more context-aware values for the sentences extracted from the raw text document data. Contextual model 535 may be a deep learning model such as an artificial neural network that may be trained to intake sentence embedding data as input and apply a set of weights in a nodal hierarchy to generate, from a set of terminal nodes in the artificial neural network, the one or more output contextually-based values. The contextual model 535 may be retrained based on an objective function to alter the parameters/weight of the nodes in the artificial neural network.

[0117] The output context-aware values of the contextual model 535 may be further input to a classification model 540 configured to generate classification labels for the sentences extracted from the raw text document data 505. Classification model 540 may be a deep learning model configured to receive, as input, the context-aware value from contextual model 535 and output one or more sentence labels as sentence label data 545. For example, classification model 540 may be a deep learning model such as an artificial neural network

that may be trained to intake the context-aware value from contextual model 535 and apply a set of weights in a nodal hierarchy to generate, from a set of terminal nodes in the artificial neural network, one or more labels for one or more sentences extracted from the raw text document data 505. In various embodiments, the contextual model 535 and the classification model 540 are combined as a single deep learning model for intaking sentence embedding data and outputting sentence label data 545.

[0118] Sentence label data output by classification model 540 may be a set of data relating to a predicted type or structure of sentence relative to other sentences in the raw text document data 505. For example, sentence label data 545 may include, for each extracted sentence, a corresponding label representing a prediction of a type and location of a sentence relative to other sentences. Labels may indicate, for example, whether a sentence related to a questions or a statement relating to the question (e.g., an "answer"). Additionally, labels may indicate if the sentence is part of a single-sentence question/statement, a multi-sentence question/statement (e.g., a set of sentences in which each sentence depends on each other sentence to provide a full context to a question/statement), and whether a particular sentence is a beginning, end, or intermediate sentence in a multi-sentence structure. Labels may also indicate if a sentence has no representation as either a question or an answer (e.g. a series of symbols with no language-based semantic value). Thus, extraction layer 501 is configured to utilize raw text document data 505 to extract one or more sentences from the document data and generate sentence label data 545 for each of the sentences.

[0119] Training layer 502 may be a grouping of components and data that interacts with extractions layer 501 to train one or more deep learning models included in the extraction layer 501. For example, sentence label data 545 may be sent to an objective function component of training layer 502 after it is generated by classification model. The sentence label data 545 may be compared to a ground-truth dataset, such as ground-truth sentence label data 555 to determined differences between the two datasets. The objective function 550 will determine a difference between the data and use the difference to retrain one or more deep learning models of extraction layer 501. For example, contextual model 535 may generate contextual data based on the sentence embedding data and classification model 540 may generate sentence label data 545 based on the contextual data. This data will be compared to corresponding ground-truth data in ground-truth sentence label data 555 to determine an objective function to alter the operational parameters and weights of contextual model 535 and classification model 540.

[0120] FIG. 6 is a simplified block diagram of a computing system for generating inferences by a machine learning model according to various embodiments. The block diagram depicted in FIG. 6 may be implemented in software (e.g., code, instructions, program) executed by one or more processing units (e.g., processors, cores) of the respective systems, hardware, or combinations thereof. The software may be stored on a non-transitory storage medium (e.g., on a memory device). The flow of data and components presented in FIG. 6 and described below is intended to be illustrative and non-limiting. Although FIG. 6 depicts the various steps occurring in a particular sequence or order, this is not intended to be limiting. In certain alternative embodiments, the steps may be performed in some different order or some steps may also be performed in parallel.

[0121] As depicted in FIG. 6, system 600 is a system for utilizing raw text document data to generate question and answer pair data. System 600 may include an extraction layer 601 and an inference layer 602. In various embodiments, system 600 is an inference subsystem 420, extraction layer 601 is part of extraction subsystem 422, and inference layer 602 is part of data generation subsystem 424. In various embodiments, system 600 operates as part of a sentence labelling model configured to generate one or more labels corresponding to sentences extracted from a digital document containing text and generate training data for chatbot models using QA data.

[0122] Extraction layer 601 of FIG. 6 may be similar to extraction layer 501 depicted in FIG. 5. For example, extraction layer 601 may be a previously trained version of the extraction layer 501, in which deep learning models such as contextual model 535 and classification model 540 were previously trained according to an objective function 550. Extraction layer 601 includes the same components 505-545 for extraction of sentence label data 545 from raw text document data 505. Sentence label data 545 may then be sent to a decoder/postprocesser 610 of inference layer 602 to transformation the sentence label data into question/answer pairs 620. For example, decoder/postprocesser 610 may be a subsystem configured to combine the sentence label data 545 include one or more sentences and one or more predicted labels for the one or more sentences into training data in a particular format the is acceptable by a chatbot model for training. In this example, the chatbot model may receive the one or more sentences in an associative format with the corresponding predicted label as a ground truth label for the one or more sentences. The chatbot model may generate a predicted intent for a sentence and the predicted intent is compared to the label associated with the sentence to form an objective function that will be used to retrain the chatbot model.

[0123] FIG. 7 is a simplified block diagram of a computing system for feature extractions of text groupings according to various embodiments. The block diagram depicted in FIG. 7 may be implemented in software (e.g., code, instructions, program) executed by one or more processing units (e.g., processors, cores) of the respective systems, hardware, or combinations thereof. The software may be stored on a non-transitory storage medium (e.g., on a memory device). The flow of data and components presented in FIG. 7 and described below is intended to be illustrative and non-limiting. Although FIG. 7 depicts the various steps occurring in a particular sequence or order, this is not intended to be limiting. In certain alternative embodiments, the steps may be performed in some different order or some steps may also be performed in parallel.

[0124] FIG. 7 depicts an example embodiment of feature extractor 525 operating as part of an extraction layer such as extraction layer 501 or extraction layer 601. Specifically, FIG. 7 depicts a block diagram of components of feature extractor 525 which facilitate the transformation of sentence data 520 to sentence embedding data 530. As described herein raw text document data is desirable for training chatbot models. For example, raw text document data (such as FAQ documents) often relate to specific policies or procedures that are useful for a chatbot model to recognize to provide better chatbot services to users. However, the raw text document data is often in an unstructured format in which training data for a chatbot is not easily extracted. Furthermore, various unstructured document types exist and the information that can be extracted is highly diverse. Feature extractor 525 may be a subsystem of an extraction layer configured to transform discrete data including sentences into sequence embedding data in a format that is acceptable by a contextual deep learning model, such as contextual model 535, classification model 540, etc. The discrete data may be sentence data 520 received from a postprocessor 515, for example, in order to generate data for input to a deep-learning model.

[0125] As depicted in FIG. 7, sentence data 520 may be input to the feature extractor 525 where it is further processed. Sentence data 520 may include text data 700, box data 702(1)-702(N), font data 704 or any other data for facilitating the embodiments described herein in any combination or configuration. Feature extractor 525 may utilize any subset of the sentence data 520 to generate a final sentence embedding as sentence embedding data 530. Various components of feature extractor 525 may be utilized to generate the sentence embedding data 530.

[0126] Text data 700 may be data related to semantic aspects of sentence data 520. For example, text data 700 may be represented by listings of ASCII characters in an ordered sequence that corresponds to a natural language. Text data 700 may be parsed by a text embedding generator 710 to generate text embeddings 715. For example, a text embedding generator 710 may receive the text data 700 as input and parse the text data. The text embedding generator 710 may generate one or more sets of embedding data based on the text data that relates to aspects of the text data 700, such as a semantic meaning, a contextual meaning, a comparative meaning between two sentences, punctuation, natural language types, etc. An example of a text embedding generator 710 is the multilingual unsupervised and supervised embeddings library (MUSE) available at https://github.com/facebookresearch/MUSE. The resulting text embeddings 715 may be an embedding parsable by a deep learning model that is related to semantic aspects of the sentence data 520.

[0127] Box data 702(1)-702(N) may be data relative to data structures for representing character or sentence-based regions of the unstructured text document, such as bounding boxes. For example, box data 702(1)-702(N) may include dual-coordinate-point data for N characters or sentences in an unstructured text document. Box data 702(1)-(N) may be sent to box data embedding generator 720 to generate embeddings for the box data as box data embeddings 722(1)-722(N). Box data embeddings 722(1)-(N) may be data relating to corresponding to embeddings of the bounding box data represented by box data 702(1)-(N). Each of the separate box data embeddings 722(1)-(N) may be sent to a layout aggregator 726. Layout aggregator 726 may be configured to utilize the received box data embeddings 722(1)-(N) to generate layout embeddings 726. Layout embeddings 726 may be data relating to the box data embeddings 722(1)-(N) in a consolidated format includes data for each of the bounding boxes in the unstructured text document. In various embodiments, layout aggregator 724 is configured to generate continuous embeddings of box data embeddings via an aggregated long short-term memory (LSTM) artificial neural network. The resulting layout embedding has includes a consolidated set of bounding box data relating to all bounding box data in the unstructured text document.

[0128] In various embodiments, layout information such as layout embeddings 726 is generated based on relative horizontal/vertical distances between the bounding boxes of two characters/sentences. For example, layout aggregator 724 may be further configured to determine horizontal, vertical, and edge differences between two bounding boxes. For

example, the horizontal/vertical distance between two bounding boxes is zero if the two bounding boxes are around two sentences that intersect with each other in the unstructured text document. The differences calculated may be included as part of layout embeddings 726. In some embodiments, a lookup embedding table is utilized in combination with the differences between boxes to determine a difference that is used as part of the layout embeddings. In some embodiments, distances/differences between the boxes can be directly encoded using mathematical formulas. For example, the formulas below illustrate example formulas for finding the difference in distance between two bounding boxes. Examples of these formulas are shown below in Appendix A.

[0129] Box data 702(1)-(N) may be further utilized to generate a set of visual embeddings to be included in the sentence embedding data 530. For example, as depicted in FIG. 7, a document image 730 may be included in the feature extractor 525. The document image 730 may correspond to digital image data for representing the unstructured text document in a visual format. The document image 730 may be passed to a visual backbone 732. Visual backbone 732 may be a system or mapping configured to determine a relevant layout of the document image 730 for generating a visual embedding. For example, if document image 730 is a PDF image, the visual backbone 732 may determine one or more relative sections of the PDF file, such as a header section, a footer section, boundaries of the document, paragraphs, images, tables, lists, etc. The information derived by the visual backbone 732 may be input, along with the box data 702(1)-(N) to a region of interest alignment subsystem 734. Region of interest alignment subsystem 734 may be a subsystem of feature extractor 525 configured to utilize box data 702(1)-(N) and visual data from visual backbone 732 to determine one or more regions of interest in the document image 730. For example, the region of interest alignment subsystem 734 may be configured to generate region of interest data relating to coordinate-based representations of significant or relatively significant portions of the document image 730.

[0130] Region of interest data from region of interest alignment subsystem 734 may be sent to visual data aggregator 736 in order to generate visual embeddings 738. Visual embeddings 738 may relate to embeddings of data related to relevant portions of document image 730 that will be used to generate sentence embedding data 530. In some embodiments, visual data aggregator 736 includes a neural network model to take, as input, the region of interest data from region of interest alignment subsystem 734 and output embedded features of a figure, such as embedded regions of interest, relatively crowded or empty areas of a

document, document features such as headers, footers, margins, symbols, icons, and/or marks in the document image 730.

[0131] Font data 704 of sentence data 520 may also be used to generate font embeddings 740. For example, font data 704 may be processed by a machine learning model configured to take, as input, metadata related to fonts of characters/sentences in the sentence data 520 and output font embeddings 740 relating to a relative difference between fonts throughout the unstructured text document. Once text embeddings 715, layout embeddings 726, visual embeddings 738, and font embeddings 740 or some subset thereof are generated, an embedding aggregator 750 may aggregate the embedding data and utilize it to generate sentence embedding data. For example, embedding aggregator may consolidate, modify, add, remove, or otherwise transform the embeddings received into a final set of sentence embedding data 530 in a format acceptable by a deep-learning. The sentence embedding data 530 may, for example, represent aspects of each of the text, bounding boxes, and font of some sentence data in relative proportions or weights sufficient for input to a deep learning model to generate predicted classification data for the sentences. Other examples of sentence properties that may be included in the final sentence embedding data include font properties, text styles, line stroke types, stroke colors, non-stroke colors, etc.

Example Feature Extraction Processes

[0132] FIG. 8 illustrates a process flow for generating text grouping embeddings for machine learning processes according to various embodiments. The processing depicted in FIG. 6 may be implemented in software (e.g., code, instructions, program) executed by one or more processing units (e.g., processors, cores) of the respective systems, hardware, or combinations thereof. The software may be stored on a non-transitory storage medium (e.g., on a memory device). The method presented in FIG. 8 and described below is intended to be illustrative and non-limiting. Although FIG. 8 depicts the various processing steps occurring in a particular sequence or order, this is not intended to be limiting. In certain alternative embodiments, the steps may be performed in some different order or some steps may also be performed in parallel.

[0133] At 810, a set of unstructured text data is received. Unstructured text data may be a digital document or file that contains character based text in an unspecified format. For example, as opposed to structured text document which may contain metadata corresponding a format of text in the structured text document, an unstructured text document may contain

no such metadata or may not be structured in a format. The unstructured text data received may be document data including some text presented in a natural language from which embeddings may be extracted to generate classification data for sentences of the unstructured text data. In some embodiments, the set of unstructured text data includes one or more documents, such as PDF files. The one or more documents may be received at an extraction subsystem configured to extract classification data from the unstructured text data as part of a process for retraining the deep learning model used to generate the classification data or to generate training data for a chatbot model.

[0134] At 820, the unstructured text data received in 810 is processed to generate one or more text groupings. The one or more text groupings may be some relative grouping of text or characters according to a natural language. For example, the one or more text groupings may be one or more sentences corresponding to semantic and grammatical groupings of text that make a proper language statement based on natural language rules.

[0135] At 830, one or more text grouping embeddings are generated based on the one or more text groupings generated in 820. The one or more text grouping embeddings are generated as embeddings that may be accepted by a deep learning model for generating predicted aspects of the one or more text groupings. For example, the one or more text groupings may be parsed by an encoder model configured to intake the one or more text groupings and output an encoded embedding of the one or text groupings that are in an input format acceptable to a deep learning model.

[0136] At 840, the one or more text grouping embeddings are input to a machine learning model to generate one or more output predictions corresponding to predicted aspects of the one or more text groupings. The one or more output predictions may be predictions for a corresponding classification of the one or more text groupings. For example, the one or more text groupings may be sentences and the one or more output predictions may correspond to predicted classifications of aspects of the sentences, such as a designation of a sentence as single, or compound, a context of a sentence such as a question or a statement, a relative position of the sentence is a multi-question or multi-statement set of sentences. etc.

Illustrative Systems

[0137] FIG. 9 depicts a simplified diagram of a distributed system 900. In the illustrated example, distributed system 900 includes one or more client computing devices 902, 904, 906, and 908, coupled to a server 912 via one or more communication networks 910. Clients

computing devices 902, 904, 906, and 908 may be configured to execute one or more applications.

[0138] In various examples, server 912 may be adapted to run one or more services or software applications that enable one or more embodiments described in this disclosure. In certain examples, server 912 may also provide other services or software applications that may include non-virtual and virtual environments. In some examples, these services may be offered as web-based or cloud services, such as under a Software as a Service (SaaS) model to the users of client computing devices 902, 904, 906, and/or 908. Users operating client computing devices 902, 904, 906, and/or 908 may in turn utilize one or more client applications to interact with server 912 to utilize the services provided by these components.

[0139] In the configuration depicted in FIG. 9, server 912 may include one or more components 918, 920 and 922 that implement the functions performed by server 912. These components may include software components that may be executed by one or more processors, hardware components, or combinations thereof. It should be appreciated that various different system configurations are possible, which may be different from distributed system 900. The example shown in FIG. 9 is thus one example of a distributed system for implementing an example system and is not intended to be limiting.

[0140] Users may use client computing devices 902, 904, 906, and/or 908 to execute one or more applications, models or chatbots, which may generate one or more events or models that may then be implemented or serviced in accordance with the teachings of this disclosure. A client device may provide an interface that enables a user of the client device to interact with the client device. The client device may also output information to the user via this interface. Although FIG. 9 depicts only four client computing devices, any number of client computing devices may be supported.

[0141] The client devices may include various types of computing systems such as portable handheld devices, general purpose computers such as personal computers and laptops, workstation computers, wearable devices, gaming systems, thin clients, various messaging devices, sensors or other sensing devices, and the like. These computing devices may run various types and versions of software applications and operating systems (e.g., Microsoft Windows®, Apple Macintosh®, UNIX® or UNIX-like operating systems, Linux or Linux-like operating systems such as Google Chrome™ OS) including various mobile operating systems (e.g., Microsoft Windows Mobile®, iOS®, Windows Phone®, Android™,

BlackBerry®, Palm OS®). Portable handheld devices may include cellular phones, smartphones, (e.g., an iPhone®), tablets (e.g., iPad®), personal digital assistants (PDAs), and the like. Wearable devices may include Google Glass® head mounted display, and other devices. Gaming systems may include various handheld gaming devices, Internet-enabled gaming devices (e.g., a Microsoft Xbox® gaming console with or without a Kinect® gesture input device, Sony PlayStation® system, various gaming systems provided by Nintendo®, and others), and the like. The client devices may be capable of executing various different applications such as various Internet-related apps, communication applications (e.g., E-mail applications, short message service (SMS) applications) and may use various communication protocols.

[0142] Network(s) 910 may be any type of network familiar to those skilled in the art that may support data communications using any of a variety of available protocols, including without limitation TCP/IP (transmission control protocol/Internet protocol), SNA (systems network architecture), IPX (Internet packet exchange), AppleTalk®, and the like. Merely by way of example, network(s) 910 may be a local area network (LAN), networks based on Ethernet, Token-Ring, a wide-area network (WAN), the Internet, a virtual network, a virtual private network (VPN), an intranet, an extranet, a public switched telephone network (PSTN), an infra-red network, a wireless network (e.g., a network operating under any of the Institute of Electrical and Electronics (IEEE) 1002.11 suite of protocols, Bluetooth®, and/or any other wireless protocol), and/or any combination of these and/or other networks.

[0143] Server 912 may be composed of one or more general purpose computers, specialized server computers (including, by way of example, PC (personal computer) servers, UNIX® servers, mid-range servers, mainframe computers, rack-mounted servers, etc.), server farms, server clusters, or any other appropriate arrangement and/or combination. Server 912 may include one or more virtual machines running virtual operating systems, or other computing architectures involving virtualization such as one or more flexible pools of logical storage devices that may be virtualized to maintain virtual storage devices for the server. In various examples, server 912 may be adapted to run one or more services or software applications that provide the functionality described in the foregoing disclosure.

[0144] The computing systems in server 912 may run one or more operating systems including any of those discussed above, as well as any commercially available server operating system. Server 912 may also run any of a variety of additional server applications

and/or mid-tier applications, including HTTP (hypertext transport protocol) servers, FTP (file transfer protocol) servers, CGI (common gateway interface) servers, JAVA® servers, database servers, and the like. Exemplary database servers include without limitation those commercially available from Oracle®, Microsoft®, Sybase®, IBM® (International Business Machines), and the like.

[0145] In some implementations, server 912 may include one or more applications to analyze and consolidate data feeds and/or event updates received from users of client computing devices 902, 904, 906, and 908. As an example, data feeds and/or event updates may include, but are not limited to, Twitter® feeds, Facebook® updates or real-time updates received from one or more third party information sources and continuous data streams, which may include real-time events related to sensor data applications, financial tickers, network performance measuring tools (e.g., network monitoring and traffic management applications), clickstream analysis tools, automobile traffic monitoring, and the like. Server 912 may also include one or more applications to display the data feeds and/or real-time events via one or more display devices of client computing devices 902, 904, 906, and 908.

[0146] Distributed system 900 may also include one or more data repositories 914, 916. These data repositories may be used to store data and other information in certain examples. For example, one or more of the data repositories 914, 916 may be used to store information such as information related to chatbot performance or generated models for use by chatbots used by server 912 when performing various functions in accordance with various embodiments. Data repositories 914, 916 may reside in a variety of locations. For example, a data repository used by server 912 may be local to server 912 or may be remote from server 912 and in communication with server 912 via a network-based or dedicated connection. Data repositories 914, 916 may be of different types. In certain examples, a data repository used by server 912 may be a database, for example, a relational database, such as databases provided by Oracle Corporation® and other vendors. One or more of these databases may be adapted to enable storage, update, and retrieval of data to and from the database in response to SQL-formatted commands.

[0147] In certain examples, one or more of data repositories 914, 916 may also be used by applications to store application data. The data repositories used by applications may be of different types such as, for example, a key-value store repository, an object store repository, or a general storage repository supported by a file system.

[0148] In certain examples, the functionalities described in this disclosure may be offered as services via a cloud environment. FIG. 10 is a simplified block diagram of a cloud-based system environment in which various services may be offered as cloud services in accordance with certain examples. In the example depicted in FIG. 10, cloud infrastructure system 1002 may provide one or more cloud services that may be requested by users using one or more client computing devices 1004, 1006, and 1008. Cloud infrastructure system 1002 may comprise one or more computers and/or servers that may include those described above for server 712. The computers in cloud infrastructure system 1002 may be organized as general purpose computers, specialized server computers, server farms, server clusters, or any other appropriate arrangement and/or combination.

[0149] Network(s) 1010 may facilitate communication and exchange of data between clients 1004, 1006, and 1008 and cloud infrastructure system 1002. Network(s) 1010 may include one or more networks. The networks may be of the same or different types. Network(s) 1010 may support one or more communication protocols, including wired and/or wireless protocols, for facilitating the communications.

[0150] The example depicted in FIG. 10 is only one example of a cloud infrastructure system and is not intended to be limiting. It should be appreciated that, in some other examples, cloud infrastructure system 1002 may have more or fewer components than those depicted in FIG. 10, may combine two or more components, or may have a different configuration or arrangement of components. For example, although FIG. 10 depicts three client computing devices, any number of client computing devices may be supported in alternative examples.

[0151] The term cloud service is generally used to refer to a service that is made available to users on demand and via a communication network such as the Internet by systems (e.g., cloud infrastructure system 1002) of a service provider. Typically, in a public cloud environment, servers and systems that make up the cloud service provider's system are different from the customer's own on-premise servers and systems. The cloud service provider's systems are managed by the cloud service provider. Customers may thus avail themselves of cloud services provided by a cloud service provider without having to purchase separate licenses, support, or hardware and software resources for the services. For example, a cloud service provider's system may host an application, and a user may, via the Internet, on demand, order and use the application without the user having to buy infrastructure resources

for executing the application. Cloud services are designed to provide easy, scalable access to applications, resources and services. Several providers offer cloud services. For example, several cloud services are offered by Oracle Corporation® of Redwood Shores, California, such as middleware services, database services, Java cloud services, and others.

[0152] In certain examples, cloud infrastructure system 1002 may provide one or more cloud services using different models such as under a Software as a Service (SaaS) model, a Platform as a Service (PaaS) model, an Infrastructure as a Service (IaaS) model, and others, including hybrid service models. Cloud infrastructure system 1002 may include a suite of applications, middleware, databases, and other resources that enable provision of the various cloud services.

[0153] A SaaS model enables an application or software to be delivered to a customer over a communication network like the Internet, as a service, without the customer having to buy the hardware or software for the underlying application. For example, a SaaS model may be used to provide customers access to on-demand applications that are hosted by cloud infrastructure system 1002. Examples of SaaS services provided by Oracle Corporation® include, without limitation, various services for human resources/capital management, customer relationship management (CRM), enterprise resource planning (ERP), supply chain management (SCM), enterprise performance management (EPM), analytics services, social applications, and others.

[0154] An IaaS model is generally used to provide infrastructure resources (e.g., servers, storage, hardware and networking resources) to a customer as a cloud service to provide elastic compute and storage capabilities. Various IaaS services are provided by Oracle Corporation®.

[0155] A PaaS model is generally used to provide, as a service, platform and environment resources that enable customers to develop, run, and manage applications and services without the customer having to procure, build, or maintain such resources. Examples of PaaS services provided by Oracle Corporation® include, without limitation, Oracle Java Cloud Service (JCS), Oracle Database Cloud Service (DBCS), data management cloud service, various application development solutions services, and others.

[0156] Cloud services are generally provided on an on-demand self-service basis, subscription-based, elastically scalable, reliable, highly available, and secure manner. For example, a customer, via a subscription order, may order one or more services provided by

cloud infrastructure system 1002. Cloud infrastructure system 1002 then performs processing to provide the services requested in the customer's subscription order. For example, a user may use utterances to request the cloud infrastructure system to take a certain action (e.g., an intent), as described above, and/or provide services for a chatbot system as described herein. Cloud infrastructure system 1002 may be configured to provide one or even multiple cloud services.

[0157] Cloud infrastructure system 1002 may provide the cloud services via different deployment models. In a public cloud model, cloud infrastructure system 1002 may be owned by a third party cloud services provider and the cloud services are offered to any general public customer, where the customer may be an individual or an enterprise. In certain other examples, under a private cloud model, cloud infrastructure system 1002 may be operated within an organization (e.g., within an enterprise organization) and services provided to customers that are within the organization. For example, the customers may be various departments of an enterprise such as the Human Resources department, the Payroll department, etc. or even individuals within the enterprise. In certain other examples, under a community cloud model, the cloud infrastructure system 1002 and the services provided may be shared by several organizations in a related community. Various other models such as hybrids of the above mentioned models may also be used.

[0158] Client computing devices 1004, 1006, and 1008 may be of different types (such as client computing devices 902, 904, 906, and 908 depicted in FIG. 9) and may be capable of operating one or more client applications. A user may use a client device to interact with cloud infrastructure system 1002, such as to request a service provided by cloud infrastructure system 1002. For example, a user may use a client device to request information or action from a chatbot as described in this disclosure.

[0159] In some examples, the processing performed by cloud infrastructure system 1002 for providing services may involve model training and deployment. This analysis may involve using, analyzing, and manipulating data sets to train and deploy one or more models. This analysis may be performed by one or more processors, possibly processing the data in parallel, performing simulations using the data, and the like. For example, big data analysis may be performed by cloud infrastructure system 1002 for generating and training one or more models for a chatbot system. The data used for this analysis may include structured data

(e.g., data stored in a database or structured according to a structured model) and/or unstructured data (e.g., data blobs (binary large objects)).

[0160] As depicted in the example in FIG. 10, cloud infrastructure system 1002 may include infrastructure resources 1030 that are utilized for facilitating the provision of various cloud services offered by cloud infrastructure system 1002. Infrastructure resources 1030 may include, for example, processing resources, storage or memory resources, networking resources, and the like. In certain examples, the storage virtual machines that are available for servicing storage requested from applications may be part of cloud infrastructure system 1002. In other examples, the storage virtual machines may be part of different systems.

[0161] In certain examples, to facilitate efficient provisioning of these resources for supporting the various cloud services provided by cloud infrastructure system 1002 for different customers, the resources may be bundled into sets of resources or resource modules (also referred to as "pods"). Each resource module or pod may comprise a pre-integrated and optimized combination of resources of one or more types. In certain examples, different pods may be pre-provisioned for different types of cloud services. For example, a first set of pods may be provisioned for a database service, a second set of pods, which may include a different combination of resources than a pod in the first set of pods, may be provisioned for Java service, and the like. For some services, the resources allocated for provisioning the services may be shared between the services.

[0162] Cloud infrastructure system 1002 may itself internally use services 1032 that are shared by different components of cloud infrastructure system 1002 and which facilitate the provisioning of services by cloud infrastructure system 1002. These internal shared services may include, without limitation, a security and identity service, an integration service, an enterprise repository service, an enterprise manager service, a virus scanning and white list service, a high availability, backup and recovery service, service for enabling cloud support, an email service, a notification service, a file transfer service, and the like.

[0163] Cloud infrastructure system 1002 may comprise multiple subsystems. These subsystems may be implemented in software, or hardware, or combinations thereof. As depicted in FIG. 10, the subsystems may include a user interface subsystem 1012 that enables users or customers of cloud infrastructure system 1002 to interact with cloud infrastructure system 1002. User interface subsystem 1012 may include various different interfaces such as a web interface 1014, an online store interface 1016 where cloud services provided by cloud

infrastructure system 1002 are advertised and are purchasable by a consumer, and other interfaces 1018. For example, a customer may, using a client device, request (service request 1034) one or more services provided by cloud infrastructure system 1002 using one or more of interfaces 1014, 1016, and 1018. For example, a customer may access the online store, browse cloud services offered by cloud infrastructure system 1002, and place a subscription order for one or more services offered by cloud infrastructure system 1002 that the customer wishes to subscribe to. The service request may include information identifying the customer and one or more services that the customer desires to subscribe to. For example, a customer may place a subscription order for a service offered by cloud infrastructure system 1002. As part of the order, the customer may provide information identifying a chatbot system for which the service is to be provided and optionally one or more credentials for the chatbot system.

[0164] In certain examples, such as the example depicted in FIG. 10, cloud infrastructure system 1002 may comprise an order management subsystem (OMS) 1020 that is configured to process the new order. As part of this processing, OMS 1020 may be configured to: create an account for the customer, if not done already; receive billing and/or accounting information from the customer that is to be used for billing the customer for providing the requested service to the customer; verify the customer information; upon verification, book the order for the customer; and orchestrate various workflows to prepare the order for provisioning.

[0165] Once properly validated, OMS 1020 may then invoke the order provisioning subsystem (OPS) 1024 that is configured to provision resources for the order including processing, memory, and networking resources. The provisioning may include allocating resources for the order and configuring the resources to facilitate the service requested by the customer order. The manner in which resources are provisioned for an order and the type of the provisioned resources may depend upon the type of cloud service that has been ordered by the customer. For example, according to one workflow, OPS 1024 may be configured to determine the particular cloud service being requested and identify a number of pods that may have been pre-configured for that particular cloud service. The number of pods that are allocated for an order may depend upon the size/amount/level/scope of the requested service. For example, the number of pods to be allocated may be determined based upon the number of users to be supported by the service, the duration of time for which the service is being

requested, and the like. The allocated pods may then be customized for the particular requesting customer for providing the requested service.

[0166] In certain examples, setup phase processing, as described above, may be performed by cloud infrastructure system 1002 as part of the provisioning process. Cloud infrastructure system 1002 may generate an application ID and select a storage virtual machine for an application from among storage virtual machines provided by cloud infrastructure system 1002 itself or from storage virtual machines provided by other systems other than cloud infrastructure system 1002.

[0167] Cloud infrastructure system 1002 may send a response or notification 1044 to the requesting customer to indicate when the requested service is now ready for use. In some instances, information (e.g., a link) may be sent to the customer that enables the customer to start using and availing the benefits of the requested services. In certain examples, for a customer requesting the service, the response may include a chatbot system ID generated by cloud infrastructure system 1002 and information identifying a chatbot system selected by cloud infrastructure system 1002 for the chatbot system corresponding to the chatbot system ID.

[0168] Cloud infrastructure system 1002 may provide services to multiple customers. For each customer, cloud infrastructure system 1002 is responsible for managing information related to one or more subscription orders received from the customer, maintaining customer data related to the orders, and providing the requested services to the customer. Cloud infrastructure system 1002 may also collect usage statistics regarding a customer's use of subscribed services. For example, statistics may be collected for the amount of storage used, the amount of data transferred, the number of users, and the amount of system up time and system down time, and the like. This usage information may be used to bill the customer. Billing may be done, for example, on a monthly cycle.

[0169] Cloud infrastructure system 1002 may provide services to multiple customers in parallel. Cloud infrastructure system 1002 may store information for these customers, including possibly proprietary information. In certain examples, cloud infrastructure system 1002 comprises an identity management subsystem (IMS) 1028 that is configured to manage customer information and provide the separation of the managed information such that information related to one customer is not accessible by another customer. IMS 1028 may be configured to provide various security-related services such as identity services, such as

information access management, authentication and authorization services, services for managing customer identities and roles and related capabilities, and the like.

[0170] FIG. 11 illustrates an example of computer system 1100. In some examples, computer system 1100 may be used to implement any of the digital assistant or chatbot systems within a distributed environment, and various servers and computer systems described above. As shown in FIG. 11, computer system 1100 includes various subsystems including a processing subsystem 1104 that communicates with a number of other subsystems via a bus subsystem 1102. These other subsystems may include a processing acceleration unit 1106, an I/O subsystem 1108, a storage subsystem 1118, and a communications subsystem 1124. Storage subsystem 1118 may include non-transitory computer-readable storage media including storage media 1122 and a system memory 1110.

[0171] Bus subsystem 1102 provides a mechanism for letting the various components and subsystems of computer system 1100 communicate with each other as intended. Although bus subsystem 1102 is shown schematically as a single bus, alternative examples of the bus subsystem may utilize multiple buses. Bus subsystem 1102 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, a local bus using any of a variety of bus architectures, and the like. For example, such architectures may include an Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus, which may be implemented as a Mezzanine bus manufactured to the IEEE P1386.1 standard, and the like.

[0172] Processing subsystem 1104 controls the operation of computer system 1100 and may comprise one or more processors, application specific integrated circuits (ASICs), or field programmable gate arrays (FPGAs). The processors may include be single core or multicore processors. The processing resources of computer system 1100 may be organized into one or more processing units 1132, 1134, etc. A processing unit may include one or more processors, one or more cores from the same or different processors, a combination of cores and processors, or other combinations of cores and processors. In some examples, processing subsystem 1104 may include one or more special purpose co-processors such as graphics processors, digital signal processors (DSPs), or the like. In some examples, some or all of the processing units of processing subsystem 1104 may be implemented using customized

circuits, such as application specific integrated circuits (ASICs), or field programmable gate arrays (FPGAs).

[0173] In some examples, the processing units in processing subsystem 1104 may execute instructions stored in system memory 1110 or on computer readable storage media 1122. In various examples, the processing units may execute a variety of programs or code instructions and may maintain multiple concurrently executing programs or processes. At any given time, some or all of the program code to be executed may be resident in system memory 1110 and/or on computer-readable storage media 1122 including potentially on one or more storage devices. Through suitable programming, processing subsystem 1104 may provide various functionalities described above. In instances where computer system 1100 is executing one or more virtual machines, one or more processing units may be allocated to each virtual machine.

[0174] In certain examples, a processing acceleration unit 1106 may optionally be provided for performing customized processing or for off-loading some of the processing performed by processing subsystem 1104 so as to accelerate the overall processing performed by computer system 1100.

[0175] I/O subsystem 1108 may include devices and mechanisms for inputting information to computer system 1100 and/or for outputting information from or via computer system 1100. In general, use of the term input device is intended to include all possible types of devices and mechanisms for inputting information to computer system 1100. User interface input devices may include, for example, a keyboard, pointing devices such as a mouse or trackball, a touchpad or touch screen incorporated into a display, a scroll wheel, a click wheel, a dial, a button, a switch, a keypad, audio input devices with voice command recognition systems, microphones, and other types of input devices. User interface input devices may also include motion sensing and/or gesture recognition devices such as the Microsoft Kinect® motion sensor that enables users to control and interact with an input device, the Microsoft Xbox® 360 game controller, devices that provide an interface for receiving input using gestures and spoken commands. User interface input devices may also include eye gesture recognition devices such as the Google Glass® blink detector that detects eye activity (e.g., "blinking" while taking pictures and/or making a menu selection) from users and transforms the eye gestures as inputs to an input device (e.g., Google Glass®). Additionally, user interface input devices may include voice recognition sensing devices that

enable users to interact with voice recognition systems (e.g., Siri® navigator) through voice commands.

[0176] Other examples of user interface input devices include, without limitation, three dimensional (3D) mice, joysticks or pointing sticks, gamepads and graphic tablets, and audio/visual devices such as speakers, digital cameras, digital camcorders, portable media players, webcams, image scanners, fingerprint scanners, barcode reader 3D scanners, 3D printers, laser rangefinders, and eye gaze tracking devices. Additionally, user interface input devices may include, for example, medical imaging input devices such as computed tomography, magnetic resonance imaging, position emission tomography, and medical ultrasonography devices. User interface input devices may also include, for example, audio input devices such as MIDI keyboards, digital musical instruments and the like.

[0177] In general, use of the term output device is intended to include all possible types of devices and mechanisms for outputting information from computer system 1100 to a user or other computer. User interface output devices may include a display subsystem, indicator lights, or non-visual displays such as audio output devices, etc. The display subsystem may be a cathode ray tube (CRT), a flat-panel device, such as that using a liquid crystal display (LCD) or plasma display, a projection device, a touch screen, and the like. For example, user interface output devices may include, without limitation, a variety of display devices that visually convey text, graphics and audio/video information such as monitors, printers, speakers, headphones, automotive navigation systems, plotters, voice output devices, and modems.

[0178] Storage subsystem 1118 provides a repository or data store for storing information and data that is used by computer system 1100. Storage subsystem 1118 provides a tangible non-transitory computer-readable storage medium for storing the basic programming and data constructs that provide the functionality of some examples. Storage subsystem 1118 may store software (e.g., programs, code modules, instructions) that when executed by processing subsystem 1104 provides the functionality described above. The software may be executed by one or more processing units of processing subsystem 1104. Storage subsystem 1118 may also provide authentication in accordance with the teachings of this disclosure.

[0179] Storage subsystem 1118 may include one or more non-transitory memory devices, including volatile and non-volatile memory devices. As shown in FIG. 11, storage subsystem 1118 includes a system memory 1110 and a computer-readable storage media 1122. System

memory 1110 may include a number of memories including a volatile main random access memory (RAM) for storage of instructions and data during program execution and a non-volatile read only memory (ROM) or flash memory in which fixed instructions are stored. In some implementations, a basic input/output system (BIOS), containing the basic routines that help to transfer information between elements within computer system 1100, such as during start-up, may typically be stored in the ROM. The RAM typically contains data and/or program modules that are presently being operated and executed by processing subsystem 1104. In some implementations, system memory 1110 may include multiple different types of memory, such as static random access memory (SRAM), dynamic random access memory (DRAM), and the like.

[0180] By way of example, and not limitation, as depicted in FIG. 11, system memory 1110 may load application programs 1112 that are being executed, which may include various applications such as Web browsers, mid-tier applications, relational database management systems (RDBMS), etc., program data 1114, and an operating system 1116. By way of example, operating system 1116 may include various versions of Microsoft Windows®, Apple Macintosh®, and/or Linux operating systems, a variety of commercially-available UNIX® or UNIX-like operating systems (including without limitation the variety of GNU/Linux operating systems, the Google Chrome® OS, and the like) and/or mobile operating systems such as iOS, Windows® Phone, Android® OS, BlackBerry® OS, Palm® OS operating systems, and others.

[0181] Computer-readable storage media 1122 may store programming and data constructs that provide the functionality of some examples. Computer-readable media 1122 may provide storage of computer-readable instructions, data structures, program modules, and other data for computer system 1100. Software (programs, code modules, instructions) that, when executed by processing subsystem 1104 provides the functionality described above, may be stored in storage subsystem 1118. By way of example, computer-readable storage media 1122 may include non-volatile memory such as a hard disk drive, a magnetic disk drive, an optical disk drive such as a CD ROM, DVD, a Blu-Ray® disk, or other optical media. Computer-readable storage media 1122 may include, but is not limited to, Zip® drives, flash memory cards, universal serial bus (USB) flash drives, secure digital (SD) cards, DVD disks, digital video tape, and the like. Computer-readable storage media 1122 may also include, solid-state drives (SSD) based on non-volatile memory such as flash-memory based SSDs, enterprise flash drives, solid state ROM, and the like, SSDs based on volatile memory

such as solid state RAM, dynamic RAM, static RAM, DRAM-based SSDs, magnetoresistive RAM (MRAM) SSDs, and hybrid SSDs that use a combination of DRAM and flash memory based SSDs.

[0182] In certain examples, storage subsystem 1118 may also include a computer-readable storage media reader 1120 that may further be connected to computer-readable storage media 1122. Reader 1120 may receive and be configured to read data from a memory device such as a disk, a flash drive, etc.

[0183] In certain examples, computer system 1100 may support virtualization technologies, including but not limited to virtualization of processing and memory resources. For example, computer system 1100 may provide support for executing one or more virtual machines. In certain examples, computer system 1100 may execute a program such as a hypervisor that facilitated the configuring and managing of the virtual machines. Each virtual machine may be allocated memory, compute (e.g., processors, cores), I/O, and networking resources. Each virtual machine generally runs independently of the other virtual machines. A virtual machine typically runs its own operating system, which may be the same as or different from the operating systems executed by other virtual machines executed by computer system 1100. Accordingly, multiple operating systems may potentially be run concurrently by computer system 1100.

[0184] Communications subsystem 1124 provides an interface to other computer systems and networks. Communications subsystem 1124 serves as an interface for receiving data from and transmitting data to other systems from computer system 1100. For example, communications subsystem 1124 may enable computer system 1100 to establish a communication channel to one or more client devices via the Internet for receiving and sending information from and to the client devices. For example, when computer system 1100 is used to implement bot system 120 depicted in FIG. 1, the communication subsystem may be used to communicate with a chatbot system selected for an application.

[0185] Communication subsystem 1124 may support both wired and/or wireless communication protocols. In certain examples, communications subsystem 1124 may include radio frequency (RF) transceiver components for accessing wireless voice and/or data networks (e.g., using cellular telephone technology, advanced data network technology, such as 3G, 4G or EDGE (enhanced data rates for global evolution), WiFi (IEEE 802.XX family standards, or other mobile communication technologies, or any combination thereof), global

positioning system (GPS) receiver components, and/or other components. In some examples, communications subsystem 1124 may provide wired network connectivity (e.g., Ethernet) in addition to or instead of a wireless interface.

[0186] Communication subsystem 1124 may receive and transmit data in various forms. In some examples, in addition to other forms, communications subsystem 1124 may receive input communications in the form of structured and/or unstructured data feeds 1126, event streams 1128, event updates 1130, and the like. For example, communications subsystem 1124 may be configured to receive (or send) data feeds 1126 in real-time from users of social media networks and/or other communication services such as Twitter® feeds, Facebook® updates, web feeds such as Rich Site Summary (RSS) feeds, and/or real-time updates from one or more third party information sources.

[0187] In certain examples, communications subsystem 1124 may be configured to receive data in the form of continuous data streams, which may include event streams 1128 of real-time events and/or event updates 1130, that may be continuous or unbounded in nature with no explicit end. Examples of applications that generate continuous data may include, for example, sensor data applications, financial tickers, network performance measuring tools (e.g. network monitoring and traffic management applications), clickstream analysis tools, automobile traffic monitoring, and the like.

[0188] Communications subsystem 1124 may also be configured to communicate data from computer system 1100 to other computer systems or networks. The data may be communicated in various different forms such as structured and/or unstructured data feeds 1126, event streams 1128, event updates 1130, and the like to one or more databases that may be in communication with one or more streaming data source computers coupled to computer system 1100.

[0189] Computer system 1100 may be one of various types, including a handheld portable device (e.g., an iPhone® cellular phone, an iPad® computing tablet, a PDA), a wearable device (e.g., a Google Glass® head mounted display), a personal computer, a workstation, a mainframe, a kiosk, a server rack, or any other data processing system. Due to the ever-changing nature of computers and networks, the description of computer system 1100 depicted in FIG. 11 is intended only as a specific example. Many other configurations having more or fewer components than the system depicted in FIG. 11 are possible. Based on

the disclosure and teachings provided herein, it should be appreciate there are other ways and/or methods to implement the various examples.

[0190] Although specific examples have been described, various modifications, alterations, alternative constructions, and equivalents are possible. Examples are not restricted to operation within certain specific data processing environments, but are free to operate within a plurality of data processing environments. Additionally, although certain examples have been described using a particular series of transactions and steps, it should be apparent to those skilled in the art that this is not intended to be limiting. Although some flowcharts describe operations as a sequential process, many of the operations may be performed in parallel or concurrently. In addition, the order of the operations may be rearranged. A process may have additional steps not included in the figure. Various features and aspects of the above-described examples may be used individually or jointly.

[0191] Further, while certain examples have been described using a particular combination of hardware and software, it should be recognized that other combinations of hardware and software are also possible. Certain examples may be implemented only in hardware, or only in software, or using combinations thereof. The various processes described herein may be implemented on the same processor or different processors in any combination.

[0192] Where devices, systems, components or modules are described as being configured to perform certain operations or functions, such configuration may be accomplished, for example, by designing electronic circuits to perform the operation, by programming programmable electronic circuits (such as microprocessors) to perform the operation such as by executing computer instructions or code, or processors or cores programmed to execute code or instructions stored on a non-transitory memory medium, or any combination thereof. Processes may communicate using a variety of techniques including but not limited to conventional techniques for inter-process communications, and different pairs of processes may use different techniques, or the same pair of processes may use different techniques at different times.

[0193] Specific details are given in this disclosure to provide a thorough understanding of the examples. However, examples may be practiced without these specific details. For example, well-known circuits, processes, algorithms, structures, and techniques have been shown without unnecessary detail in order to avoid obscuring the examples. This description

provides example examples only, and is not intended to limit the scope, applicability, or configuration of other examples. Rather, the preceding description of the examples will provide those skilled in the art with an enabling description for implementing various examples. Various changes may be made in the function and arrangement of elements.

[0194] The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense. It will, however, be evident that additions, subtractions, deletions, and other modifications and changes may be made thereunto without departing from the broader spirit and scope as set forth in the claims. Thus, although specific examples have been described, these are not intended to be limiting. Various modifications and equivalents are within the scope of the following claims.

[0195] In the foregoing specification, aspects of the disclosure are described with reference to specific examples thereof, but those skilled in the art will recognize that the disclosure is not limited thereto. Various features and aspects of the above-described disclosure may be used individually or jointly. Further, examples may be utilized in any number of environments and applications beyond those described herein without departing from the broader spirit and scope of the specification. The specification and drawings are, accordingly, to be regarded as illustrative rather than restrictive.

[0196] In the foregoing description, for the purposes of illustration, methods were described in a particular order. It should be appreciated that in alternate examples, the methods may be performed in a different order than that described. It should also be appreciated that the methods described above may be performed by hardware components or may be embodied in sequences of machine-executable instructions, which may be used to cause a machine, such as a general-purpose or special-purpose processor or logic circuits programmed with the instructions to perform the methods. These machine-executable instructions may be stored on one or more machine readable mediums, such as CD-ROMs or other type of optical disks, floppy diskettes, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, flash memory, or other types of machine-readable mediums suitable for storing electronic instructions. Alternatively, the methods may be performed by a combination of hardware and software.

[0197] Where components are described as being configured to perform certain operations, such configuration may be accomplished, for example, by designing electronic circuits or other hardware to perform the operation, by programming programmable

electronic circuits (e.g., microprocessors, or other suitable electronic circuits) to perform the operation, or any combination thereof.

[0198] While illustrative examples of the application have been described in detail herein, it is to be understood that the inventive concepts may be otherwise variously embodied and employed, and that the appended claims are intended to be construed to include such variations, except as limited by the prior art.

WHAT IS CLAIMED IS:

1. A method, comprising:

receiving, at a data processing system, a set of unstructured text data;

generating, by the data processing system, one or more text groupings by processing the set of unstructured text data;

generating, by the data processing system and based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model; and

generating, by the data processing system, one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model, each output prediction of the one or more output predictions corresponding to a predicted aspect of a text grouping of the one or more text groupings.

2. The method of claim 1, wherein the set of unstructured text data is one or more portable document format (PDF) text files.

3. The method of claim 1, wherein:

processing the set of unstructured text data comprises extracting, from the set of unstructured text data, one or more sets of character; and

generating the one or more text groupings comprises grouping the one or more sets of characters according to a relative position of each character in the set of unstructured text data.

4. The method of claim 1, wherein generating the one or more text grouping embeddings comprises generating a plurality of sub-embeddings based on the set of unstructured text data or the one or more text groupings and generating the one or more text groupings based on the plurality of sub-embeddings.

5. The method of claim 4, wherein a sub-embedding of the plurality of sub-embeddings is a text sub-embedding generated based on one or more semantic aspects of the one or more text groupings.

6. The method of claim 4, wherein a sub-embedding of the plurality of sub-embeddings is a bounding sub-embedding generated based on one or more extracted spatial bounds of characters in the set of unstructured text data.

7. The method of claim 4, wherein a sub-embedding of the plurality of sub-embeddings is a visual sub-embedding generated based on one or more extracted image-based aspects of the set of unstructured text data.

8. The method of claim 4, wherein a sub-embedding of the plurality of sub-embeddings is a relative font sub-embedding generated based on one or more different visual fonts of text in the set of unstructured text data.

9. The method of claim 1, wherein:

the one or more text groupings are one or more sentences of structured characters extracted by processing the set of unstructured text data; and

the one or more output predictions are one or more sentence labels, each sentence label corresponding to predicted relative order of a sentence in a group of related sentences.

10. The method of claim 9, further comprising:

determining, by the data processing system, a set of ground-truth training data, the set of ground-truth training data comprising at least a known label corresponding to a sentence label of the one or more sentence labels; and

training, by the data processing system, the machine learning model by comparing the sentence label of the one or more sentence labels to the a corresponding known label to determine an objective value and modifying a configuration of the machine learning model based on the objective value.

11. The method of claim 9, further comprising processing, by the data processing system, the one or more sentences and the one or more sentence labels to generate one or more question-and-answer pairs, each of the one or more question-and-answer pairs associated with at least one sentence as a textual question and at least one corresponding sentence as a textual answer to the textual question.

12. A system comprising:

one or more processors; and

a non-transitory computer-readable medium coupled to the one or more processors, the non-transitory computer-readable medium storing instructions executable by the one or more processors to cause the one or more processors to perform operations comprising:

receiving a set of unstructured text data;

generating one or more text groupings by processing the set of unstructured text data;

generating, based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model; and

generating one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model, each output prediction of the one or more output predictions corresponding to a predicted aspect of a text grouping of the one or more text groupings.

13. The system of claim 12, wherein:

the operation of processing the set of unstructured text data comprises extracting, from the set of unstructured text data, one or more sets of character; and

the operation of generating the one or more text groupings comprises grouping the one or more sets of characters according to a relative position of each character in the set of unstructured text data.

14. The system of claim 12, wherein the operation of generating the one or more text grouping embeddings comprises generating a plurality of sub-embeddings based on the set of unstructured text data or the one or more text groupings and generating the one or more text groupings based on the plurality of sub-embeddings.

15. The system of claim 14, wherein:

a sub-embedding of the plurality of sub-embeddings is a text sub-embedding generated based on one or more semantic aspects of the one or more text groupings;

wherein a sub-embedding of the plurality of sub-embeddings is a bounding sub-embedding generated based on one or more extracted spatial bounds of characters in the set of unstructured text data;

wherein a sub-embedding of the plurality of sub-embeddings is a visual sub-embedding generated based on one or more extracted image-based aspects of the set of unstructured text data; or

wherein a sub-embedding of the plurality of sub-embeddings is a relative font sub-embedding generated based on one or more different visual fonts of text in the set of unstructured text data.

16. The system of claim 12, wherein:

the one or more text groupings are one or more sentences of structured characters extracted by processing the set of unstructured text data; and

the one or more output predictions are one or more sentence labels, each sentence label corresponding to predicted relative order of a sentence in a group of related sentences.

17. A non-transitory computer-readable medium storing instructions executable by one or more processors for causing the one or more processors to perform operations comprising:

receiving a set of unstructured text data;

generating one or more text groupings by processing the set of unstructured text data;

generating, based on the one or more generated text groupings, one or more text grouping embeddings in a format for input to a machine learning model; and

generating one or more output predictions by inputting the one or more text grouping embeddings into the machine learning model, each output prediction of the one or more output predictions corresponding to a predicted aspect of a text grouping of the one or more text groupings.

18. The non-transitory computer-readable medium of claim 17, wherein:

the operation of processing the set of unstructured text data comprises extracting, from the set of unstructured text data, one or more sets of character; and

the operation of generating the one or more text groupings comprises grouping the one or more sets of characters according to a relative position of each character in the set of unstructured text data.

19. The non-transitory computer-readable medium of claim 17, wherein the operation of generating the one or more text grouping embeddings comprises generating a plurality of sub-embeddings based on the set of unstructured text data or the one or more text groupings and generating the one or more text groupings based on the plurality of sub-embeddings.

20. The non-transitory computer-readable medium of claim 19, wherein:

a sub-embedding of the plurality of sub-embeddings is a text sub-embedding generated based on one or more semantic aspects of the one or more text groupings;

wherein a sub-embedding of the plurality of sub-embeddings is a bounding sub-embedding generated based on one or more extracted spatial bounds of characters in the set of unstructured text data;

wherein a sub-embedding of the plurality of sub-embeddings is a visual sub-embedding generated based on one or more extracted image-based aspects of the set of unstructured text data; or

wherein a sub-embedding of the plurality of sub-embeddings is a relative font sub-embedding generated based on one or more different visual fonts of text in the set of unstructured text data.

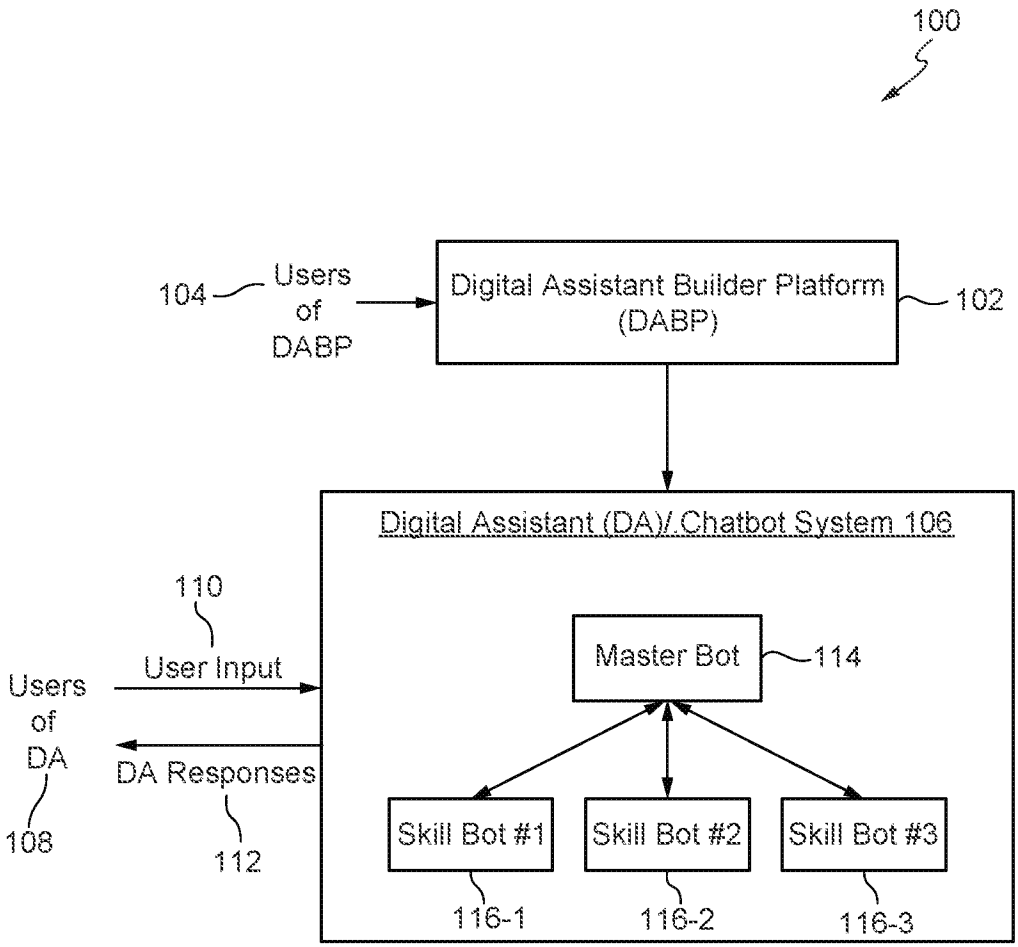

100
104
Users of DABP
Digital Assistant Builder Platform (DABP)
102
Digital Assistant (DA)/ Chatbot System 106
110
User Input
Users of DA
108
DA Responses
112
Master Bot
114
Skill Bot #1
Skill Bot #2
Skill Bot #3
116-1
116-2
116-3

FIG. 1

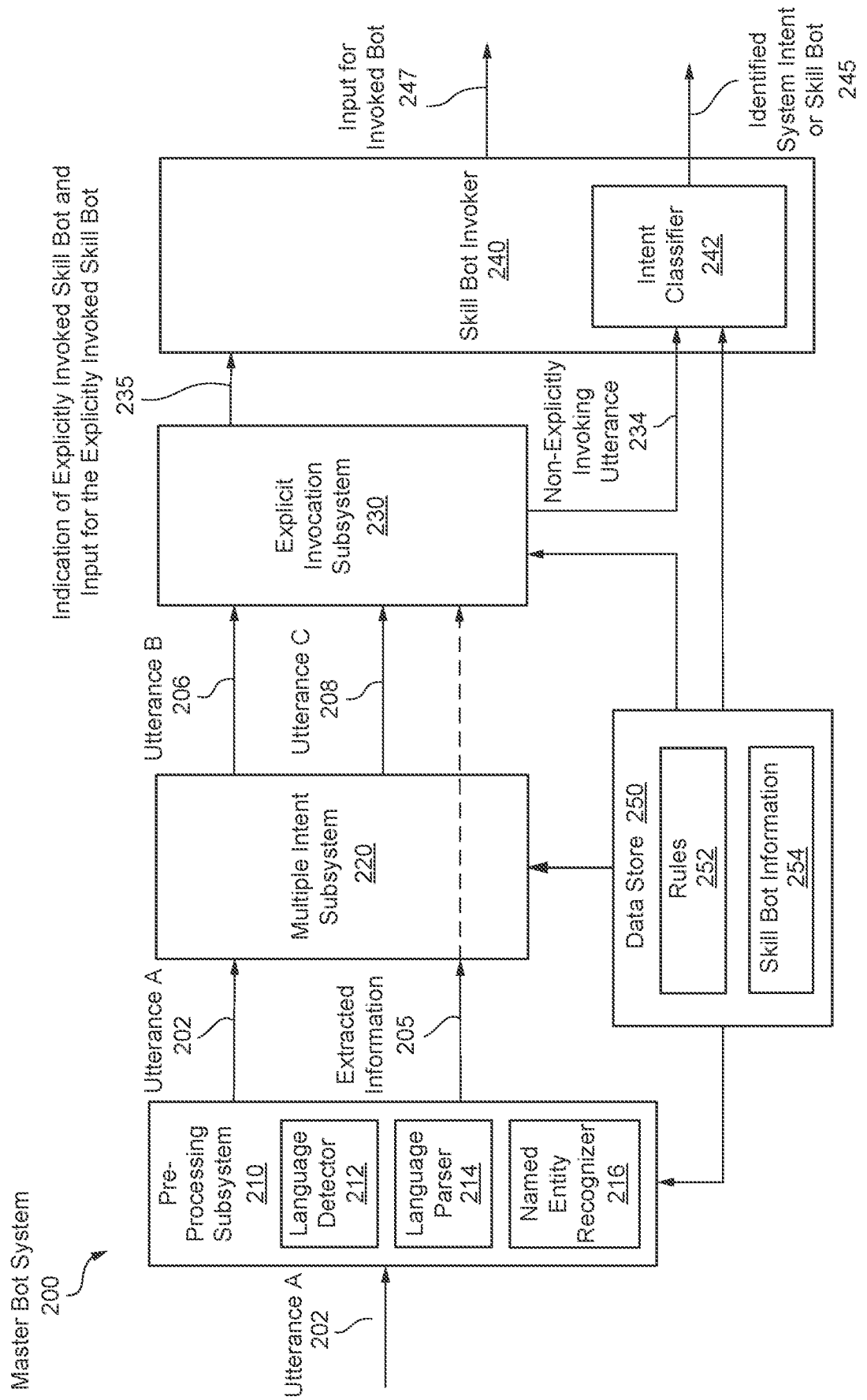

Master Bot System
200
Utterance A
202
Pre-Processing Subsystem
210
Language Detector
212
Language Parser
214
Named Entity Recognizer
216
Utterance A
202
Extracted Information
205
Multiple Intent Subsystem
220
Utterance B
206
Utterance C
208
Explicit Invocation Subsystem
230
Indication of Explicitly Invoked Skill Bot and Input for the Explicitly Invoked Skill Bot
235
Non-Explicitly Invoking Utterance
234
Skill Bot Invoker
240
Intent Classifier
242
Input for Invoked Bot
247
Identified System Intent or Skill Bot
245
Data Store
250
Rules
252
Skill Bot Information
254

FIG. 2

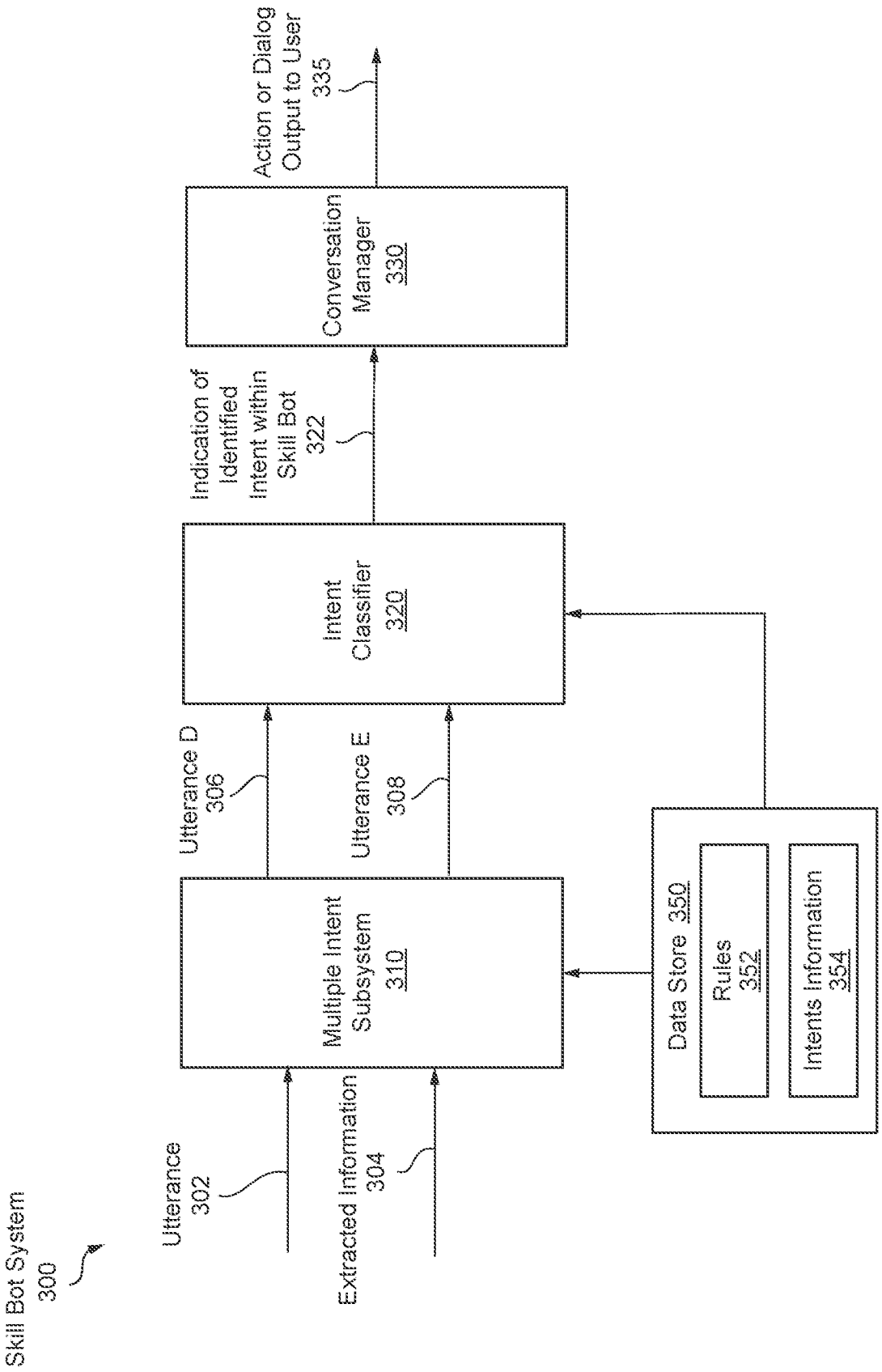

Skill Bot System
300
Utterance
302
Extracted Information
304
Multiple Intent Subsystem
310
Utterance D
306
Utterance E
308
Intent Classifier
320
Indication of Identified Intent within Skill Bot
322
Conversation Manager
330
Action or Dialog Output to User
335
Data Store 350
Rules
352
Intents Information
354

FIG. 3

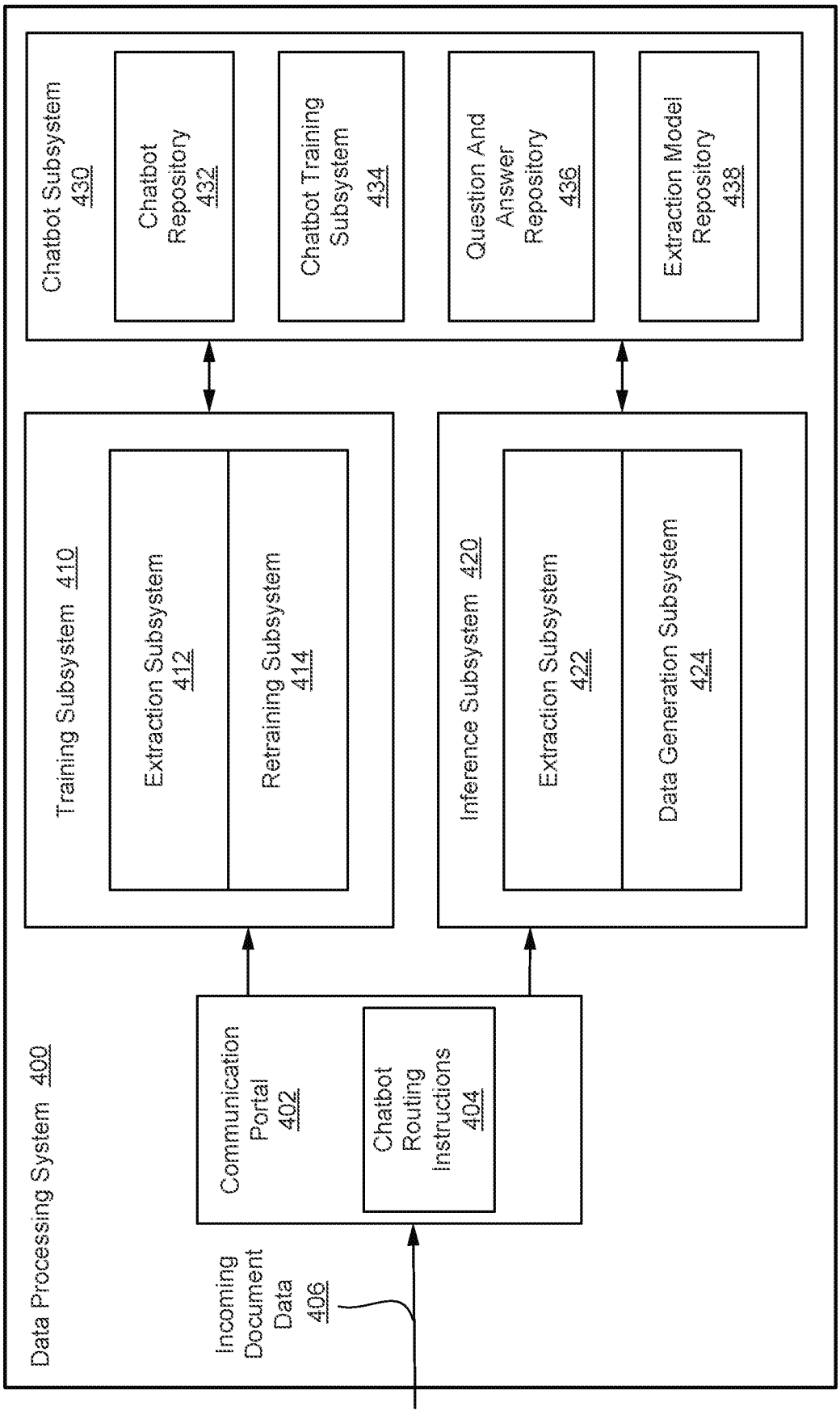

Data Processing System 400
Incoming Document Data 406
Communication Portal 402
Chatbot Routing Instructions 404
Training Subsystem 410
Extraction Subsystem 412
Retraining Subsystem 414
Inference Subsystem 420
Extraction Subsystem 422
Data Generation Subsystem 424
Chatbot Subsystem 430
Chatbot Repository 432
Chatbot Training Subsystem 434
Question And Answer Repository 436
Extraction Model Repository 438

FIG. 4

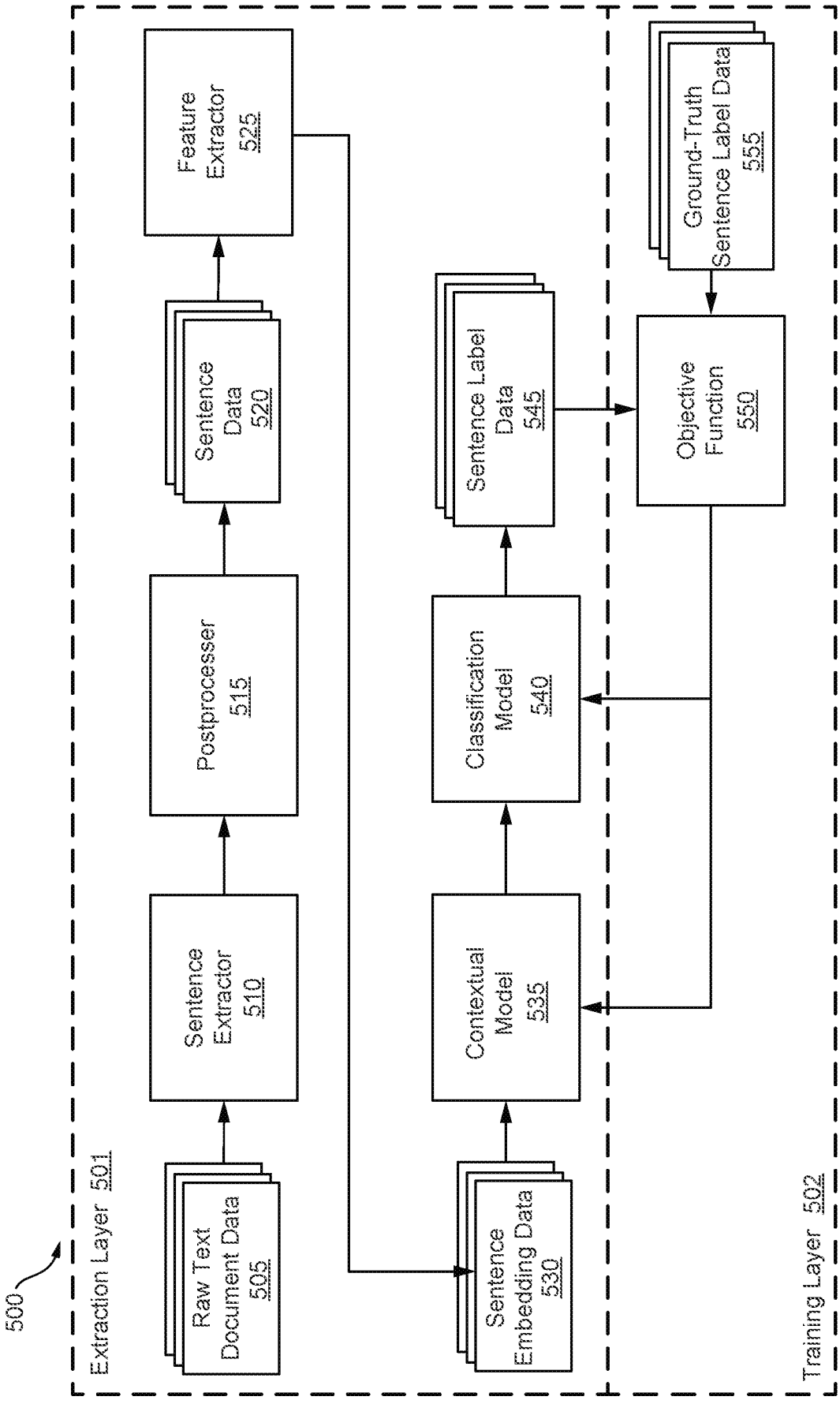

500
Extraction Layer 501
Training Layer 502
Raw Text Document Data 505
Sentence Extractor 510
Postprocesser 515
Sentence Data 520
Feature Extractor 525
Sentence Embedding Data 530
Contextual Model 535
Classification Model 540
Sentence Label Data 545
Ground-Truth Sentence Label Data 555
Objective Function 550

FIG. 5

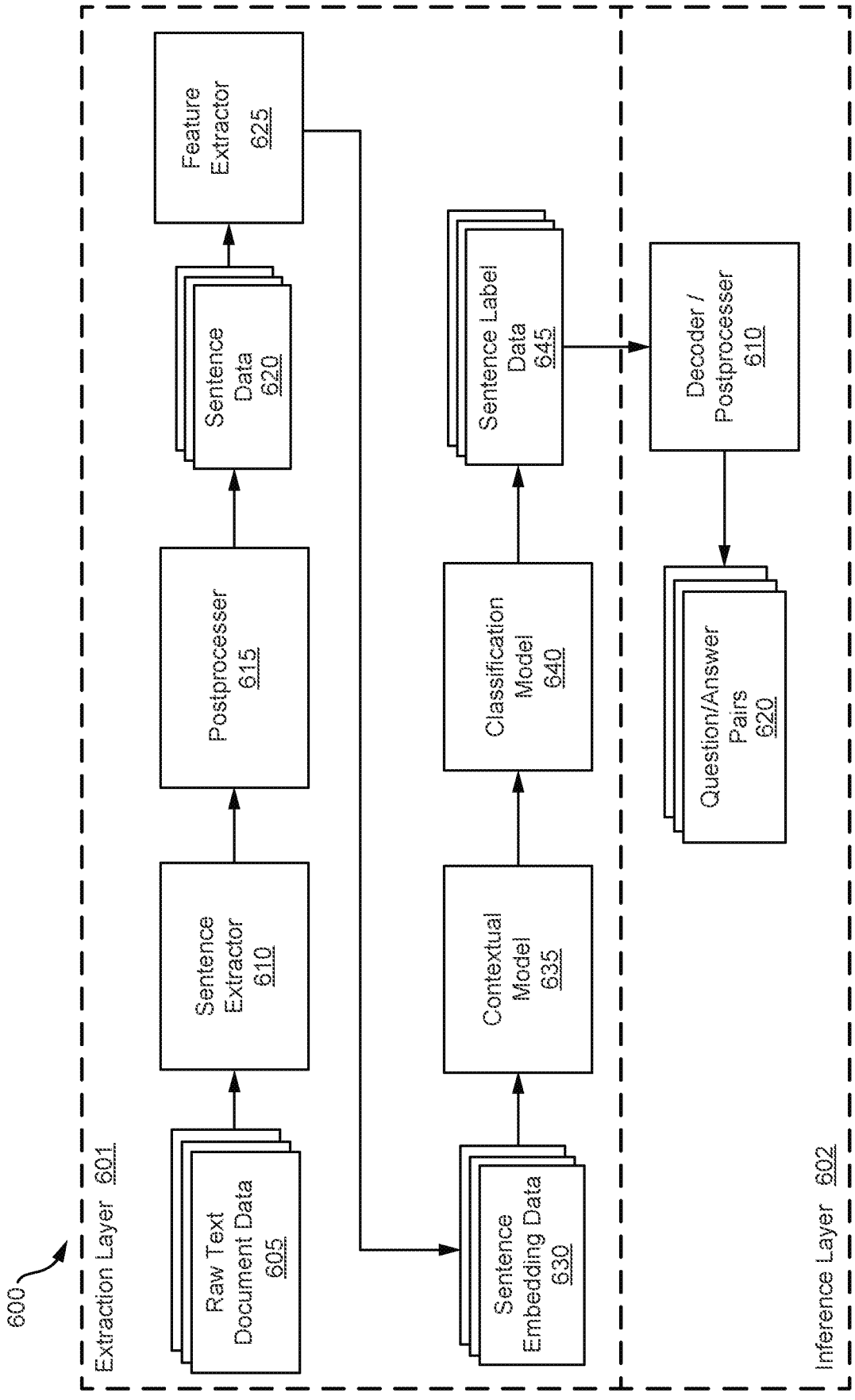

600
Extraction Layer 601
Raw Text Document Data 605
Sentence Extractor 610
Postprocesser 615
Sentence Data 620
Feature Extractor 625
Sentence Embedding Data 630
Contextual Model 635
Classification Model 640
Sentence Label Data 645
Decoder / Postprocesser 610
Question/Answer Pairs 620
Inference Layer 602

FIG. 6

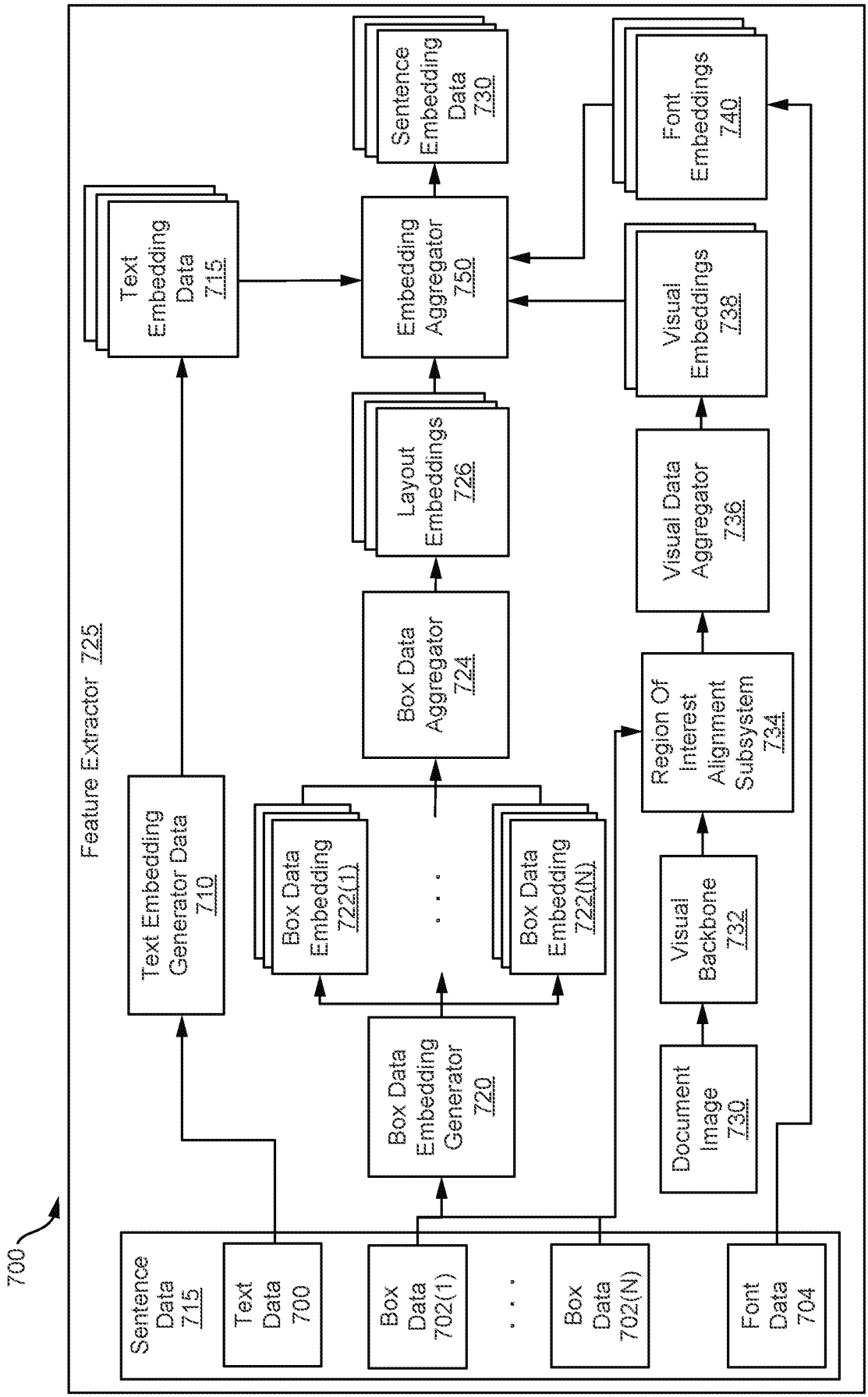

700
Feature Extractor 725
Sentence Data 715
Text Data 700
Box Data 702(1)
. . .
Box Data 702(N)
Font Data 704
Text Embedding Generator Data 710
Box Data Embedding Generator 720
Box Data Embedding 722(1)
Box Data Embedding 722(N)
Box Data Aggregator 724
Layout Embeddings 726
Text Embedding Data 715
Sentence Embedding Data 730
Embedding Aggregator 750
Document Image 730
Visual Backbone 732
Region Of Interest Alignment Subsystem 734
Visual Data Aggregator 736
Visual Embeddings 738
Font Embeddings 740

FIG. 7

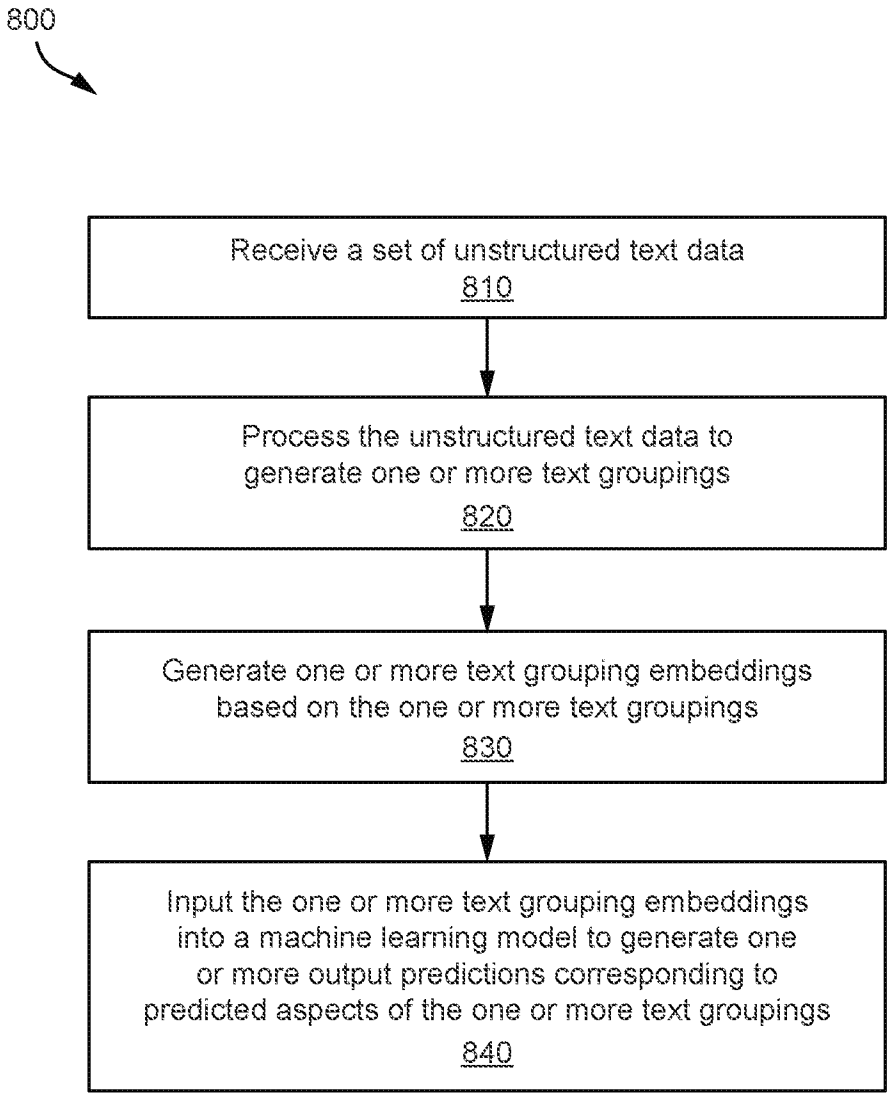

800
Receive a set of unstructured text data
810
Process the unstructured text data to generate one or more text groupings
820
Generate one or more text grouping embeddings based on the one or more text groupings
830
Input the one or more text grouping embeddings into a machine learning model to generate one or more output predictions corresponding to predicted aspects of the one or more text groupings
840

FIG. 8

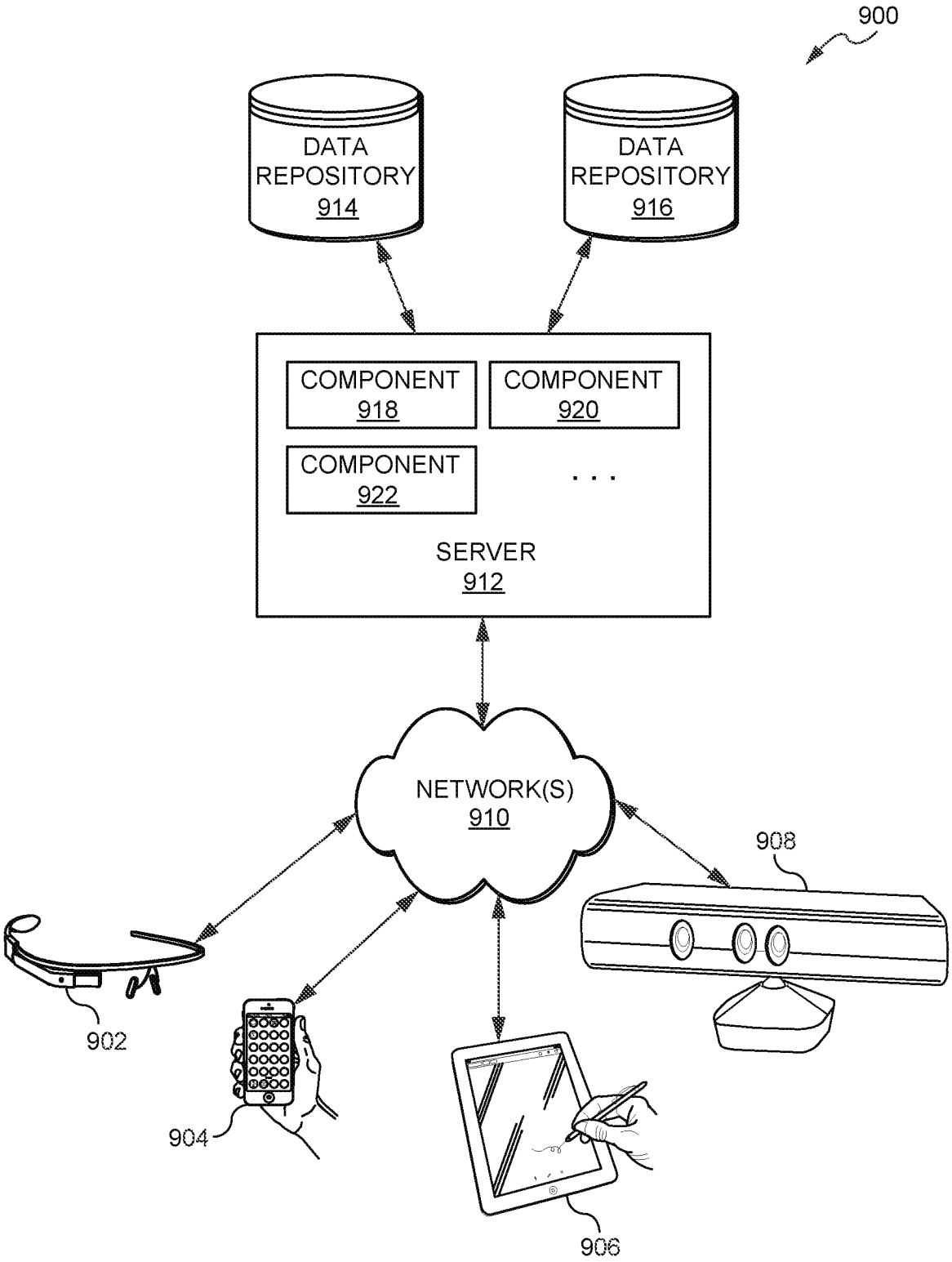

900
DATA REPOSITORY 914
DATA REPOSITORY 916
COMPONENT 918
COMPONENT 920
COMPONENT 922
. . .
SERVER 912
NETWORK(S) 910
908
902
904
906

FIG. 9

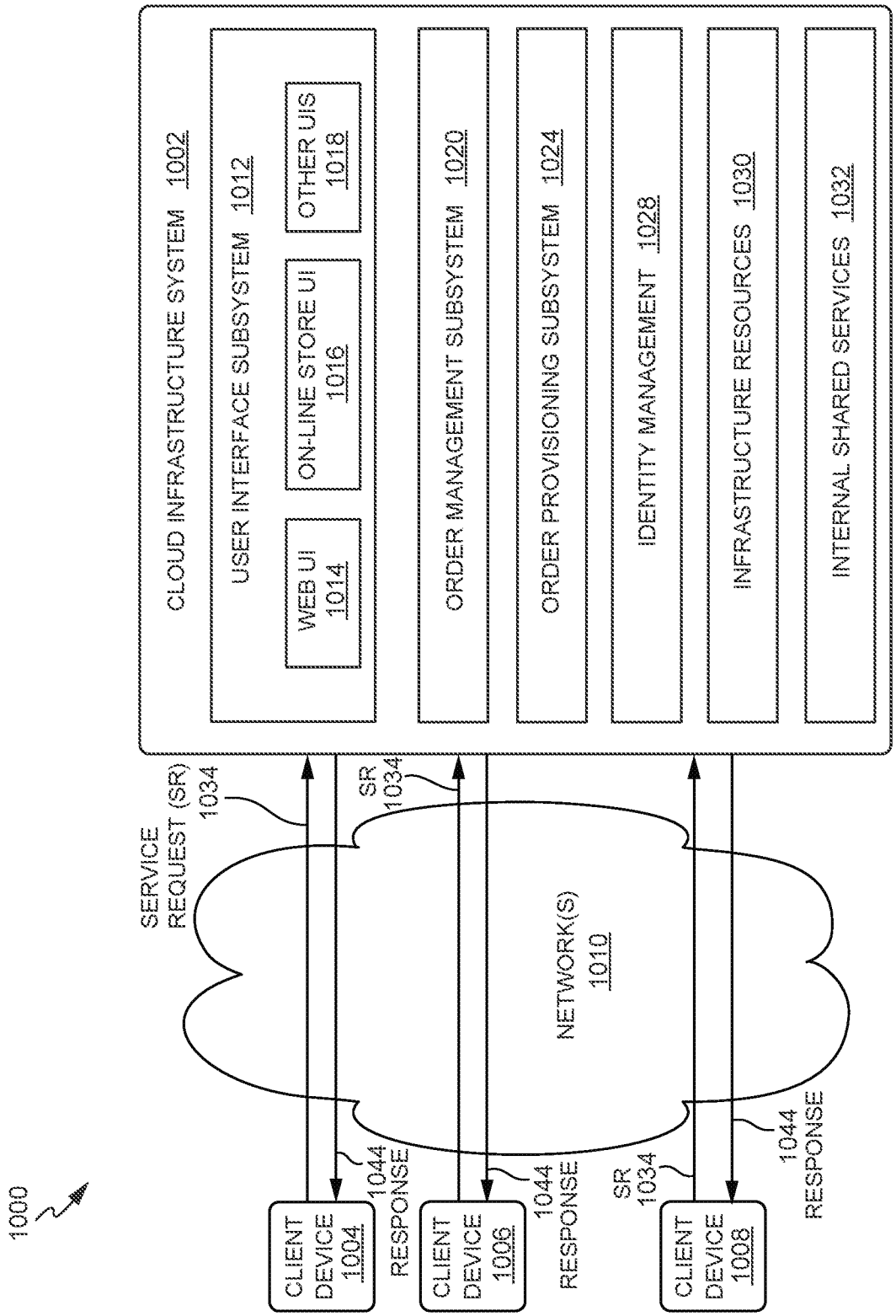

1000
CLOUD INFRASTRUCTURE SYSTEM 1002
USER INTERFACE SUBSYSTEM 1012
WEB UI 1014
ON-LINE STORE UI 1016
OTHER UIS 1018
ORDER MANAGEMENT SUBSYSTEM 1020
ORDER PROVISIONING SUBSYSTEM 1024
IDENTITY MANAGEMENT 1028
INFRASTRUCTURE RESOURCES 1030
INTERNAL SHARED SERVICES 1032
NETWORK(S) 1010
SERVICE REQUEST (SR) 1034
SR 1034
SR 1034
1044 RESPONSE
1044 RESPONSE
1044 RESPONSE
CLIENT DEVICE 1004
CLIENT DEVICE 1006
CLIENT DEVICE 1008

FIG. 10

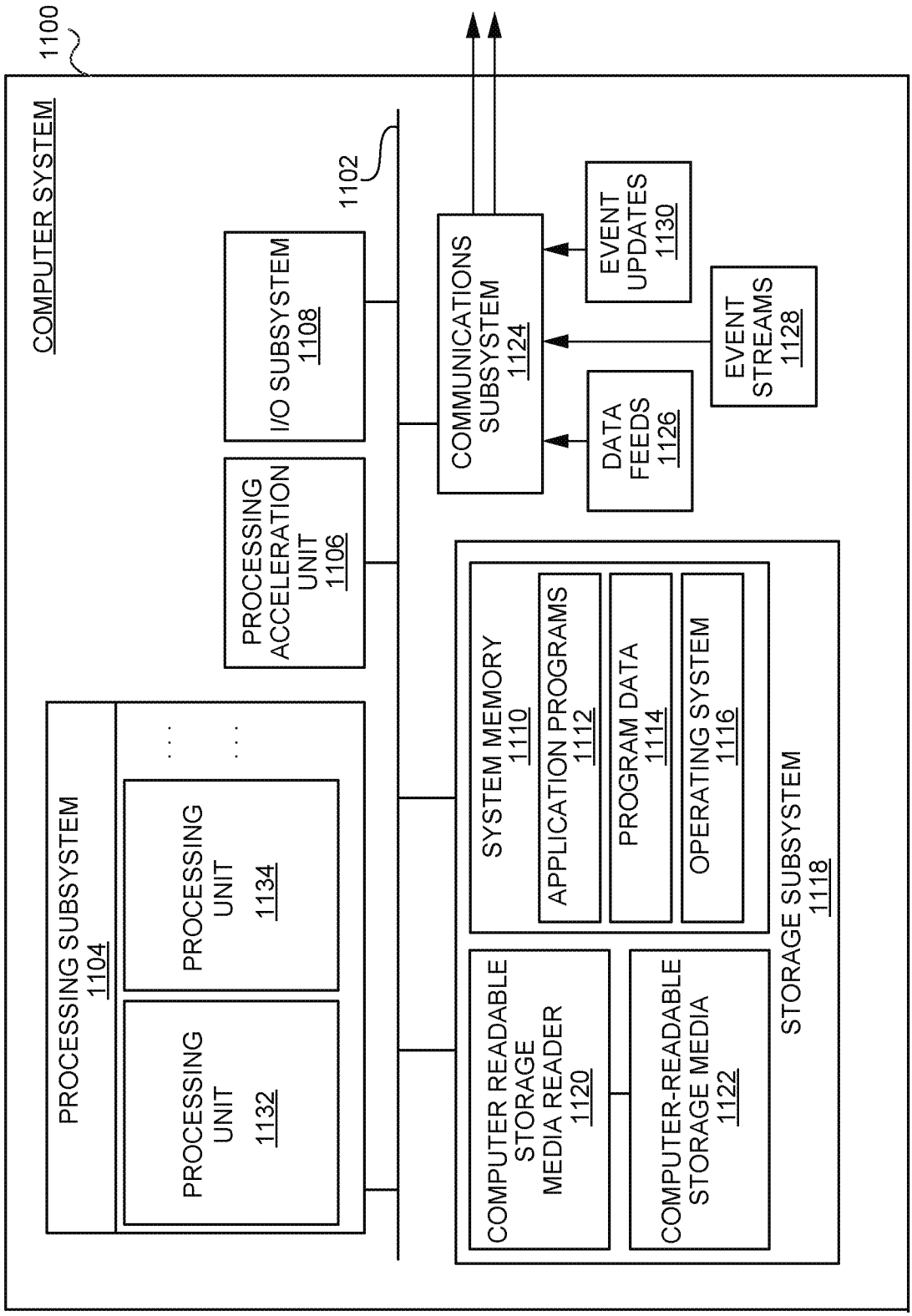

1100
COMPUTER SYSTEM
1102
PROCESSING SUBSYSTEM
1104
PROCESSING UNIT
1132
PROCESSING UNIT
1134
PROCESSING ACCELERATION UNIT
1106
I/O SUBSYSTEM
1108
COMMUNICATIONS SUBSYSTEM
1124
EVENT UPDATES
1130
EVENT STREAMS
1128
DATA FEEDS
1126
SYSTEM MEMORY
1110
APPLICATION PROGRAMS
1112
PROGRAM DATA
1114
OPERATING SYSTEM
1116
COMPUTER READABLE STORAGE MEDIA READER
1120
COMPUTER-READABLE STORAGE MEDIA
1122
STORAGE SUBSYSTEM
1118

FIG. 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2022/074974

A. CLASSIFICATION OF SUBJECT MATTER

G06F 16/332(2019.01)i; **G06F 16/33**(2019.01)i; **G06F 16/35**(2019.01)i; **H04L 51/02**(2022.01)i; **G06N 3/08**(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 16/332(2019.01); G06F 15/18(2006.01); G06F 16/00(2019.01); G06F 16/35(2019.01); G06F 16/93(2019.01); G06F 17/30(2006.01); G06N 5/02(2006.01)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: unstructured text data, grouping, embedding, machine learning model, prediction

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	KR 10-2019-0058935 A (WISENUT, INC.) 30 May 2019 (2019-05-30) paragraphs [0036]-[0087]; claims 1-10; and figures 1-3	1-2,12,17
A		3-11,13-16,18-20
Y	JP 2004-326600 A (FUJITSU LTD.) 18 November 2004 (2004-11-18) paragraphs [0022]-[0023], [0034]-[0050]; claims 1-3; and figures 1, 6-7	1-2,12,17
Y	US 10607042 B1 (LIVE OBJECTS, INC.) 31 March 2020 (2020-03-31) column 4, lines 29-33; claims 1-10; and figures 2-6	2
A	US 2020-0073882 A1 (ACCENTURE GLOBAL SOLUTIONS LIMITED) 05 March 2020 (2020-03-05) paragraphs [0029]-[0082]; claims 1-10; and figures 1-4	1-20
A	US 2019-0073420 A1 (BORISLAV AGAPIEV) 07 March 2019 (2019-03-07) paragraphs [0030]-[0033]; claims 1, 3-6; and figures 2, 4	1-20

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
29 November 2022	**29 November 2022**

Name and mailing address of the ISA/KR	Authorized officer
Korean Intellectual Property Office **189 Cheongsa-ro, Seo-gu, Daejeon** **35208, Republic of Korea**	**YANG, Jeong Rok**
Facsimile No. **+82-42-481-8578**	Telephone No. **+82-42-481-5709**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US2022/074974

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
KR 10-2019-0058935 A	30 May 2019	WO 2019-103224 A1	31 May 2019
JP 2004-326600 A	18 November 2004	None	
US 10607042 B1	31 March 2020	EP 3924865 A1	22 December 2021
		US 10592544 B1	17 March 2020
		US 10949455 B2	16 March 2021
		US 11100153 B2	24 August 2021
		US 2020-0293564 A1	17 September 2020
		US 2020-0334282 A1	22 October 2020
		US 2021-0374348 A1	02 December 2021
		US 2021-0390128 A1	16 December 2021
		US 2021-0406297 A1	30 December 2021
		WO 2020-167558 A1	20 August 2020
		WO 2020-167559 A1	20 August 2020
US 2020-0073882 A1	05 March 2020	None	
US 2019-0073420 A1	07 March 2019	US 10503791 B2	10 December 2019