

TASK-ORIENTED DIALOG METHOD, STORAGE MEDIUM, AND COMPUTER DEVICE

TECHNICAL FIELD

5 **[0001]** The present disclosure relates to the technical field of the Internet, and in particular, to a task-oriented dialogue method, a storage medium, and a computer device.

BACKGROUND

10 **[0002]** A task-oriented dialogue is an interactive dialogue manner that aims to assist users in efficiently completing a particular goal or task. In such dialogues, the artificial intelligence system aims to deeply understand the intent of a user and provide targeted information and suggestions, or perform related operations accordingly, so as to meet the diverse requirements of the user. At present, the task-oriented dialogue is widely used in various domains such as enterprise automation assistants, customer service support, and smart home controls, which can greatly improve work efficiency, optimize user experience, and reduce the requirement for human intervention.

15 **[0003]** Currently, the task-oriented dialogue mainly relies on natural language processing technologies such as intent recognition, slot filling, and knowledge graphs for implementation. However, when processing user intent, these methods usually rely on text classification to understand the core requirement in a dialog, or rely on a knowledge graph to find a solution. These methods require the system to pre-enumerate all problems that the user may encounter and corresponding solutions to the problems. However, the personalized requirements of the user are extremely diverse and difficult to exhaustively enumerate. Therefore, the construction of sample data for such methods is extremely difficult. In addition, these enumeration-based methods often have significant lag, and the accuracy of solution matching is limited by the predefined enumeration scope, which makes it difficult
20 to meet the changing actual requirements of the user.
25

[0004] Based on this, when dealing with some complex or specialized tasks, the user often needs to turn to human services for help, which not only reduces the timeliness of problem solving, but also increases the human resource costs.

SUMMARY

[0005] In view of this, embodiments of the present disclosure provide a task-oriented dialogue method, a storage medium, and a computer device, primarily aimed at solving the technical problems of low success rate, poor accuracy, and poor timeliness in existing task-oriented dialogues when solving complex and specialized tasks.

[0006] According to a first aspect of the present disclosure, a task-oriented dialogue method is provided, including:

[0007] in response to receiving dialogue information initiated by a user, sending, by a client, the dialogue information to a server;

10 [0008] in response to receiving the dialogue information, determining, by the server, intent information of the user through a large model, generating a tool execution plan according to the intent information, executing the tool execution plan to obtain a plan execution result, performing a check on the plan execution result, generating reply content based on a plan execution result that passes the check, and sending the reply content to the client; and

15 [0009] receiving and displaying, by the client, the reply content.

[0010] According to a second aspect of the present disclosure, a storage medium is provided, having a computer program stored thereon, where the program, when executed by one or more processors, implements the foregoing task-oriented dialogue method.

20 [0011] According to a third aspect of the present disclosure, a computer device is provided, including a storage medium, one or more processors, and a computer program stored on the storage medium and executable on the one or more processors, where the one or more processors, when executing the computer program, implement the foregoing task-oriented dialogue method.

25 [0012] According to the foregoing technical solutions, embodiments of the present disclosure provide a task-oriented dialogue method, a storage medium, and a computer device. The method may include: firstly, in response to receiving dialogue information initiated by a user, determining intent information of the user through a large model; subsequently, generating a tool execution plan according to the intent information; and

executing the tool execution plan to obtain a plan execution result; furthermore, performing a check on the plan execution result, and generating reply content based on a plan execution result that passes the check; and finally, sending the reply content to the client for display. In the foregoing method, intent recognition, tool invocation, tool execution, text check, and text summarization are performed through a large model, so that the text understanding and generation capabilities of the large model can be fully utilized to improve the timeliness and accuracy of intent recognition and text processing in a task-oriented dialogue. Compared to the traditional case-matching task-oriented dialogue method, the foregoing method can reduce the workload of task enumeration and improve the success rate of solving tasks, so that the task-oriented dialogue can be more flexibly applied to various professional domains and used to solve more complex, specialized and personalized tasks. In addition, in the foregoing method, the dialogue information of the user is decomposed into a simpler and clearer tool execution plan, and the scope of the reply content can be limited within the scope of the tool execution result. Therefore, the task decomposed by the large model can be prevented from becoming too divergent and deviating from the theme, thereby improving the accuracy of task execution. And at the same time, difficulty for the large model in task decomposition and execution can also be reduced, thus improving the timeliness of the task-oriented dialogue.

BRIEF DESCRIPTION OF DRAWINGS

[0013] FIG. 1 is a flowchart of a task-oriented dialogue method according to an embodiment of the present disclosure;

[0014] FIG. 2 is a flowchart of another task-oriented dialogue method according to an embodiment of the present disclosure;

[0015] FIG. 3 is a diagram of a system architecture of a task-oriented dialogue method according to an embodiment of the present disclosure;

[0016] FIG. 4 is a diagram of a system architecture of a case generation link according to an embodiment of the present disclosure;

[0017] FIG. 5 is a diagram of a system architecture of an information retrieval model according to an embodiment of the present disclosure;

[0018] FIG. 6 is a graph of variation trends of precision and recall rate versus threshold according to an embodiment of the present disclosure;

[0019] FIG. 7 is a structure diagram of a task-oriented dialogue apparatus according to an embodiment of the present disclosure;

5 [0020] FIG. 8 is a structure diagram of another task-oriented dialogue apparatus according to an embodiment of the present disclosure.

DETAILED DESCRIPTION

[0021] The following describes the present disclosure in detail with reference to the accompanying drawings and embodiments. It should be noted that the embodiments in the
10 present disclosure and the features in the embodiments may be combined with each other without conflict.

[0022] In view of the limitations of the existing Agent architecture in the implementation of task-oriented dialogues, especially the shortcomings in solving tasks in complex or professional domains, the present disclosure provides a task-oriented dialogue method based
15 on an E-Agent architecture. The method can find a better balance between speed, accuracy, stability, and easy training of models, so as to more effectively meet actual requirements of task-oriented dialogues. The E-Agent architecture includes an intent recognition module, a planning formulation module (Planner), an invocation execution module (Caller), a checking and verification module (Checker), and a content summarization module
20 (Summary). The working collaboratively between the modules can quickly and stably implement task-oriented dialogue and meet the individual requirement of users. The specific implementation of the E-Agent architecture will be specifically described in the following embodiments of the task-oriented dialogue method.

[0023] In an embodiment, as illustrated in FIG. 1, a task-oriented dialogue method is
25 provided, which is illustrated by taking the application of the method to a computer device such as a client and a server as an example. The method may include the steps described below.

[0024] At step 101, in response to receiving dialogue information initiated by a user, a

client sends the dialogue information to a server.

[0025] For example, the user may input any dialogue information through an information input box displayed on a client (e.g., an application installed on a smart terminal or a web page, etc.) and send the dialogue information. After receiving the dialogue information initiated by the user, the client may encapsulate the dialogue information into a standard format and send the dialogue information to a server through a network protocol. In this embodiment, the information input box is any component (e.g., the information input box may be a chat input box, a search input box, etc.) capable of receiving text information, and the dialogue information may be any text information input by the user and may also include other forms of information such as voice information, video information, picture information, etc.

[0026] At step 102, in response to receiving the dialogue information, the server determines intent information of the user through a large model, generates a tool execution plan according to the intent information, executes the tool execution plan to obtain a plan execution result, performs a check on the plan execution result, generates reply content based on a plan execution result that passes the check, and sends the reply content to the client.

[0027] The large model refers to an advanced artificial intelligence model, e.g., a large language model (LLM), a large multi-modal models (MMMs), etc., which is capable of processing complex tasks and large data volumes. The large model may be a large model in a general domain, or may be a large model in a vertical domain (the vertical domain model described below) that has undergone parameter optimization using sample data in a predefined domain. In this embodiment, each module in the E-Agent architecture may be implemented based on a large model. Before the function of each module is implemented by the large model, the server may perform various forms of pre-processing such as continued pre-training, supervised fine-tuning (SFT), and model quantization on the large model, to improve timeliness, stability, and accuracy of task execution of the large model. It may be understood that, a specific type of the large model used by each module in the E-Agent architecture is not specifically limited in this embodiment.

[0028] For example, after receiving the dialogue information initiated by the user, the

server may input the acquired dialogue information as a parameter into a pre-trained large model, so that the large model can accurately locate the intent information of the user based on the dialogue information. In this embodiment, when performing intent recognition, the large model may further acquire a plurality of types of information such as context information and application scenario information of a current dialogue, and user information acquired with user authorization, so that the large model can more accurately identify the intent information of the user. In this embodiment, the parameter input into the large model for intent recognition is not specifically limited, and a process of intent recognition may be implemented by the intent recognition module in the E-Agent architecture.

[0029] Further, after determining the intent information of the user, the server may input the intent information into another large model, so that the large model may select at least one tool based on the intent information, to generate a tool execution plan. In this embodiment, the tool execution plan may include at least one tool (for example, a program function interface) related to the intent information, where the scope of the tool selection may be predefined or obtained through a preceding vectorization-based coarse recall stage. By invoking a tool, the function corresponding to the tool can be implemented. For example, when the intent information is “query product sales quantity”, the tool selected by the large model may be a dish data query tool “dish_data_query”, where the dish data query tool “dish_data_query” is a tool execution plan generated based on the intent information “query product sales quantity”. It may be understood that, when the intention information is simple and clear, the tool execution plan is also simple and clear; and when the intention information is complex, more tools are needed, and the possibility of missing a tool is not ruled out. Therefore, a check is subsequently performed on the plan execution result of the tool execution plan to ensure that no tool is missing. In this embodiment, the scope of the tools selected by the large model and the specific functions of the tools are not specifically limited, where the process of generating the tool execution plan may be implemented by the planning formulation module (Planner) in the E-Agent architecture.

[0030] Further, after obtaining the tool execution plan, the large model may execute the tool execution plan to obtain a plan execution result. In this embodiment, the execution process of the tool execution plan is a process of invoking each tool in the tool execution plan. In this process, the large model may acquire tool parameters, invoke respective tools

based on the tool parameters to obtain respective invocation results of the tools, and subsequently, integrate the respective invocation results of the tools, so that the plan execution result is obtained. In this embodiment, the tool parameter required by a tool is generally conventional information such as a user ID, a product ID, an order ID, and a time range. Therefore, the tool parameter may be generally extracted from information such as dialogue information of the user, context information, and user information. In this embodiment, a manner of extracting the tool parameter by the large model is not specifically limited, and an execution process of the tool execution plan may be implemented by the invocation execution module (Caller) in the E-Agent architecture.

[0031] Further, after obtaining the plan execution result, the large model may perform a check on a completeness of the plan execution result, and determine whether the plan execution result passes the check based on the completeness of the plan execution result. If the plan execution result passes the check, the plan execution result may be input into a next stage to generate the reply content; if the plan execution result does not pass the check, a tool needs to be re-selected through the planning formulation module, and the tool execution plan is updated, then the updated tool execution plan is executed to obtain a updated plan execution result, and whether the updated plan execution result is complete is determined. This process may continue to loop until a modified plan execution result reaches the completeness requirement and/or the number of loops reaches a certain threshold, then the plan execution result is output, and finally the reply content is generated based on the updated plan execution result. In this embodiment, the large model may determine the completeness of the plan execution result according to the content of the dialogue information. The large model may determine that the plan execution result is complete when it is determined that at least one of the following two conditions is met: the tools related to the dialogue information have all been invoked, and/or, the plan execution result can solve the problems raised in the dialogue information. Then, the large model may output the plan execution result to use it to generate the reply content. The check process of the plan execution result may be implemented by the checking and verification module (Checker) in the E-Agent architecture.

[0032] Further, after the check on the plan execution result is passed, the large model may generate the reply content based on the plan execution result that passes the check, where

the reply content may be generated in combination with the dialogue information input by the user, i.e., when the reply content is generated, the content related to the dialogue information in the plan execution result may be retained, and the content unrelated to the dialogue information is removed, so as to obtain the reply content more related to the dialogue information. In this embodiment, the large model may further perform other forms of processing on the reply content, e.g., adding reassurance information, adding link information, etc., which is not specifically limited in this embodiment. A process of generating the reply content may be implemented by the content summarization module (Summary) in the E-Agent architecture.

10 [0033] In this embodiment, generating the intent information of the user based on the dialogue information of the user can clarify user's task requirement, and generating a tool execution plan based on the intent information can decompose the complex or personalized task requirement into a simple tool invocation list, so that the complexity of text information processing can be reduced. And compared to the traditional task decomposition method, 15 formulating the tool execution plan can avoid the task being decomposed overly fragmented or lengthy, which can thus improve the concentration of information processing and reduce the probability of errors in complex tasks. Further, executing the tool execution plan and performing a check on the plan execution result can improve the checking and feedback capabilities for the text information and reduce the possibility of information missing, and 20 finally, generating reply content based on a plan execution result that passes the check can improve the pertinence and specialization of the reply content.

[0034] At step 103, the client receives and displays the reply content.

[0035] For example, after generating the reply content, the server may send the reply content to the client, and the client may display the reply content in response to receiving 25 the reply content. For example, the client may display the reply content through various pages such as an instant messaging page and a search result page. In this embodiment, when outputting the reply content, the server may further output other information, for example, the server may further output the user's dialogue information and intent information, the tool names and tool invocation results, and the thought information of the large model, etc. In 30 this scenario, the server may send only part of the information output by the large model to the client for display, for example, the server may send only the reply content to the client,

and then record other information in a log, so as to facilitate future analysis and processing.

[0036] Further, as a refinement and extension of the specific implementation of the foregoing embodiment, to completely illustrate the specific implementation process of the present embodiment, another task-oriented dialogue method is provided, which may be
5 applied to the E-Agent architectures illustrated in FIG. 3 and FIG. 4, and as illustrated in FIG. 2, the method may include the steps described below.

[0037] At step 201, in response to receiving dialogue information initiated by a user, a client sends the dialogue information to a server.

[0038] At step 202, in response to receiving the dialogue information, the server acquires
10 context information, and determines intent information of the user through a large model based on the dialogue information and the context information.

[0039] In this embodiment, when performing intent recognition, the large model may further acquire a plurality of types of information such as application scenario information and user information acquired with user authorization, so that the large model can more
15 accurately identify the intent information of the user. In this embodiment, the parameter input into the large model for intent recognition is not specifically limited, and a process of intent recognition may be implemented by the intent recognition module in the E-Agent architecture.

[0040] In an optional implementation, step 202 may be implemented by: in response to
20 receiving the dialogue information, acquiring, by the server, context information and predefined application scenario information corresponding to the dialogue information; and determining the intent information of the user based on the dialogue information and the context information by using the application scenario information as a constraint condition.

[0041] For example, in response to receiving the dialogue information of the user, the
25 server may first acquire the context information and the predefined application scenario information corresponding to the dialogue information, where the context information may include information such as previous historical dialogue information and a session state of the user, and the application scenario information may be determined based on a scenario where the dialogue occurs, a service type used by the user, or a domain-specific information

processing logic. For example, in the scenario of the merchant assistant, the application scenario information is an application scenario of merchant service. In this scenario, all tasks revolve around the core theme of merchant service. Further, the server may use the acquired application scenario information as a constraint condition, which is combined with the dialogue information and the context information, and input into the pre-trained large model to accurately understand the natural language in the foregoing information through the large model and infer core intent of the user.

[0042] In the foregoing implementation, the large model may comprehensively consider the dialogue information, the context information, and the application scenario as a constraint condition to obtain the user intent information that is better in line with the current application scenario. For example, in a merchant service scenario, merchant users usually have expression habits such as colloquial expression content, lengthy dialogue content, and splitting the dialogue content into multiple sentences for sending. At this time, if the original multi-round dialogue is directly used for tool matching and case searching, the recall accuracy of tools and cases may face great challenges. Based on this, by configuring a multi-round rewrite module, the present implementation can utilize the robust natural language understanding capability of the large model, associate the context to rewrite the core intent of the current round, and ultimately output a clear and explicit goal (Goal).

[0043] In an optional implementation, the server may determine the intent information of the user in at least one of the manners described below.

[0044] At step 202-1, the server rewrites the dialogue information and the context information within a topic scope constrained by the application scenario information to obtain at least one piece of intent information of the user.

[0045] For example, when performing intent recognition, the server may rewrite the dialogue information of the user and the context information within a topic scope constrained by the application scenario information. The rewriting manner may include, but is not limited to, synonym replacement, semantic expansion, sentence restructuring, etc., to ensure that the rewritten content complies with the constraint condition of the application scenario. In this embodiment, for different themes of the recognized intent information, the large model may output at least one piece of intent information in a form of a list, so that the

large model can accurately select all available tools to formulate a relatively complete tool execution plan.

5 **[0046]** At step 202-2, in response to determining that the dialogue information and the context information are not within the topic scope constrained by the application scenario information, the server determines the dialogue information as the intent information of the user.

10 **[0047]** For example, in response to determining that the dialogue information and the context information are not within the topic scope constrained by the application scenario information, the server may no longer rewrite the dialogue information, but directly determine the dialogue information as the intention information of the user. At this time, the server may perform subsequent processing based on the dialogue information actually input by the user, e.g., the server may prompt the user to re-input or guide the user to output a more effective task requirement, etc.

15 **[0048]** At step 202-3, the server retrieves at least one case from an intent information experience repository through a predefined information retrieval model based on the dialogue information and the context information, and configures the at least one case into a prompt template to determine the intent information of the user based on the prompt template.

20 **[0049]** For example, the server may further retrieve a similar case through a predefined information retrieval model, and perform intent recognition with assistance of the similar case. The information retrieval model may include at least one of a sparse retrieval model, a dense vector retrieval model/ a dense retrieval model, or a multi-vector retrieval model. The server may retrieve at least one case related to a query condition from the intent information experience repository by inputting the dialogue information and the context
25 information as the query condition into the foregoing information retrieval model. Then, the server may configure these cases in the prompt template of the large model, and determine the intent information of the user according to the content of the prompt template. In this embodiment, the intent information experience repository can pre-store intent recognition cases in a plurality of scenarios, where the input of the intent recognition cases is multi-
30 round dialogue information formed by combining the context information and the dialogue

information, and the output is single-round intent information corresponding to the multi-round dialogue information.

[0050] In an optional implementation, when determining the intent information of the user, the dialogue information and the context information of the user may further be rewritten in at least one of the manners described below.

[0051] At step 202-1-1, the server retains information related to the application scenario information from the dialogue information and the context information, and generates at least one piece of intent information according to a theme of the retained content.

[0052] For example, the server may identify information closely related to the application scenario information from the dialogue information and the context information. The information may be a domain-specific term, concept, or procedure, etc. By retaining the information and rewriting the information according to the theme of the retained content, it can be ensured that the generated intent information is concise and clear. For example, the dialogue information and the context information are as follows: Merchant: "Did I get a positive review yesterday?"; Assistant: "You got a positive review yesterday after querying: 'The dish is perfectly to my taste, Great.'"; Merchant: "Can I pin this message? I want more consumers to see it.". After rewriting, the following two pieces of intent information may be retained: "How to pin yesterday's positive review?" and "How to let more people see yesterday's positive review?". Through the foregoing rewriting, the core intent of the merchant user can be effectively clarified.

[0053] In the foregoing implementation, the core intent related to the application scenario information is extracted from the complex dialogue information and the context information, so that the generated intent information can be more accurate and easy to be subsequently processed. In addition, the foregoing manner also helps to improve the understanding capability of the system for a specific domain, to better meet the requirements of users.

[0054] At step 202-1-2, the server retains time information and numerical information from the dialogue information and the context information in the intent information of the user.

[0055] For example, the server may extract time information and numerical information

from the dialogue information and the context information. The time information and the numerical information may be used to clarify the core intent of the user, and may be used as tool parameters to facilitate subsequent extraction. The server can ensure integrity and accuracy of the intent by retaining the information in the rewritten intent information. For example, when the merchant user asks: “Help me analyze the most recent week's positive reviews.”, the server may retain the time information of “the most recent week” in the intent information.

[0056] At step 202-1-3, the server replaces referential information from the dialogue information and the context information with name information, and retains the name information in the intent information of the user.

[0057] For example, the server may process referential information from the dialogue information and the context information, where the referential information refers to a pronoun, a noun phrase, etc., used by the user in the dialogue, and the referential information may be combined with the context information to clarify an object referred to by the referential information. Based on this, the server may, by analyzing the context information, replace the referential information with explicit name information, and retain the name information in the rewritten intent information. For example, if the merchant user asks: “How to contact him?”, the server can identify that “him” refers to a delivery person of an order, and then replace “him” with the delivery person of the order and retain it in the intent information.

[0058] In the foregoing implementation, the referential information is replaced with the explicit name information, so that the rewritten intent information can more clearly express the actual requirement of the user. The foregoing rewriting manner may help to eliminate ambiguity and uncertainty caused by the referential information, thereby improving accuracy and reliability of a system response.

[0059] At step 202-1-4, in response to identifying that the context information includes a differentiated topic different from the dialogue information, the server deletes content related to the differentiated topic from the intent information.

[0060] For example, the server may identify a differentiated topic different from the

dialogue information of the current round in the context information. The differentiated topic refers to a theme or topic changed by the user in the same dialogue, and information related to the differentiated topic may interfere with the system's understanding of the core intent. Based on this, content related to these differentiated topics may be deleted from the rewritten intent information. For example, if a merchant user first asks a problem about a product, and then asks a problem about an order, the server can identify a difference between the two topics, and then deletes content related to the product when generating intent information.

[0061] In the foregoing implementation, content related to the differentiated topic is deleted, so that rewritten intent information can be more focused on a current core intent, and this rewriting manner helps to reduce a processing amount of unnecessary information of the system, and improve a response speed and accuracy. In addition, because the content related to the differentiated topic is deleted, the rewritten intent information can be easier to be subsequently processed and analyzed.

[0062] In an optional implementation, the intent recognition process may be implemented by configuring a prompt template in the large model of the intent recognition module, for example, the content described below may be configured in the prompt template of the large model.

[0063] In combination with historical dialogues and application scenarios, the current problem of the merchant is restated with the identity of a merchant, making the statement clearer and easier to understand:

1. considering whether the historical dialogue is related to the application scenario, and if not, the current problem of the merchant is not to be rewritten;

2. in chatting scenarios such as greeting, criticism, and admiration, the current problem of the merchant is not to be rewritten;

3. after restatement, the time and order number conditions related to the problem cannot be defaulted;

4. self-invented conditions are prohibited in the restated result.

[0064] nResult: a merchant problem is expressed as a merchant identity, and multiple intents need to be decomposed into a list of multiple independent issues.

[0065] Case 1: Merchant: Did I get a positive review yesterday? Assistant: You got a positive review yesterday after querying: “The dish is perfectly to my taste, Great.”.

5 Merchant: Can I pin this message? I want more consumers to see it. Thought: The pinned object mentioned by the merchant is the foregoing positive review record, and the problem has a time condition: yesterday. Result: “How to pin yesterday's positive review?” and “How to let more people see yesterday's positive review?”

[0066] In an optional implementation, the intent information is generated by a pre-trained
10 large model, where the pre-training method for the large model may be implemented in the following manner: performing, by the server, continued pre-training on a large model in a general domain based on a sample dataset in a predefined domain, where the sample dataset is provided with random noise and does not include sample user information; then, performing parameter fine-tuning on the large model after the continued pre-training based
15 on an intent recognition case in the predefined domain; and finally, generating intent recognition information of a predefined problem in the predefined domain based on the fine-tuned large model, and iteratively updating a parameter of the large model through a reinforcement learning algorithm based on positive and negative feedback information of the intent recognition information to obtain the pre-trained large model.

20 [0067] For example, when performing model training, the server may load a large model in a general domain as a base model, and then use a sample dataset in a predefined domain to perform continued pre-training on the base model. The dataset may include domain-specific task-oriented dialogue data, and meanwhile, random noise may be added to the sample data to enhance the robustness of the model, while ensuring that the dataset does not include any
25 sample user information to protect the privacy of the user. After completing the continued pre-training, the server may perform parameter fine-tuning (SFT) on the model based on the intent recognition cases in the predefined domain. The intent recognition cases may include multi-round dialogue information in a predefined domain and corresponding intent recognition results, and these cases may be used to guide the model to recognize domain-specific user intent. During the fine-tuning process, the large model may adjust its internal
30 parameters according to the input and expected output of the case to optimize the domain-

specific performance of the large model. Subsequently, the server may generate intent recognition information of predefined problems in the predefined domain by utilizing the fine-tuned model. The intent recognition information may be used to interact with the user and be iteratively optimized according to the user feedback. For example, the server may collect positive and negative feedback information of the user for the intent recognition information, and iteratively update parameters of the model through a reinforcement learning algorithm (RLHF). The reinforcement learning algorithm may adjust a behavior policy of the model based on the feedback of the user, so that the model can more accurately identify the intent of the user. Finally, performance of the model may be further improved through technologies such as deep pseudo-reinforcement learning (proximal policy optimization, DPO) to obtain a pre-trained large model. The pre-trained model not only has the ability to accurately recognize the user intent in a specific domain, but also can enable continuous learning and optimization during interaction with users.

[0068] At step 203, the server selects at least one tool based on the intent information, to generate a tool execution plan.

[0069] For example, after determining the intent information of the user, the server may input the intent information into the large model, so that the large model may select at least one tool based on the intent information to obtain the tool execution plan. In this embodiment, the tool execution plan may include at least one tool (i.e., a program function interface) related to the intent information, where the scope of the tool selection may be predefined or obtained through a preceding vectorization-based coarse recall stage. By invoking a tool, the function corresponding to the tool can be implemented. For example, when the intent information is “query product sales quantity”, the tool selected by the large model may be a dish data query tool “dish_data_query”, where the dish data query tool “dish_data_query” is a tool execution plan generated based on the intent information “query product sales quantity”. It may be understood that, when the intention information is simple and clear, the tool execution plan is also simple and clear; and when the intention information is complex, more tools are needed, and the possibility of missing a tool is not ruled out. Therefore, a check is subsequently performed on the plan execution result of the tool execution plan to ensure that no tool is missing. In this embodiment, the scope of the tools selected by the large model and the specific functions of the tools are not specifically

limited, where the process of generating the tool execution plan may be implemented by the planning formulation module (Planner) in the E-Agent architecture.

[0070] In an optional implementation, step 203 may be implemented in the following manner: acquiring, by the server, context information, and according to the context
5 information, searching a tool library for a plurality of candidate tools related to the intent information to generate a candidate tool list; and then, selecting at least one tool from the candidate tool list based on the intent information, and arranging and combining the at least one selected tool to obtain the tool execution plan.

[0071] For example, the server may search a pre-constructed tool library for a plurality of
10 candidate tools related to the context information according to the context information, and this process may be completed before the Planner module selects the tools and implemented as a coarse recall stage. This process may be completed by matching the context information with information such as descriptions, functions, or labels of tools in the tool library to obtain a candidate tool list. Further, the server may use a predefined algorithm or rule to select at
15 least one tool from the candidate tool list based on the specific content of the intent information. These tools may be arranged and combined based on logical relationships and dependency relationships in a task completion process, to form a tool execution plan of one or more tools. It should be noted that, the generation method of the tool execution plan is not only suitable for the merchant service scenario, but also has strong universality and
20 extensibility, which can be widely applied to other scenarios that require the assistance of tools to complete tasks.

[0072] Compared to the manner of directly decomposing a task in the XAgent architecture, the foregoing implementation can avoid problems of the task being decomposed too divergently and lengthy task lists. In addition, compared to a manner of the ReWOO
25 architecture generating a complete plan at one time, in the foregoing implementation, by accurately matching the intent information and a candidate tool, the accuracy of tool selection can be improved, thereby improving feasibility and execution efficiency of the execution of the entire plan. In addition, by adding a vectorization-based coarse recall stage of tools into the preceding stage, it is possible to more rapidly locate the most relevant tool
30 set matching the intent information, which can thereby enhance both the efficiency and accuracy of tool selection.

[0073] In an optional implementation, the server may select the tool in at least one of manners described below.

[0074] At step 203-1, the server performs keyword matching on the candidate tool list based on the intent information to select at least one tool.

5 [0075] For example, the server may quickly select at least one tool related to the intent information from a pre-constructed candidate tool list through a keyword matching algorithm based on the intent information and information such as a description and a label of the tool. The keyword used for matching may either originate from intent information of the user, or be other extracted key elements through processing (e.g., word segmentation,
10 stop word removal, etc.).

[0076] In the foregoing implementation, the keyword matching algorithm is adopted to select the tool, so that the tool directly related to the user intent can be quickly located, enhancing both the efficiency and accuracy of tool selection. In addition, the foregoing manner does not require a complex natural language processing process, which can reduce
15 system complexity and computing resource requirements.

[0077] At step 203-2, in response to determining that intent information of a current round has a same meaning as intent information of a previous round, the server determines a tool different from a tool of the previous round as a selected tool of the current round.

[0078] For example, the server may record historical intent information of the user. When
20 determining that the intent information of the current round has the same meaning as that of the previous round (or the previous several rounds), the server can avoid repeatedly selecting the same tool, but add a new tool to the tools already used in the previous round as the selection of the current round, or replace the tools already used in the previous round as the selection of the current round. This tool selection strategy helps to provide diversified tool
25 support in continuous tasks and enhance user experience.

[0079] At step 203-3, the server determines a tool mutually exclusive with an already selected tool as a disabled tool of the current round.

[0080] For example, the server can maintain a tool mutex rule repository, which can include mutex relationships between tools. When selecting a tool, the server may check whether the to-be-selected tool has a mutex relationship with the already selected tool. If there is a mutex relationship, the server will determine the tool as the tool disabled in the current round to ensure that the tool conflicting with the already selected tool will not be selected.

[0081] At step 203-4, in response to determining that no tool is selected based on the intent information, the server determines a knowledge retrieval tool as a selected tool of the current round.

[0082] For example, when a suitable tool is unable to be selected based on the intent information, the server may adopt an alternative strategy, i.e., the knowledge retrieval tool is determined as the tool selected in the current round. The knowledge retrieval tool has a powerful information search and integration capability in a specific domain, which can help the server search a more professional knowledge base for information or resources related to intent information, thereby providing support for subsequent task execution.

[0083] At step 203-5, the server retrieves at least one case from an execution plan experience repository through a predefined information retrieval model based on the intent information, and selects at least one tool from the candidate tool list based on the at least one case.

[0084] The information retrieval model may include at least one of a sparse retrieval model, a dense vector retrieval model, or a multi-vector retrieval model.

[0085] For example, the server may retrieve a historical case similar to the intent information in the execution plan experience repository by utilizing a predefined information retrieval model (e.g., a sparse retrieval model, a dense vector retrieval model, or a multi-vector retrieval model) based on the intent information of the user. These historical cases include experience and tool combination information for successful completion of similar tasks. By analyzing these cases, the server may select at least one tool similar to the tool used in the cases from the candidate tool list.

[0086] In an optional implementation, the server may select a tool based on the retrieved case in at least one of the manners described below.

[0087] At step 203-5-1, in response to determining that a matching degree between a case and the intent information is greater than a preset threshold, the server determines a tool in the case as a selected tool.

5 [0088] At step 203-5-2, in response to determining that the matching degree between the case and the intent information is less than or equal to the preset threshold, the server configures the case into a prompt template to select the at least one tool from the candidate tool list based on the prompt template.

10 [0089] For example, after retrieving the case related to the intent information in the pre-constructed execution plan experience repository by utilizing the predefined information retrieval model, the server may calculate a matching degree between each case and the intent information, and compare the calculated matching degree with the preset threshold. If the matching degree between the retrieved case and the intent information is greater than the preset threshold, the tool used in the case may be used as the selected tool. At this time, a case with a high matching degree usually means that a tool used by the case is highly
15 correlated with current intent information, and can well meet the requirement of a user. On the contrary, if the matching degree between the selected case and the intent information is less than or equal to the preset threshold, the retrieved case may be configured into the prompt template of the large model, so that the large model selects at least one tool from the candidate tool list according to the description and related information of the case in the
20 prompt template. This tool selection manner allows the user to make further determination and selection based on partially matched cases, thereby improving the flexibility and accuracy of tool selection.

[0090] In the foregoing implementation, the tool is selected based on the matching degree between the case and the intent information, so that accuracy and efficiency of tool selection
25 can be effectively improved.

[0091] In an optional implementation, the server may arrange and combine the selected tools to obtain the tool execution plan in the following manner: arranging, by the server, tools in sequence and/or in parallel based on a dependency relationship between tool parameters of the tools; and performing operational connection on the tools arranged in sequence and/or
30 in parallel through a predefined operator to obtain the tool execution plan.

[0092] For example, the server may select at least one tool from the tool library based on the intent information of the user. For example, in the merchant service domain, tools that may be selected by the server include, but are not limited to, various tools such as a knowledge retrieval tool, business diagnosis, merchant XX basic information query, merchant XX business data query, and write operation interface for configuring an XX function, and a total of more than hundreds of tools are available. After selecting the appropriate tools, the server can analyze the dependency relationships between the tool parameters of each tool. These dependency relationships may be based on factors such as the continuity of data flow, requirements for the sequential order of operations, or requirements of data processing logic, etc. The server can arrange the selected tools in sequence and/or in parallel based on these dependency relationships. The tools arranged in sequence need to be executed in a specific order to ensure the accuracy of data and the coherence of operations; and the tools arranged in parallel can be executed in parallel to improve the processing efficiency. After completing the tool arrangement, the server may use predefined operators, e.g., a bracket, a comma, etc., to perform operational connection on the tools arranged in sequence and/or in parallel. The operators can not only define the execution order of the tools, but also ensure the logical correctness of the entire tool execution plan. By connection of the operators, the server can eventually generate a tool execution plan such as “[(T1, T2), (T3)]”, where T1 and T2 represent tools that need to be executed sequentially, and T3 represents the tool that can be executed in parallel with T1 and T2.

[0093] In an optional implementation, the tool execution plan may be implemented by configuring a prompt template in the large model of the planning formulation module, for example, the content described below may be configured into the prompt template of the large model.

[0094] You are an intelligent robot of a delivery platform, referred to as a merchant assistant, and you may have the highest authority of a merchant-side APP to communicate with a merchant in the APP. Your core goal is to understand and solve merchant problems and improve their customer satisfaction, and the current responsibility is to select multiple tools to invoke based on merchant problems.

[0095] Objective: The intent of the merchant must be correctly analyzed and a useful tool selected from the list of tools provided based on this intent. The API tools may be used are described below.

[0096] Unclear_attention: represents that the intent is unclear or meaningless interjection;

5 unrelated_biz: represents that the intent of the merchant is irrelevant to the delivery scenario; Question: the user problem; Thought: please peruse the chat records between user and you, analyze the user problem, so as to select the list of tools that are most suitable for the merchant's problem; Action: the list of tools that are most suitable for solving the problem, which cannot be empty, a plurality of tools are separated by comma in English, and the
10 selected tools must be within {tool_names}; Observation: the feedback given after invoking the tools.

[0097] Note:

[0098] * A tool must be prioritized to use, and if every other tool cannot be matched and used, then “knowledge_retrieval_tool” is used;

15 [0099] * Note that when a user repeatedly asks the same question, it may indicate that an error tool was invoked in the last time. Please consider changing the tool as appropriate;

[0100] * The tools are mutex, such as open and close. Do not invoke them simultaneously. For example, open and close;

[0101] * If the merchant's problem is a meaningless interjection or incomplete input, you
20 must use “unclear_attention”;

[0102] * Guessing the intent of the merchant is prohibited, selecting a tool that is not provided is prohibited, not selecting a tool is prohibited, directly answering is prohibited, and waiting for a new problem of the merchant is prohibited;

[0103] Case 1: Question: How to bind XX (Note: XX is used to refer to an instant messaging
25 application software). Thought: XX is not directly associated with the delivery scenario, but as an online delivery platform, it may be associated with an XX communication tool, and “knowledge_retrieval_tool” is tried to retrieve a knowledge to answer. Action: knowledge_retrieval_tool.

[0104] In an embodiment, the tool execution plan may be generated by a pre-trained large model. The pre-training method for the large model may be implemented in the following manner: performing, by the server, continued pre-training on a large model in a general domain based on a sample dataset in a predefined domain, then, performing parameter fine-tuning on the large model after the continued pre-training based on a tool invocation case in the predefined domain, and finally, converting a parameter weight of the fine-tuned large model from a floating-point number to an integer with a preset number of bits to obtain the pre-trained large model.

[0105] For example, when training the model, the server may load a large model in a general domain as a base model, and then use a sample dataset in a predefined domain to perform continued pre-training on the base model. The dataset may include domain-specific tool invocation data, and through the foregoing training process, the model may be better adapted to data distribution and language patterns in a specific domain. After the continued pre-training is completed, the server may further perform parameter fine-tuning (SFT) on the large model after the continued pre-training based on the tool invocation case in the predefined domain. These tool invocation cases may include a mapping relationship between a user problem and a tool, so that the model can be guided to learn how to accurately identify a tool that needs to be invoked from the user problem. Through model fine-tuning, the large model can better understand the intent of the user problem and accurately map it to the corresponding tool. After completing the fine-tuning, the server may convert a parameter weight of the fine-tuned large model from a floating-point number to an integer with a preset number of bits (int4 quantization), and this step can maintain the performance of the model to some extent, while greatly reducing the complexity of the model and the computing resources consumption during running. Finally, the large model processed by the foregoing steps may be used to generate a tool execution plan. When a user raises requirements, the model can accurately generate corresponding tool execution plan based on the user's input problems.

[0106] At step 204, the server extracts a tool parameter from the context information, and invokes the at least one tool in the tool execution plan based on the tool parameter, to generate a plan execution result.

[0107] For example, after obtaining the tool execution plan, the large model may execute the tool execution plan to obtain a plan execution result. In this embodiment, the execution process of the tool execution plan is a process of invoking each tool in the tool execution plan, and in this process, the large model can obtain the tool parameters, and invoke each tool based on the tool parameters to obtain the invocation result of each tool, and subsequently, the plan execution result can be obtained by integrating the respective invocation results of each tool. In this embodiment, the tool parameter required by the tool is generally conventional information such as a user ID, a product ID, an order ID, and a time range, and therefore, the tool parameter may be generally extracted from information such as dialogue information of the user, context information, and user information. In this embodiment, a manner of extracting the tool parameter by the large model is not specifically limited, and an execution process of the tool execution plan may be implemented by an invocation execution module (Caller) in the E-Agent architecture.

[0108] In an optional implementation, step 204 may be implemented in the following manner: acquiring, by the server, context information, extracting, from the dialogue information and the context information, tool parameters required by tools in the tool execution plan; and then, invoking the tools sequentially and/or in parallel based on the tool parameters in accordance with an order of the tools as arranged and combined in the tool execution plan to obtain the plan execution result.

[0109] For example, the server may extract the respective tool parameters of each tool in the tool execution plan from the context information, where the tool parameters may include, for example, a plurality of types of data such as a time parameter, a product ID, and an order ID. By extracting the tool parameters, it can be ensured that each tool in the tool execution plan can be executed correctly and efficiently. After determining the tool parameters, the server may invoke the tools in accordance with an order of tool arrangement and combination defined in the tool execution plan. In many domains, such as the merchant service domain, there is no explicit dependency relationship between most tools, and at this time, the server may prioritize parallel invocation to the tools, to minimize the overall execution delay of the tools. In addition, when there is a clear dependency relationship between tools, the server may also perform sequential invocations on the tools according to the dependency order to ensure correct execution of the tools.

[0110] In the foregoing implementation, the tools in the tool execution plan are invoked in parallel, so that an overall execution delay of the tools can be reduced to a large extent, thereby improving a response speed and working efficiency of task execution. In addition, automatically extracting tool parameters based on the dialogue information and the context information can ensure that each tool is accurately invoked, thereby reducing the risk of operation errors and improving the automation and intelligence level of the system.

[0111] In an optional implementation, the server may further acquire the tool parameters in at least one of the manners described below.

[0112] At step 204-1, in response to identifying that there is a missing tool parameter, the server postpones an execution order of a tool whose tool parameter is missing, and extracts the missing tool parameter from an execution result of an already executed tool.

[0113] For example, when the server executes the tool execution plan, and identifies that an input parameter on which a tool depends is missing in the current context, a fault tolerance mechanism may be triggered, i.e., the server may postpone the execution order of the tool, so that the previous tools may have an opportunity to generate or provide a missing parameter in the execution process. After the execution of the previous tools is completed, the server may check again whether new parameter information is obtained. If the missing parameter is obtained, the server may continue to execute the tool in the original plan; otherwise, the server may consider adopting other strategies, such as querying the user to obtain the tool parameter or performing partial execution according to existing information.

[0114] In the foregoing embodiment, by combining the parameter automatic extraction strategy and the parameter extraction fault tolerance mechanism, the robustness of the tool execution plan and the success rate of the tool execution can be significantly improved, and the foregoing method can effectively reduce unnecessary user interaction and improve the efficiency and success rate of the tool execution.

[0115] At step 204-2, the server initiates query information based on the missing tool parameter, and extracts the missing tool parameter from user response information to the query information.

[0116] For example, when the server identifies that there is a missing tool parameter, in addition to triggering the fault tolerance mechanism, the server may also initiate query information to the user based on the missing parameter, and display the query information to the user through the user interface. The user may provide the missing parameter by replying to the query information. After receiving the response from the user, the server may extract the missing parameter therefrom, and continue to execute the tools in the original plan.

[0117] In the foregoing implementation, the parameter is obtained by actively querying the user in the process of plan execution, so as to ensure that the plan can be continuously and stably executed. This method of dynamically obtaining parameters can not only improve the efficiency of tool execution, but also improve the success rate of plan execution, avoid execution failure during the plan, and avoid resource waste.

[0118] In an optional implementation, the server may process the tool parameter in at least one of the manners described below.

[0119] At step 204-3, the server extracts a time parameter from the tool parameters, and modifies relative time in the time parameter to absolute time.

[0120] For example, the time parameter extracted by the server may include user expressions in relative time, such as “last three days” or “last week”. To enhance the accuracy of time conversion, the server introduces a dedicated date tool. The tool can convert the relative time expressed by the user into a precise absolute time. For example, “last three days” is converted into a specific date range “start: 20240510, end: 20240512”. After completing the conversion of the time parameter, the server may use the updated tool parameter for subsequent tool invocation.

[0121] At step 204-4, the server extracts an order parameter from the tool parameters, and modifies a user expression in the order parameter to a standard expression in a predefined domain.

[0122] For example, the server may extract a parameter related to the order from the tool parameters, so that the user expression may be converted into the standard expression of the predefined domain. For example, the server may convert the user expression “order number

XX” into the standard expression “order serial number XX”. In this way, no matter how the user expresses the order information, the server can map the order information to a unified standard expression to ensure correct tool invocation.

5 [0123] In the foregoing implementation, standardizing the order parameters can implement effective matching between the user expression and the tool invocation. This can not only improve the accuracy of tool invocation, but also reduce the risk of misoperation caused by user expression differences.

10 [0124] In an optional implementation, the tools in the tool execution plan may include at least one of a knowledge base search tool, an information query tool, a function configuration tool, or an information parsing tool. In this embodiment, by pre-constructing the knowledge base and the knowledge base search tool, the reply content can be limited to the predefined domain, thereby reducing the risk of errors in the reply content. In addition, by combining a plurality of types of tools such as an information query tool, a function configuration tool, and an information parsing tool, a plurality of types of functions such as
15 information query, information parsing, and function configuration can be implemented, thereby enriching the functionality of the task-oriented dialogue. It may be understood that, the specific functions implemented by each tool and the scope of application domain are not specifically limited in this embodiment.

20 [0125] In an optional implementation, a process in which the server invokes the tools in the tool execution plan based on the tool parameters to obtain the plan execution result may be implemented in the following manner: invoking, by the server, tools arranged in sequence in the tool execution plan sequentially, and tools arranged in parallel in the tool execution plan in parallel based on the tool parameters to obtain the plan execution result.

25 [0126] For example, the server may invoke the tools arranged in sequence sequentially and invoke the tools arranged in parallel in parallel according to the execution order and the dependency relationship of the tools. During the invocation process, the server may also monitor the execution state of the tools in real time, and intervene and adjust it when necessary. Finally, the server may collect execution results of all tools, and generate a comprehensive plan execution result.

[0127] At step 205, the server performs a check on the plan execution result based on the dialogue information and the context information.

[0128] For example, after obtaining the plan execution result, the large model may perform a check on the completeness of the plan execution result, and determine whether the plan execution result passes the check based on the completeness of the plan execution result. If the check on the plan execution result is passed, the plan execution result may be input into the next stage to generate the reply content; if the check on the plan execution result is not passed, a tool needs to be re-selected through the planning formulation module, and the tool execution plan is updated, and then the updated tool execution plan is executed to obtain the updated plan execution result, and then whether the updated plan execution result is complete is determined. This process may continue to loop until a modified plan execution result reaches the completeness requirement and/or the number of loops reaches a certain threshold, then the plan execution result is output, and finally the reply content is generated through the updated plan execution result. In this embodiment, the large model may determine the completeness of the plan execution result according to the content of the dialogue information. The plan execution result is determined to be complete when it is determined that at least one of the following two conditions is met: the tools related to the dialogue information have all been invoked, and/or, the plan execution result can solve the problems raised in the dialogue information. Then, the plan execution result may be output and used to generate the reply content. The check process of the plan execution result may be implemented by a checking and verification module (Checker) in the E-Agent architecture.

[0129] In an optional implementation, step 205 may be implemented in the manner described below.

[0130] At step 205-1, the server acquires context information, and performs a check on each tool in the tool execution plan and the plan execution result based on the dialogue information and the context information.

[0131] At step 205-2, in response to determining that no tool is missing from the tool execution plan and/or the plan execution result is complete, the server determines that the plan execution result passes the check.

[0132] For example, when checking the plan execution result, the server may first acquire context information related to the dialogue information, and then check completeness of the tool in the tool execution plan and the plan execution result according to the acquired context information and dialogue information. The check process mainly involves two aspects: the first is to confirm whether the tool execution plan includes all necessary tools, with none missing; and the second is to evaluate whether the execution result of the plan is complete and can meet the requirement of generating the reply content. As long as at least one of the foregoing two conditions is satisfied, it can be determined that the plan execution result passes the check.

[0133] In this embodiment, the reply content is limited within the scope of the tool execution result, the tool execution result is obtained by invoking the tool, and the selection of the tool is limited in a scope, so that after all tools related to the dialogue information and the context information are invoked, the generated plan execution result may still fail to meet the requirement of generating the reply content. In this situation, if it is checked that the tool execution plan already includes all necessary tools, it is considered that the large model has reached the boundary of problem solving, and even if the problem proposed by the user is not solved at this time, it can still be considered that the plan execution result passes the check, so as to avoid that the large model randomly expands other content to exceed the topic boundary of the predefined domain, causing errors and deviations in the reply content. In some other situations, if it is checked that there is a tool that is not invoked, but the plan execution result can meet the requirement of generating the reply content, it may also be considered that the plan execution result passes the check. In this way, the consumption of computing resources can be reduced, the system delay can be reduced, and the timeliness of problem solving can be improved. Based on this, as long as one of the foregoing two conditions is met, it can be considered that the plan execution result passes the check.

[0134] In the foregoing implementation, the check is performed on the tool execution plan through the two aspects: the completeness of the tool invocation and the completeness of the tool invocation result, so that the completeness of the plan execution result can be ensured, thereby avoiding a problem of a reply error or information loss caused by tool missing or incomplete execution result. The foregoing method not only improves the user experience, but also enhances the reliability and stability of the system. In addition, in the

foregoing method, the problem can be identified and corrected in an early stage of problem occurrence, thereby reducing a probability of an error in reply content.

[0135] In an optional implementation, step 205 may further include the steps described below.

5 [0136] At step 205-3, in response to determining that a tool is missing from the tool execution plan and/or the plan execution result is incomplete, the server cyclically executes following process until a modified plan execution result passes the check, and/or a number of checks on the plan execution result reaches a preset threshold: generating, by the server, feedback information based on the tool execution plan and a check result of the plan
10 execution result, modifying the tool execution plan based on the feedback information and the intent information, and executing the modified tool execution plan to obtain the modified plan execution result.

[0137] For example, when determining that the tool execution plan does not include all necessary tools, or the plan execution result is incomplete, the server may determine that the
15 check on the plan execution result is not passed. In this case, the checker may generate text format information (i.e., feedback information) including the thought and the feedback signal, and feed the information back to the planner module. After receiving the feedback information, the planner module may reselect or adjust a tool (generally select a tool in an incremental manner) based on the feedback information and the intent information of the
20 user, and generate a modified tool execution plan. This modified plan is then passed to the caller module for execution and a new plan execution result is generated. Then, the new plan execution result is sent to the checker module again to be checked. This process will be repeated until the check by the checker module is passed, i.e., it is confirmed that the information is complete and sufficient to generate the reply content, or the number of checks
25 on the plan execution result reaches the preset threshold upper limit.

[0138] In this embodiment, if it is checked that the tool execution plan does not include all necessary tools, it is considered that the large model has not reached the boundary of problem solving, and at this time, regardless of whether the plan execution result can solve the problem raised by the user, it can be considered that the check on the plan execution
30 result is not passed, thereby improving the comprehensiveness of the plan execution result.

In some other situations, if it is checked that the plan execution result cannot meet the requirement for generating the reply content, it may also be determined that the check on the plan execution result is not passed, and the problem solving path is searched again through a tool, to ensure validity of problem solving. Based on this, as long as one of the foregoing two conditions is met, it can be considered that the check on the plan execution result is not passed. It may be understood that, the specific condition of whether the planned execution result passes the check may be designed according to the actual form of the product, which is not specifically limited in this embodiment.

[0139] In the foregoing implementation, the tool selection and the execution plan are intelligently adjusted based on the feedback information of the checker and the intent of the user, to implement automatic completion of information, so that the system can maintain high accuracy and response speed when facing a complex and changeable dialogue scenario. In addition, by setting the threshold upper limit of the number of checks, the system can control the number of cycles to a certain extent, avoiding infinite cycles caused by special situations, thereby enhancing the robustness of the system, and also improving the response timeliness of the system.

[0140] In an optional implementation, the method for modifying the tool execution plan based on the feedback information and the intent information in step 205-3 may be implemented in the following manner: selecting, by the server, at least one tool based on the feedback information and the intent information, and adding the at least one selected tool to the tool execution plan to obtain the modified tool execution plan.

[0141] For example, in response to determining that a tool is missing from the tool execution plan and/or the plan execution result is incomplete, the server may acquire the feedback information provided by the checker module and the intent information of the user, and select at least one tool from the predefined tool set again through the Planner module and add the at least one tool to the existing tool execution plan, thereby generating the modified tool execution plan. The large model may additionally select at least one tool to make up for the deficiency of the current tool execution plan or improve the current plan execution result based on a function of the tool and a requirement of an application scenario. Subsequently, the Planner module may submit the modified tool execution plan to the caller module for

execution, and then the checker module performs determining to obtain a more complete and accurate plan execution result.

[0142] In this embodiment, the re-selected tool may be added to the original tool execution plan in an incremental manner, or an incorrect tool selected by the original tool execution plan may be replaced, or an already selected tool (e.g., a knowledge base search tool) may be repeatedly invoked, so as to generate a new invocation result. In addition, the feedback information provided by the checker module may include a check result of a tool in the tool execution plan and/or a check result of a plan execution result, so that the Planner module may determine a tool that is missing or incorrect in the tool execution plan based on the feedback information, and supplement a new tool, thereby improving efficiency and accuracy of tool selection.

[0143] In the foregoing implementation, a new tool is intelligently selected and added to the tool execution plan based on the feedback information of the checker module and the intent information of the user, so that the system can flexibly adjust and optimize the tool execution plan for different situations, to ensure completeness and accuracy of information, and the system can maintain high accuracy and response speed when facing complex and changeable dialogue scenarios, thereby reducing a need for human intervention, reducing maintenance costs of the system, and improving overall performance and user experience of the system.

[0144] In an alternative embodiment, the plan execution result may be checked by assistance of a prompt template configured in the large model of the checking and verification module, for example, the content described below may be configured into the prompt template of the large model.

[0145] Task: You will receive a merchant's problem, context information, and a tool that the assistant has invoked and a result, and determine correctly whether the assistant has tried the best. Please always remember the tool list of the assistant (you can only refer to these tools for evaluation): {tool_hum_names}.

[0146] Task focuses:

[0147] 1. Understanding the Problem: You need to carefully understand the merchant's problem in order to assess whether the understanding of the assistant is accurate.

[0148] 2. assistant capability boundary: You must pay attention to the capability boundary of the assistant, and if a solution cannot be obtained through existing tools, it is considered that the assistant has made the best effort.

[0149] 3. Prohibiting tool retry: If a tool fails to be invoked, no retry is necessary.

5 [0150] Question: user problem; Observation: invoked tool and tool invocation result; Thought: you must use the dialectical thinking to think around the user problem, determine whether the assistant has made the best effort, and whether there is a tool related to the problem that has not been invoked, etc.; Check: Yes or No; whether the assistant has made the best effort within the capability range; Final Answer: when Check is No, please select
10 the correct tool from the tool list; when Check is Yes, reply with “none”.

[0151] In the foregoing embodiment, by configuring the checking steps, precautions and related cases of the plan execution result into the prompt template of the large model, the checking result of the plan execution result can be more comprehensive and accurate, with none missing.

15 [0152] At step 206, the server integrates the plan execution result that passes the check based on the dialogue information and the context information to obtain the reply content, and sends the reply content to the client.

[0153] For example, after the check on the plan execution result is passed, the large model may generate the reply content based on the plan execution result that passes the check,
20 where the reply content may be generated in combination with the dialogue information input by the user, i.e., when the reply content is generated, the content related to the dialogue information in the plan execution result may be retained, and the content unrelated to the dialogue information is removed, so as to obtain the reply content more related to the dialogue information. In this embodiment, the large model may also perform other forms of
25 processing on the reply content, e.g., adding reassurance information, adding link information, etc., which is not specifically limited in this embodiment. A process of generating the reply content may be implemented by a content summarization module (Summary) in the E-Agent architecture.

[0154] In an optional implementation, step 206 may be implemented in the following manner: acquiring, by the server, context information, and integrating, based on the dialogue information and the context information, content related to the dialogue information in the plan execution result to obtain the reply content.

5 [0155] For example, during content integration, the server may first acquire current dialogue information and context information, and then organize, for dialogue information sent by a user in a current round, results returned by a plurality of tools and content of a document block according to a specific logical order, to ensure coherence and accuracy of the information. In addition, the server may also exclude noise information unrelated to the
10 dialogue information of the current round, to prevent the noise information from interfering with the quality of the final answer. Finally, the server may convert the integrated information into a natural language, to generate reply content for the user problem.

[0156] In the foregoing implementation, the results returned by the plurality of tools and the content of the document block are integrated for the dialogue information of the user, to
15 generate the reply content for the dialogue information, so that a comprehensive, coherent, and accurate response can be provided for the user, thereby effectively improving accuracy and pertinence of the reply content. In addition, since the server can answer within the range defined by the tool invocation result based on the given dialogue information, no additional information is introduced, thereby ensuring the specialization of the reply content.

20 [0157] In an optional implementation, the reply content may be generated in at least one of the manners described below.

[0158] At step 206-1, the server deletes content unrelated to the dialogue information in the plan execution result, and integrates content remained in the plan execution result to obtain the reply content.

25 [0159] For example, when performing content integration, the server may parse the content in the plan execution result through a natural language processing technology, to identify content irrelevant to the dialogue information, and delete the content. The server may then logically integrate the remaining relevant content to form the reply content in coherent,

accurate natural language. The foregoing processing manner can ensure that the reply content can focus on a core demand of the user, and avoid introducing unnecessary noise.

[0160] In the foregoing implementation, irrelevant content is deleted, and remaining information is integrated to form reply content, so that pertinence and accuracy of the reply content can be effectively improved, and a risk of misleading a user caused by noise interference can be reduced. In addition, the foregoing processing manner can further improve simplicity and readability of the reply content, so that the user can quickly read required information.

[0161] At step 206-2, the server acquires at least one piece of link information based on the dialogue information, and adds the at least one piece of link information to the reply content.

[0162] For example, during performing content integration, if determining that the reference information or detailed information may be further added to the dialogue information, the server may acquire related link information (e.g., a picture link, a document link, and a web page link) according to the dialogue information, and add the link information to the reply content. In this embodiment, the link information may be a document link in the knowledge base or a link of an external website.

[0163] In the foregoing implementation, by adding the link information into the reply content, more reference information and detailed information can be provided for the user, so that the user can be more comprehensively and quickly learned of a solution to a related problem, thereby improving user experience and the timeliness of problem solving.

[0164] At step 206-3, in response to identifying that the dialogue information includes a negative emotion, the server generates reassurance information based on the dialogue information, and adds the reassurance information to the reply content.

[0165] For example, during performing content integration, when the server identifies that the dialogue information includes a negative emotion (e.g., complaint, dissatisfaction, etc.), the server may further generate reassurance information based on the dialogue information, e.g., apology information, explanation information, or suggestion information, etc., and add the reassurance information into the reply content.

[0166] In the foregoing implementation, the reassurance information is added to the reply content for the dialogue information initiated by the user, so that the negative emotion of the user can be alleviated as much as possible in the reply content, thereby improving user experience.

- 5 [0167] At step 206-4, in response to identifying that the dialogue information includes human service requirement information, the server generates guidance information based on the dialogue information, and adds the guidance information into the reply content.

[0168] For example, during performing content integration, if the server identifies that the dialogue information includes human service requirement information (e.g., the user requests to convert to a human service), the server may generate guidance information based on the dialogue content, e.g., providing contact information of a customer server, describing service content that can be provided by the task-oriented dialogue for the user, and add the guidance information into the reply content, to guide the user to quickly solve the problem.

10

[0169] In the foregoing implementation, by adding the guide information to the reply content, a more effective solution can be provided for the user, thereby reducing waiting time and communication costs of the user. In addition, the foregoing implementation can further improve a response speed and a problem solving capability of the server, to enhance user experience.

15

[0170] At step 206-5, in response to identifying that the dialogue information does not match a predefined domain, the server generates recognition rejection information based on the dialogue information, and determines the recognition rejection information as the reply content.

20

[0171] For example, during performing content integration, when the server identifies that the dialogue information does not match the predefined domain (e.g., in the merchant service domain, the user asks a problem unrelated to the merchant service content), the server may generate the recognition rejection information based on the dialogue content, and send the recognition rejection information as the reply content to the user. The recognition rejection information refers to related information that expresses that a problem proposed by a user

25

cannot be processed, and the information may be generated based on the dialogue information initiated by the user in a current round.

5 **[0172]** In the foregoing implementation, when the dialogue information does not match the current domain, the recognition rejection information is added to the reply content, so that answering a problem inconsistent with the predefined domain can be avoided, thereby reducing confusion and misunderstanding of the user.

[0173] At step 206-6, the server integrates content related to the dialogue information in the plan execution result under a constraint condition of predefined word count and language style to generate structured reply content.

10 **[0174]** For example, when performing content integration, the server may integrate and segment the content related to the dialogue information in the plan execution result under the constraint condition of the preset word count and language style, thereby obtaining the structured reply content. The structured reply content may include the respective parts such as the main text divided into paragraphs according to themes and the conclusion, following
15 a certain logical structure and a language style to arrange.

[0175] In the foregoing implementation, the structured reply content is generated according to a specific word count requirement and language style, so that clear and readable reply content can be provided for the user, thereby helping the user quickly understand a solution.

20 **[0176]** At step 206-7, the server retrieves at least one case from a reply content experience repository through a predefined information retrieval model based on the intent information, and configures the case into a prompt template to determine the reply content based on the prompt template.

25 **[0177]** For example, the server may, according to the intent information corresponding to the dialogue information, retrieve at least one case related to the current intent from a reply content experience repository through a predefined information retrieval model (e.g., at least one of a sparse retrieval model, a dense vector retrieval model, or a multi-vector retrieval model), and configure the retrieved case into a prompt template of the large model, to determine the final reply content based on the prompt template.

[0178] In the foregoing implementation, similar cases are retrieved based on the intent information, and the reply content is determined based on the retrieved cases, so that more personalized and targeted reply content can be provided for the user. In addition, in the foregoing implementation, information in historical data and an experience repository can be fully used, to improve accuracy and reliability of reply content.

[0179] In an optional implementation, the reply content may be generated by assistance of a prompt template set in a large model of the content summarization module, for example, the content described below may be configured into the prompt template of the large model.

[0180] Task Goals:

[0181] 1. For the problems related to the non-delivery domain proposed by merchant, you need to euphemistically reject the answer.

[0182] 2. For a merchant's problem, you need to combine known information and the merchant's emotion to make a humanized summary answer according to the information.

[0183] 3. For a merchant's problem, you need to combine the problem with historical dialogues for judgement together.

[0184] 4. Guided consultation: When merchants attempt to contact an account manager or customer server, you need to actively guide them to ask you directly, and make every effort to handle and solve their problems.

[0185] 5. The output content needs to be segmented hierarchically, which is convenient for merchant to read quickly.

[0186] ninput: <history_dialogue>: the historical dialogue between the merchant and the assistant; <tools>: the currently invoked tools and results; <Links You Can Quote>: the list of links you may quote; <Query>: the content currently input by the merchant;

[0187] output: Thought: first determining whether the merchant has a negative emotion from <Query>, and then referring to the contents of <history_dialogue>, <tools>, <Links You Can Quote>; and providing a summary reply;

[0188] Final Answer: when the merchant has negative emotions, first providing reassurance; then offering a reply for the merchant.

[0189] Note:

[0190] * When referring to order number XX, it refers to order serial number XX.

5 [0191] * When Action and Observation in <tools> are not empty, your Thought must be reasonable and concise.

[0192] * Screening out only the content related to the current problem of the merchant from the result of Observation for answering;

10 [0193] * <knowledge> may include some links and pictures, and the information will be given in the form of <link-1> and <picture-1>, which are also important information, and the original formats such as <link-1> and <picture-1> are directly kept in the answer for quote; and do not make up <link>, <link>, <knowledge>, pictures, etc., for answer.

15 [0194] * Final Answer requires conciseness and clarity, please strictly screen out the appropriate content related to the merchant's problem from the given Observation to answer the problem, which cannot be answered based on one's own knowledge, and does not provide suggestions or practices beyond "Observation".

[0195] Case:

20 [0196] ninput: <history_dialogue>: empty; <tools>: Action: knowledge_retrieval_tool, Observation: the customer service telephone number of the delivery platform is 10105757. <Links You Can Quote>:[]

[0197] <Query>: human service?

25 [0198] noutput: Thought: if the merchant has no negative emotion, I will answer according to the merchant's problem; if the merchant wants to find a human customer service, my responsibility is to guide them to ask me, so I guide the merchant to ask me while providing a phone number. Final Answer: the customer service telephone number of the delivery is

10105757, if you have any problems on the delivery platform, welcome to ask me, and I will try my best to help you answer.

[0199] In an optional implementation, the reply content may be generated by a pre-trained large model, where the pre-training method of the large model may be implemented in the following manner: generating, by the server, a plurality of types of information reply cases based on a plurality of predefined dialogue scenarios in a predefined domain; and then, performing parameter fine-tuning on a predefined large model based on the plurality of types of information reply cases to obtain a pre-trained large model. The predefined large model may be a large model in a general domain or a large model obtained after continued pre-training with a sample dataset in a predefined domain.

[0200] For example, when performing model training, the server may construct a plurality of types of information reply cases based on a plurality of predefined dialogue scenarios in a predefined domain, where the constructed cases may cover different user intents, problem types, and reply styles, to ensure that the large model can adapt to various complex dialogue environments. Subsequently, the server may use these constructed information reply cases to perform parameter fine-tuning on the predefined large model. In this embodiment, the base model of the large model may be a large model in a general domain, or may be a vertical domain model obtained after continued pre-training based on a sample dataset in a predefined domain. In the process of parameter fine-tuning, the server may further adjust parameters of the model by utilizing an optimization algorithm, to minimize an error between the predicted reply and the real cases, thereby improving reply accuracy of the model in the predefined domain.

[0201] At step 207, the client receives and displays the reply content.

[0202] For example, after generating the reply content, the server may send the reply content to the client, and the client may display the reply content in response to receiving the reply content. For example, the client may display the reply content through various pages such as an instant messaging page and a search result page. In this embodiment, when outputting the reply content, the server may further output other information, for example, the server may further output the user's dialogue information and intent information, the tool names and tool invocation results, and the thought information of the large model, etc. In this

scenario, the server may send only part of the information output by the large model to the client for display, for example, the server may send only the reply content to the client, and then record other information in a log, so as to facilitate future analysis and processing.

[0203] In an optional implementation, the client may send feedback information of the reply content to the server. The feedback information may include positive feedback information and negative feedback information. In this embodiment, the positive feedback information and the negative feedback information may be generated through the triggering operation of the feedback control and/or the evaluation content from the user regarding the reply content. After receiving the feedback information sent by the client, the server may perform a corresponding operation (e.g., adding a new case, increasing an experience score to the case, reducing an experience score of the case, and deleting an existing case, etc.) on the cases in the experience repository based on the feedback information, to improve accuracy and effectiveness of the cases in the experience repository.

[0204] At step 208, the server generates at least one type of case, and stores the case in a corresponding experience repository according to a type of the case.

[0205] In this embodiment, the cases stored by the server may include at least one type of cases of an intent recognition case, a tool invocation case, or an information reply case. The intent recognition case may be stored in an intent information experience repository, the tool invocation case may be stored in an execution plan experience repository, and the information reply case may be stored in a reply content experience repository. The foregoing plurality of types of cases may be added, modified, and deleted at various stages such as before training of the large model, in a training process of the large model, and in a use process of the large model, to ensure that the cases in the experience repository can cover a plurality of dialogue scenarios in the predefined domain as many as possible, and can resolve a real problem faced by the user in the predefined domain, thereby improving a speed and quality of problem solving.

[0206] In this embodiment, the cases in the experience repository may include two types: a result direct use type and a sample type. The result direct use type refers to that when the problem proposed by the user hits a certain type of case, the case can be directly used, and at this time, it is not necessary to invoke the large model to regenerate the results such as the

tool execution plan and the reply content, which can effectively reduce the computing resources; the sample type refers to that it can be put into the context of the prompt template of the large model as few-shot (a small number of samples) to guide the large model to generate the content such as the intent information, the tool execution plan and the reply content according to the content in the prompt template and the samples configured in the prompt module. In this embodiment, each of the stages (including the intent recognition module, the planning formulation module, and the content summarization module) of the E-Agent has its own independent experience repository, and each of the experience repositories stores the corresponding cases. These cases can be used to guide the large model of each stage to generate corresponding content such as intent information, a tool execution plan, and reply content, so as to improve the speed and quality of generating a tool execution plan and reply content.

[0207] In this embodiment, the case in the experience repository may be stored through a K-V structure. V refers to the specific content of the direct use type case or the sample type case corresponding to the stage where the case is located, and K refers to the single-round core intent information output in the intent extraction stage. The exception is that the experience K in the intent extraction stage needs to be specially processed, which may be specifically formed by splicing dialogue information of users in at least one recent round.

[0208] Further, in each stage of task-oriented dialogue implementation, if a case in the experience repository is matched, it indicates that the current scenario is a familiar and experienced scenario, and at this time, experience data (i.e., a case) may be used. In this embodiment, a case recall process may be considered as a retrieval matching task, and a core thereof is how to efficiently and accurately determine whether a current scenario accurately matches a case in a database, which has characteristics of extremely high performance and matching precision requirements. Based on this, in this embodiment, retrieval model integration (HybridRetrieval) is used to perform information retrieval and case recall by combining three retrieval models: sparse retrieval (BM25), dense vector retrieval (BGE), and multi-vector retrieval (Multi-Vector). The architecture of the information retrieval model is illustrated in FIG. 5, the retrieval effect of the information retrieval model is illustrated in FIG. 6. It can be seen from FIG. 6 that when the selected threshold is greater than 0.79, the accuracy rate of hit@1 (an evaluation index in the information retrieval task,

which refers to under a given user request, whether the first result (top-1 result) returned by the system hits the goal or requirement of the user) can be stabilized above 82%, and at the same time, the recall rate can be stabilized at 93%, and the performance requirement of controlling time consumption within 500ms can be met.

5 **[0209]** Further, in the aspect of case generation, as illustrated in FIG. 4, the cases in the experience repository are mainly generated and optimized in two manners: expert-oriented (i.e., full-process feedback in FIG. 4) and user-oriented (i.e., result-oriented feedback in FIG. 4). In the expert-oriented aspect, overall guidance may be performed in the process of generating the reply content, and improvement and learning may be performed in real time
10 to accumulate experience solutions; and in the real-user-oriented aspect, result-oriented feedback from online users can be used to influence the experience value of the scheme in real time.

[0210] In an optional implementation, the case may be generated in at least one of the manners described below.

15 **[0211]** At step 208-1, intent information, a work execution plan, and reply content generated during interaction of the user with the large model are received, and a feedback operation is performed on the intent information, the work execution plan, and the reply content based on first positive and negative feedback information from the user regarding the intent information, the work execution plan, and the reply content to obtain the at least one type of
20 case.

[0212] For example, a user (i.e., an expert) in the predefined domain may generate and optimize a case by asking problems and interacting with a task-oriented dialogue model based on an E-Agent architecture, and the expert can observe an intermediate process and a final reply of the Agent in real time in this process. For example, in the interaction process,
25 if there is an error in the reply or intermediate process of the Agent, the expert can immediately indicate the problem and provide an improvement suggestion. Then, the server may generate corresponding feedback information through a large model or human intervention according to the feedback of the expert. Further, the expert may initiate a retry request for incorrect information, and during the retry, the server may add the feedback
30 information of the previous round to the Prompt requested by the large model, initiate the

invocation again, and then repeat this process until the expert is satisfied. Finally, the server may extract sample type and direct use type experience from the large model input, output and feedback information of each stage, and store the sample type and direct use type experience in the experience repository.

5 **[0213]** In the foregoing implementation, through user whole process guidance and real-time feedback in a predefined domain, a problem of a model in a reply content generation process can be accurately identified and improved, to ensure high quality and accuracy of a generated case. In addition, by continuously accumulating expert experience, the system can continuously optimize the model performance and improve the adaptability of the model in
10 similar scenarios. In addition, by extracting and storing sample type and direct use experience, the system can quickly respond to new scenarios and new problems, and improve the content reply quality of task-oriented dialogues.

[0214] At step 208-2, second positive and negative feedback information from the user regarding the reply content is received, and a feedback operation is performed on intent
15 information, a work execution plan, and reply content generated in a process of generating the reply content based on the second positive and negative feedback information to obtain the at least one type of case.

[0215] For example, the server may further receive positive and negative feedback information of the online user for the reply content in real time. For example, in a process
20 in which the online user interacts with the large model, the server may receive positive and negative feedback information of the online user for the reply content in real time. The user may generate positive and negative feedback information of the reply content in a plurality of manners such as clicking dislike, clicking like, and sending a comment, so that the server may perform a positive and negative feedback operation on a corresponding case based on
25 the feedback information made by the user, for example, adding or reducing cases, or increasing or decreasing the experience scores of cases. This method based on real-time user feedback can quickly capture user requirements and continuously optimize the case repository.

[0216] In the foregoing implementation, the positive and negative feedback information of
30 the user is received and processed in real time, so that the user requirement can be quickly

identified, and the case repository can be optimized and updated accordingly. This manner can significantly reduce the negative impact of incorrect experiences on user experience, and can improve user satisfaction. In addition, by collecting the positive and negative feedback information of the user for the reply content, the server can automatically identify and store valuable user experience, so that the case repository can be further enriched and improved, thereby improving the quality of service of the task-oriented dialogue.

[0217] In an optional implementation, referring to the process of generating and optimizing cases in a process-oriented feedback manner in FIG. 4, step 208-1 may be implemented in at least one of the manners described below.

[0218] At step 208-1-1, in response to receiving positive feedback information from the user regarding the intent information, the work execution plan, and the reply content, the server performs a positive feedback operation on a case corresponding to the intent information, the work execution plan, and the reply content corresponding to the positive feedback information, where the positive feedback operation includes increasing an experience score of the case, and/or, adding a new case.

[0219] For example, in response to receiving positive feedback information from a user regarding the intent information, the work execution plan, and the reply content, the server may first identify and locate a case corresponding to the positive feedback information. Then, the system may perform a positive feedback operation on the case, and the specific operation includes but is not limited to increasing the experience score of the case, so as to increase the priority of the case in subsequent similar problems. In addition, the server may further merge or integrate similar or related positive feedback cases based on specific content of the positive feedback information, to generate a new case.

[0220] In the foregoing implementation, a positive feedback operation performed by the user on the case is collected, so that a case that the user is satisfied can be quickly identified and strengthened, thereby improving quality and availability of the case repository. In addition, the newly generated case can further enrich the content of the case repository to improve the adaptability and flexibility of the system.

[0221] At step 208-1-2, in response to receiving negative feedback information from the user regarding the intent information, the work execution plan, and the reply content, the server performs error correction processing on the intent information, the work execution plan, and the reply content corresponding to the negative feedback information to generate error correction information, and configures the error correction information into a prompt template to regenerate intent information, a work execution plan, and reply content based on the prompt template.

[0222] For example, in response to receiving negative feedback information from a user regarding the intent information, the work execution plan, and the reply content, the server may first analyze a specific cause and content of the negative feedback, and locate an erroneous case. Then, error correction processing may be performed on the erroneous case, and error correction information is generated. The error correction information may include a specific description of error information, a correct processing manner, a recommended improvement direction, or the like. Then, the server may set the error correction information in a prompt template of the large model, and the template may be used to guide subsequent case generation. Finally, the large model may regenerate the intent information, the work execution plan, and the reply content based on the prompt template, to ensure that the new content can avoid a previous error, thereby improving user satisfaction.

[0223] In an optional implementation, the method for regenerating intent information, a work execution plan, and reply content based on the prompt template in step 208-1-2 may be implemented in the following manner: in response to identifying that a case corresponding to intent information, a work execution plan, and reply content that are generated through the case is erroneous, replacing, by the server, the erroneous case when regenerating intent information, a work execution plan, and reply content, and performing a negative feedback operation on the erroneous case, where the negative feedback operation includes decreasing an experience score of the case and/or deleting the case.

[0224] For example, when the server identifies that an error exists in the intent information, the work execution plan, and the reply content that are generated through the case, the server may first identify the erroneous case, and determine at which stage of generating intent information, the work execution plan, and the reply content the case has an error. Then, the server may replace the erroneous case, and based on other correct or more suitable cases, re-

perform intent recognition, formulation of work execution plan, and generation of reply content. In the process of regeneration, the server may also perform a negative feedback operation on the erroneous case. These negative feedback operations include, but are not limited to, reducing the experience score of the case to reduce its priority in future similar situations; in specific situations, if the error of the case is severe or similar errors occur multiple times, the server can also choose to delete the case to avoid the case affecting the performance of the system.

[0225] In an optional implementation, referring to the process of generating and optimizing cases in a result-oriented feedback manner in FIG. 4, step 208-2 may be implemented in at least one of the manners described below.

[0226] At step 208-2-1, in response to receiving positive feedback information from the user regarding the reply content, the server performs a positive feedback operation on a case corresponding to intent information, a work execution plan, and reply content that are generated through cases, where the positive feedback operation includes increasing an experience score of the case.

[0227] For example, in response to receiving positive feedback information from a user regarding the reply content, the server may locate the case corresponding to the reply content, i.e., locate the case corresponding to the intent information, the work execution plan, and the reply content that are generated through the case, and then the server may perform a positive feedback operation on these cases, mainly including increasing experience scores of the cases. Increasing the experience score of the case can increase the priority of the case in future similar scenarios, so that the system is more inclined to use a case with a higher experience score to generate the reply content.

[0228] At step 208-2-2, in response to receiving positive feedback information from the user regarding the reply content, the server adds intent information, a work execution plan, and the reply content that are not generated through cases as a new case.

[0229] For example, in response to receiving positive feedback information from a user regarding the reply content, if the intent information, the tool execution plan, and the reply content are not generated through cases, but are regenerated through a large model based on

the intent information, the server may add the newly generated intent information, work execution plan, and reply content to the case repository as a brand-new case. In this way, the system can continuously learn and accumulate new knowledge and experience to cope with more diverse and complicated user requirements.

- 5 **[0230]** At step 208-2-3, in response to receiving negative feedback information from the user regarding the reply content, the server performs a negative feedback operation on a case corresponding to intent information, a work execution plan, and reply content that are generated through cases, where the negative feedback operation includes decreasing an experience score of the case and/or deleting the case.
- 10 **[0231]** For example, in response to receiving negative feedback information from a user regarding the reply content, the server may locate corresponding cases, and then perform a negative feedback operation on these cases, mainly including reducing experience scores of the case, and in addition, in some situations, may also delete these cases. Reducing the experience scores of the cases can reduce the probability of these cases being selected in the
- 15 future, and deleting a case directly removes potential erroneous or low-quality cases, thereby helping to improve the overall performance and accuracy of the system.

[0232] At step 208-2-4, in response to receiving negative feedback information from the user regarding the reply content, the server deletes intent information, a work execution plan, and reply content that are not generated through cases.

- 20 **[0233]** For example, in response to receiving negative feedback information from a user regarding the reply content, if the intent information, the tool execution plan, and the reply content that correspond to the reply content are not generated through cases, the server may directly delete the intent information, the work execution plan, and the reply content, to help to maintain cleanliness and efficiency of the case repository, and avoid storing invalid or
- 25 low-quality content.

[0234] In the foregoing implementation, cases in an experience repository are dynamically optimized and adjusted based on real-time feedback of a user, so that system performance and user experience can be improved. First, in the foregoing manner, an experience score of a case is increased through a positive feedback operation, and a high-quality case can be

more accurately identified and used to generate a reply, thereby improving accuracy and reliability of the reply. In addition, the negative feedback operation can reduce or delete low-quality cases to reduce the probability of error reply. Moreover, by adding new cases and deleting invalid cases, the system can continuously learn and accumulate new knowledge and experience to meet more diversified and complex user requirements, so that the system has stronger adaptability and flexibility, thereby better meeting personalized needs of users. In addition, by optimizing the case repository, the system can generate more accurate and useful reply, thereby improving the user experience. Meanwhile, the user feedback can also be fully utilized, enhancing the interactivity between the user and the system.

5 [0235] It should be noted that user information (including but not limited to user equipment information, user personal information, etc.) and data (including but not limited to data for analysis, stored data, displayed data, etc.) involved in the present disclosure are both information and data authorized by the user or fully authorized by parties. In addition, the reference numbers corresponding to the steps in the foregoing embodiments are only for
10 identification, and are not intended to limit the execution order of the steps, and the execution order of the steps in the embodiments may be configured according to actual situations.

[0236] Further, as a specific implementation of the methods illustrated in FIG. 1 to FIG. 6, an embodiment of the present disclosure provides a task-oriented dialogue apparatus. As
15 shown in FIG. 7, the apparatus includes:

[0237] a dialogue information sending module 31, configured to send dialogue information in response to receiving the dialogue information initiated by a user; and

[0238] a reply content receiving module 32, configured to receive and display reply content, where a generation process of the reply content includes: determining intent information of
20 the user through a large model based on the dialogue information, generating a tool execution plan according to the intent information, executing the tool execution plan to obtain a plan execution result, performing a check on the plan execution result, and generating reply content based on a plan execution result that passes the check.

[0239] Further, as a specific implementation of the methods illustrated in FIG. 1 to FIG. 6, an embodiment of the present disclosure provides a task-oriented dialogue apparatus. As shown in FIG. 8, the apparatus includes:

[0240] an intent recognition module 41, configured to: in response to receiving dialogue
5 information initiated by a user, determine intent information of the user through a large model;

[0241] a planning formulation module 42, configured to generate a tool execution plan according to the intent information;

[0242] an invocation execution module 43, configured to execute the tool execution plan to
10 obtain a plan execution result;

[0243] a checking and verification module 44, configured to perform a check on the plan execution result; and

[0244] a content summarization module 45, configured to generate a reply content based on the plan execution result that passes check.

[0245] It should be noted that, for other corresponding descriptions of the functional units involved in the task-oriented dialogue apparatus provided in the embodiments of the present disclosure, reference may be made to the corresponding descriptions in the methods in FIG. 1 to FIG. 6, and details are not described herein again.

[0246] An embodiment of the present disclosure further provides a computer device, which
20 may be a personal computer, a server, a network device, etc., and the computer device includes a bus, one or more processors, a memory, and a communication interface, and may also include an input/output interface and a display device. The processor of the computer device is configured to provide computing and control capabilities. The memory of the computer device includes a non-volatile storage medium and an internal memory. The non-
25 volatile storage medium stores an operating system, a computer program, and a database. The internal memory provides an environment for running the operating system and the computer program in the non-volatile storage medium. The database of the computer device is configured to store location information. The network interface of the computer device is

configured to communicate with an external terminal through a network connection. The computer program is executed by the one or more processors to implement the steps in the method embodiments.

[0247] A person skilled in the art may understand that the foregoing structure of the computer device is merely a part of structure related to the solutions of the present disclosure, and does not constitute a limitation on the computer device to which the solutions of the present disclosure are applied. A specific computer device may include more or less components, or combine some components, or have different component arrangements.

[0248] In an embodiment, a computer-readable storage medium is provided, where the computer-readable storage medium may be non-volatile or volatile, and stores a computer program, and when the computer program is executed by one or more processors, the steps in the foregoing method embodiments are implemented.

[0249] In an embodiment, a computer program product is provided, including a computer program, where when the computer program is executed by one or more processors, the steps in the foregoing method embodiments are implemented.

[0250] A person of ordinary skill in the art may understand that all or some of the processes in the methods of the foregoing embodiments may be implemented by instructing related hardware through a computer program, the computer program may be stored in a non-volatile computer-readable storage medium, and when executed, the computer program may implement the processes of the foregoing method embodiments. Any reference to a memory, a database, or another medium used in the embodiments provided in the present disclosure may include at least one of a non-volatile memory or a volatile memory. The non-volatile memory may include a read-only memory (ROM), a magnetic tape, a floppy disk, a flash memory, an optical memory, a high-density embedded non-volatile memory, a resistive random access memory (ReRAM), a magnetoresistive random access memory (MRAM), a ferroelectric random access memory (FRAM), a phase change memory (PCM), a graphene memory, etc. The volatile memory may include a random access memory (RAM) or an external cache memory. By way of illustration and not limitation, the RAM may be in various forms, such as a static random access memory (SRAM) or a dynamic random access memory (DRAM). The database in the embodiments provided in the present disclosure may

include at least one of a relational database or a non-relational database. The non-relational database may include a blockchain-based distributed database, and is not limited thereto. The processor involved in the embodiments provided by the present disclosure may be a general-purpose processor, a graphics processor, a digital signal processor, a programmable logic device, a data processing logic device based on quantum computing, etc., and is not limited thereto.

[0251] The technical features of the foregoing embodiments may be combined arbitrarily, and to make the description concise, all possible combinations of the technical features in the foregoing embodiments are not described, however, as long as there is no contradiction between these combinations of technical features, they should be considered as the scope recorded in the present specification.

[0252] The foregoing embodiments only express several implementations of the present disclosure, and the description thereof is more specific and detailed, but it should not be construed as limiting the scope of the present disclosure. It should be noted that a person of ordinary skill in the art may further make several variations and improvements without departing from the concept of the present disclosure, which all fall within the protection scope of the present disclosure. Therefore, the protection scope of the present disclosure shall be subject to the appended claims.

CLAIMS

1. A task-oriented dialogue method, comprising:
 - in response to receiving dialogue information initiated by a user, sending, by a client,
 - 5 the dialogue information to a server;
 - in response to receiving the dialogue information, determining, by the server, intent information of the user through a large model, generating a tool execution plan according to the intent information, executing the tool execution plan to obtain a plan execution result, performing a check on the plan execution result, generating reply content based on the plan
 - 10 execution result that passes the check, and sending the reply content to the client; and
 - receiving and displaying, by the client, the reply content.
2. The method according to claim 1, wherein in response to receiving the dialogue information, determining, by the server, the intent information of the user through the large
- 15 model comprises:
 - in response to receiving the dialogue information, acquiring, by the server, context information and predefined application scenario information corresponding to the dialogue information; and
 - determining, by the server, the intent information of the user based on the dialogue
 - 20 information and the context information by using the application scenario information as a constraint condition.
3. The method according to claim 2, wherein determining, by the server, the intent information of the user based on the dialogue information and the context information by
- 25 using the application scenario information as the constraint condition comprises:
 - rewriting, by the server, the dialogue information and the context information within a topic scope constrained by the application scenario information to obtain at least one piece of intent information of the user; and/or
 - retaining, by the server, information related to the application scenario information
 - 30 from the dialogue information and the context information, and generating at least one piece of intent information according to a theme of the retained content; and/or
 - retaining, by the server, time information and numerical information from the

dialogue information and the context information in the intent information of the user; and/or
replacing, by the server, referential information from the dialogue information and the context information with name information, and retaining the name information in the intent information of the user; and/or

5 in response to identifying that the context information comprises a differentiated topic different from the dialogue information, deleting, by the server, content related to the differentiated topic from the intent information; and/or

in response to determining that the dialogue information and the context information are not within the topic scope constrained by the application scenario information, using, by
10 the server, the dialogue information as the intent information of the user; and/or

retrieving, by the server, at least one case from an intent information experience repository through a predefined information retrieval model based on the dialogue information and the context information, and configuring the at least one case into a prompt template to determine the intent information of the user based on the prompt template,
15 wherein the information retrieval model comprises at least one of a sparse retrieval model, a dense vector retrieval model, or a multi-vector retrieval model.

4. The method according to any one of claims 1 to 3, wherein the intent information is generated by a pre-trained large model, and a pre-training method for the large model
20 comprises:

performing, by the server, continued pre-training on a large model in a general domain based on a sample dataset in a predefined domain, wherein the sample dataset is provided with random noise and does not comprise sample user information;

performing, by the server, parameter fine-tuning on the large model after the
25 continued pre-training based on an intent recognition case in the predefined domain; and

generating, by the server, intent recognition information of a predefined problem in the predefined domain based on the fine-tuned large model, and iteratively updating a parameter of the fine-tuned large model through a reinforcement learning algorithm based on positive and negative feedback information of the intent recognition information to obtain
30 the pre-trained large model.

5. The method according to claim 1, wherein generating, by the server, the tool

execution plan according to the intent information comprises:

acquiring, by the server, context information, and according to the context information, searching a tool library for a plurality of candidate tools related to the intent information to generate a candidate tool list; and

5 selecting, by the server, at least one tool from the candidate tool list based on the intent information, and arranging and combining the at least one tool to obtain the tool execution plan.

6. The method according to claim 5, wherein selecting, by the server, the at least one
10 tool from the candidate tool list based on the intent information comprises:

performing, by the server, keyword matching on the candidate tool list based on the intent information to select at least one tool; and/or

in response to determining that intent information of a current round has a same meaning as intent information of a previous round, determining, by the server, a tool
15 different from a tool of the previous round as a selected tool of the current round; and/or

determining, by the server, a tool mutually exclusive with an already selected tool as a disabled tool of the current round; and/or

in response to determining that no tool is selected based on the intent information, determining, by the server, a knowledge retrieval tool as a selected tool of the current round;
20 and/or

retrieving, by the server, at least one case from an execution plan experience repository through a predefined information retrieval model based on the intent information, and selecting at least one tool from the candidate tool list based on the at least one case, wherein the information retrieval model comprises at least one of a sparse retrieval model,
25 a dense vector retrieval model, or a multi-vector retrieval model; and

selecting, by the server, the at least one tool from the candidate tool list based on the at least one case comprises:

in response to determining that a matching degree between a case and the intent information is greater than a preset threshold, determining, by the server, a
30 tool in the case as a selected tool; and/or

in response to determining that the matching degree between the case and the intent information is less than or equal to the preset threshold, configuring, by

the server, the case into a prompt template to select the at least one tool from the candidate tool list based on the prompt template.

7. The method according to claim 5, wherein arranging and combining the at least one tool to obtain the tool execution plan comprises:

arranging, by the server, tools in sequence and/or in parallel based on a dependency relationship between tool parameters of the tools; and

performing, by the server, operational connection on the tools arranged in sequence and/or in parallel through a predefined operator to obtain the tool execution plan.

8. The method according to any one of claims 5 to 7, wherein the tool execution plan is generated by a pre-trained large model, and a pre-training method for the large model comprises:

performing, by the server, continued pre-training on a large model in a general domain based on a sample dataset in a predefined domain;

performing, by the server, parameter fine-tuning on the large model after the continued pre-training based on a tool invocation case in the predefined domain; and

converting, by the server, a parameter weight of the fine-tuned large model from a floating-point number to an integer with a preset number of bits to obtain the pre-trained large model.

9. The method according to claim 1, wherein executing, by the server, the tool execution plan to obtain the plan execution result comprises:

acquiring, by the server, context information, and extracting, from the dialogue information and the context information, tool parameters required by tools in the tool execution plan;

invoking, by the server, the tools sequentially and/or in parallel based on the tool parameters in accordance with an order of the tools as arranged and combined in the tool execution plan to obtain the plan execution result; and

wherein invoking, by the server, the tools sequentially and/or in parallel based on the tool parameters in accordance with the order of the tools as arranged and combined in the tool execution plan to obtain the plan execution result comprises:

invoking, by the server, tools arranged in sequence in the tool execution plan sequentially, and invoking tools arranged in parallel in the tool execution plan in parallel based on the tool parameters to obtain the plan execution result.

- 5 10. The method according to claim 9, further comprising:
 in response to identifying that there is a missing tool parameter, postponing, by the
server, an execution order of a tool whose tool parameter is missing, and extracting the
missing tool parameter from an execution result of an already executed tool; and/or
 initiating, by the server, query information based on the missing tool parameter, and
10 extracting the missing tool parameter from user response information to the query
information; and/or
 extracting, by the server, a time parameter from the tool parameters, and modifying
relative time in the time parameter to absolute time; and/or
 extracting, by the server, an order parameter from the tool parameters, and modifying
15 a user expression in the order parameter to a standard expression in a predefined domain;
and/or
 the tools in the tool execution plan comprises at least one of a knowledge base search
tool, an information query tool, a function configuration tool, or an information parsing tool.
- 20 11. The method according to claim 1, wherein performing the check, by the server, on
the plan execution result comprises:
 acquiring, by the server, context information, and performing a check on each tool
in the tool execution plan and the plan execution result based on the dialogue information
and the context information;
25 in response to determining that no tool is missing from the tool execution plan and/or
the plan execution result is complete, determining, by the server, that the plan execution
result passes the check; and/or
 in response to determining that a tool is missing from the tool execution plan and/or
the plan execution result is incomplete, cyclically executing, by the server, following process
30 until a modified plan execution result passes the check, and/or a number of checks on the
plan execution result reaches a preset threshold: generating, by the server, feedback
information based on the tool execution plan and a check result of the plan execution result,

modifying the tool execution plan based on the feedback information and the intent information, and executing the modified tool execution plan to obtain the modified plan execution result;

5 wherein modifying, by the server, the tool execution plan based on the feedback information and the intent information comprises:

selecting, by the server, at least one tool based on the feedback information and the intent information, and adding the at least one tool to the tool execution plan to obtain the modified tool execution plan.

10 12. The method according to claim 1, wherein generating, by the server, the reply content based on the plan execution result that passes the check comprises:

acquiring, by the server, context information, and integrating content related to the dialogue information in the plan execution result based on the dialogue information and the context information to obtain the reply content;

15 wherein integrating, by the server, the content related to the dialogue information in the plan execution result based on the dialogue information and the context information to obtain the reply content comprises:

20 deleting, by the server, content unrelated to the dialogue information in the plan execution result, and integrating content remained in the plan execution result to obtain the reply content; and/or

acquiring, by the server, at least one piece of link information based on the dialogue information, and adding the at least one piece of link information to the reply content; and/or

25 in response to identifying that the dialogue information comprises a negative emotion, generating, by the server, reassurance information based on the dialogue information, and adding the reassurance information to the reply content; and/or

30 in response to identifying that the dialogue information comprises human service requirement information, generating, by the server, guidance information based on the dialogue information, and adding the guidance information into the reply content; and/or

in response to identifying that the dialogue information does not match a

predefined domain, generating, by the server, recognition rejection information based on the dialogue information, and using the recognition rejection information as the reply content; and/or

integrating, by the server, content related to the dialogue information in the plan execution result under a constraint condition of predefined word count and language style to generate structured reply content; and/or

retrieving, by the server, at least one case from a reply content experience repository through a predefined information retrieval model based on the intent information, and configuring the at least one case into a prompt template to determine the reply content based on the prompt template, wherein the information retrieval model comprises at least one of a sparse retrieval model, a dense vector retrieval model, or a multi-vector retrieval model.

13. The method according to claim 1 or 12, wherein the reply content is generated by a pre-trained large model, and a pre-training method for the large model comprises:

generating, by the server, a plurality of types of information reply cases based on a plurality of predefined dialogue scenarios in a predefined domain; and

performing, by the server, parameter fine-tuning on a predefined large model based on the plurality of types of information reply cases to obtain a pre-trained large model, wherein the predefined large model is a large model in a general domain or a large model obtained after continued pre-training with a sample dataset in a predefined domain.

14. The method according to claim 1, further comprising:

generating, by the server, at least one type of case, and storing the case in a corresponding experience repository according to a type of the case, wherein the case comprises at least one of an intent recognition case, a tool invocation case, or an information reply case;

wherein generating the at least one type of case comprises:

receiving intent information, a work execution plan, and reply content generated during interaction of the user with the large model, and performing a feedback operation on the intent information, the work execution plan, and the reply content based on first positive and negative feedback information from the user

regarding the intent information, the work execution plan, and the reply content to obtain the at least one type of case; and/or

receiving second positive and negative feedback information from the user regarding the reply content, and performing a feedback operation on intent information, a work execution plan, and reply content generated in a process of generating the reply content based on the second positive and negative feedback information to obtain the at least one type of case.

15. The method according to claim 14, wherein performing, by the server, the feedback operation on the intent information, the work execution plan, and the reply content based on the first positive and negative feedback information to obtain the at least one type of case comprises:

in response to receiving positive feedback information from the user regarding the intent information, the work execution plan, and the reply content, performing, by the server, a positive feedback operation on a case corresponding to the intent information, the work execution plan, and the reply content corresponding to the positive feedback information, wherein the positive feedback operation comprises increasing an experience score of the case, and/or, adding a new case; and/or

in response to receiving negative feedback information from the user regarding the intent information, the work execution plan, and the reply content, performing, by the server, error correction processing on the intent information, the work execution plan, and the reply content corresponding to the negative feedback information to generate error correction information, and configuring the error correction information into a prompt template to regenerate intent information, a work execution plan, and reply content based on the prompt template;

wherein regenerating, by the server, the intent information, the work execution plan, and the reply content based on the prompt template comprises:

in response to identifying that a case corresponding to intent information, a work execution plan, and reply content that are generated through cases is erroneous, replacing, by the server, the erroneous case when regenerating intent information, a work execution plan, and reply content, and performing a negative feedback operation on the erroneous case, wherein the negative feedback operation comprises

decreasing an experience score of the case and/or deleting the case.

16. The method according to claim 14, wherein performing, by the server, the feedback operation on the intent information, the work execution plan, and the reply content generated in the process of generating the reply content based on the second positive and negative feedback information to obtain the at least one type of case comprises:

in response to receiving positive feedback information from the user regarding the reply content, performing, by the server, a positive feedback operation on a case corresponding to intent information, a work execution plan, and reply content that are generated through cases, wherein the positive feedback operation comprises increasing an experience score of the case; and/or

in response to receiving positive feedback information from the user regarding the reply content, adding, by the server, intent information, a work execution plan, and the reply content that are not generated through cases as a new case; and/or

in response to receiving negative feedback information from the user regarding the reply content, performing, by the server, a negative feedback operation on a case corresponding to intent information, a work execution plan, and reply content that are generated through cases, wherein the negative feedback operation comprises decreasing an experience score of the case and/or deleting the case; and/or

in response to receiving negative feedback information from the user regarding the reply content, deleting, by the server, intent information, a work execution plan, and reply content that are not generated through cases.

17. A task-oriented dialogue method, comprising:

in response to receiving dialogue information initiated by a user, sending the dialogue information;

receiving and displaying reply content, wherein a process of generating the reply content comprises:

determining intent information of the user through a large model based on the dialogue information, generating a tool execution plan according to the intent information, executing the tool execution plan to obtain a plan execution result, performing a check on the plan execution result, generating reply content based on

the plan execution result that passes the check.

18. A task-oriented dialogue method, comprising:

in response to receiving dialogue information initiated by a user, determining intent

5 information of the user through a large model;

generating a tool execution plan according to the intent information;

executing the tool execution plan to obtain a plan execution result;

performing a check on the plan execution result; and

generating reply content based on the plan execution result that passes the check.

10

19. A storage medium having a computer program stored thereon, wherein the computer program, when executed by one or more processors, implements the method according to any one of claims 1 to 18.

15 20. A computer device, comprising a storage medium, one or more processors, and a computer program stored on the storage medium and executable on the one or more processors, wherein the one or more processors, when executing the computer program, implement the method according to any one of claims 1 to 18.