

TEXT GENERATION METHOD, DEVICE AND SYSTEM, COMPUTING DEVICE, READABLE STORAGE MEDIUM

[0001] The present application claims priority to Chinese Patent Application No. 202410625183.X, titled "TEXT GENERATION METHOD, DEVICE AND SYSTEM, COMPUTING DEVICE, READABLE STORAGE MEDIUM", filed on May 20, 2024 with the National Intellectual Property Administration, PRC, which is incorporated herein by reference in its entirety.

FIELD

[0002] Embodiments of the present disclosure relate to the field of computer technology, and in particular to a text generation method, and in more particular to a text generation device, a text generation system, a computing device, a computer readable storage medium, and a computer program product.

BACKGROUND

[0003] With the development of computer technology, many tasks in people's daily lives and works may be executed by computers.

[0004] For example, basic data required for writing may be analyzed by computers to generate texts that meet writing requirements. Generally, a text generation template is set for a corresponding text to be generated, which may include multiple pieces of configuration information, the configuration information may indicate a topic and required content of the text to be generated. The computer may analyze the basic data based on the text generation template to generate a text that conforms to the text generation template.

[0005] However, the current text generation efficiency still needs to be improved.

SUMMARY

[0006] In view of this, a text generation method is provided according to an embodiment of the present disclosure, to improve the text generation efficiency. A text generation device, a

text generation system, a computing device, a computer readable storage medium, and a computer program product are further provided according to one or more embodiments of the present disclosure.

[0007] In an aspect of the embodiments according to the present disclosure, a text generation method is provided, which includes:

acquiring target data to be analyzed, and generating, based on the target data, a text generation template for generating a text;

determining, based on pieces of configuration information in the text generation template, multiple tasks to be executed for the target data;

performing topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determining, from the multiple tasks, target tasks that support parallel execution; and

executing the target tasks in parallel for the target data, and combining execution results of the target tasks to generate a text that conforms to the text generation template.

[0008] In another aspect of the embodiments according to the present disclosure, a text generation device is provided, which includes: an acquisition module, a first determination module, a second determination module and a text generation module. The acquisition module is configured to acquire target data to be analyzed, and generate, based on the target data, a text generation template for generating a text. The first determination module is configured to determine, based on pieces of configuration information in the text generation template, multiple tasks to be executed for the target data. The second determination module is configured to perform topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determine, from the multiple tasks, target tasks that support parallel execution. The text generation module is configured to execute the target tasks in parallel for the target data, and combine execution results of the target tasks to generate a text that conforms to the text generation template.

[0009] In still another aspect of the embodiments according to the present disclosure, a text generation system is provided, which includes: a terminal and a server. The terminal is configured to send a text generation instruction for target data to the server and generate, based on the target data, a text generation template for generating a text. The server is

configured to perform the above method in response to the text generation instruction, and generate a text that conforms to the text generation template, and send the generated text to the terminal.

[0010] In still another aspect of the embodiments according to the present disclosure, a computing device is provided, which includes a memory and a processor. The memory is configured to store computer programs/instructions, the processor is configured to execute the computer programs/instructions, where the computer programs/instructions, when executed by the processor, perform the steps of the method described above.

[0011] In still another aspect of the embodiments according to the present disclosure, a computer-readable storage medium is provided, which includes computer programs/instructions stored thereon, where the computer programs/instructions, when executed by a processor, perform the steps of the method described above.

[0012] In still another aspect of the embodiments according to the present disclosure, a computer program product is provided, which includes computer programs/instructions, where the computer programs/instructions in the computer program product, when executed by a processor, perform the steps of the method described above.

[0013] In the embodiment of the present disclosure, target data to be analyzed and a text generation template for generating a text may be acquired. In a process of generating a text based on the target data to be analyzed according to the text generation template, multiple tasks to be executed for the target data may be determined based on pieces of configuration information in the text generation template, and the tasks to be executed may be topologically sorted based on a dependency relationship of the tasks to determine target tasks that support parallel execution and execute the target tasks in parallel. Then, execution results of the tasks may be combined to generate a text that conforms to the text generation template. In this way, the process of generating a template text (such as paradigm writing) can be performed flexibly, and the target tasks in the text generation process may be performed reasonably in parallel, the text generation efficiency can be improved.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] Figure 1 is a simple flowchart of a process of generating a text based on a template

according to the conventional art;

[0015] Figure 2 is a schematic diagram of a structure of a text generation system according to an embodiment of the present disclosure;

5 [0016] Figure 3 is a flowchart of a text generation method according to an embodiment of the present disclosure;

[0017] Figure 4 is a schematic diagram of a result obtained by topologically sorting multiple tasks according to an embodiment of the present disclosure;

[0018] Figure 5 is a schematic diagram of a result obtained by topologically sorting multiple subtasks in a target task according to an embodiment of the present disclosure;

10 [0019] Figure 6 is a schematic diagram of a structure of a text generation device according to an embodiment of the present disclosure; and

[0020] Figure 7 is a block diagram of a structure of a computing device according to an embodiment of the present disclosure.

15 **DETAILED DESCRIPTION OF EMBODIMENTS**

[0021] Many specific details are set forth in the following description to facilitate a full understanding of the present disclosure. However, the present disclosure may be implemented in many other ways different from those described herein, and those skilled in the art may make similar generalizations without violating the connotations of the present disclosure, and
20 thus the present disclosure is not limited by the specific embodiments described below.

[0022] Terms used in one or more embodiments of the present disclosure are used only for the purpose of describing a particular embodiment and are not intended to limit one or more embodiments of the present disclosure. The singular forms of "a", "the", and "said" used in one or more embodiments of the present disclosure and the claims are also intended to
25 encompass the plural form, unless the context clearly indicates otherwise. It should also be understood that the term "and/or" used in one or more embodiments of the present disclosure refers to and encompasses any or all possible combinations of one or more of the listed items in association. The term "at least one" used in one or more embodiments of the present disclosure refers to "one or more", and term "multiple" refers to "two or more". The term
30 "including" is non-exclusive description and should be understood as "including but not

limited to", which may also include other content on the basis of the already described content.

[0023] It should be understood that while the terms "first", "second", etc. may be used in one or more embodiments of the present disclosure to describe various information, such information should not be limited to these terms. These terms are used only to distinguish the same type of information from one another. For example, without departing from the scope of the one or more embodiments of the present disclosure, "a first..." may also be referred to as "a second...", and similarly, "a second..." may be referred to as "a first...". Depending on the context, the phrase "if" as used herein may be interpreted as "at the time of" or "when" or "in response to a determination".

[0024] In addition, the user information (including but not limited to user device information, user personal information, etc.) and data (including but not limited to data used for analysis, stored data, displayed data, etc.) involved in one or more embodiments of the present disclosure are authorized by the user or sufficiently authorized by the parties, and that the set, use, and processing of the relevant data are subject to relevant standards and requirements, and a corresponding operation portal is provided for the user to choose to authorize or reject.

[0025] In one or more embodiments of the present disclosure, a large model refers to a deep learning model with large-scale model parameters, typically including billions, tens of billions, hundreds of billions, trillions, or even tens of billions of model parameters. The large model, also known as a foundation model, is pre-trained using large-scale unlabeled corpora to produce a pre-trained model with over 100 million parameters. This type of model may adapt to a wide range of downstream tasks and has good generalization capability, such as a Large Language Model (LLM) and a multi-modal pre-training model.

[0026] In practice, the large model may be applied to different tasks by fine tuning a pre-trained model with a small number of samples. The large model may be widely used in fields of natural language processing (NLP), computer vision, and the like. Specifically, the large model may be applied to computer vision tasks such as visual question answering (VQA), image caption (IC), image generation, as well as natural language processing tasks such as text-based sentiment classification, text summarization generation, machine translation. The main application scenarios of the large model include digital assistants,

intelligent robots, search, online education, office software, e-commerce, intelligent design, etc. The models involved in the embodiments of the present disclosure include but are not limited to the aforementioned large models, and may be any type of machine learning model, such as convolutional neural network (CNN) models and Transformer models.

5 **[0027]** With the development of computer technology and information digitization, the types of data are increasing, and the requirements for data efficiency are also becoming higher. Many tasks that were originally performed manually may be executed by computers to reduce labor costs and improve the task execution efficiency. For example, in many fields, it is required to analyze existing data and write an analysis result of the data into a text according
10 to a set template, which may be called paradigm writing. In the financial field, financial institutions need to analyze their financial reports in order to obtain review texts on the financial reports. The types of content that need to be included in a review text may be indicated by a set template. For example, the template may indicate that the review text should include a financial institution for the review, authenticity and completeness of a
15 financial report, as well as a revenue trend of a financial institution.

[0028] At present, the application of natural language models is becoming increasingly widespread. The natural language model may be applied to a writing scenario to generate a text that meets a writing requirement. For example, the natural language model may include a large language model LLM. The large language model has demonstrated powerful capabilities
20 in natural language processing, but still faces certain challenges in practice, for example, the large language model has the illusion phenomena, slow knowledge updates, and low information transparency in the process of outputting replies. To address these issues, Retrieval-Augmented Generation (RAG) technology is commonly used for improvement. RAG refers to a process of retrieving relevant information from an external knowledge base
25 before using the LLM to answer questions, for the LLM to combine this information with data that needs to be analyzed to answer the questions. With the RAG, the accuracy of the answers outputted by the LLM can be significantly improved, the illusion phenomenon of the LLM is reduced, especially for knowledge intensive tasks. The LLM can display the reference source of the output answer via the RAG, making it facilitate for users to verify the accuracy of the
30 answer and increase trust in an output of the model. With the RAG, knowledge updates and the introduction of domain specific knowledge are also promoted, enabling the LLM to effectively combine its parameterized knowledge with non-parameterized external knowledge

bases, ensuring the reliability of the LLM.

[0029] Paradigm writing may also be implemented by using the LLM based on the RAG. For example, for the target data to be analyzed for paradigm writing, the LLM may be used to query external data sources to obtain relevant information of the target data, and the LLM answers the corresponding query or generates a text, ensuring that the answers outputted by the LLM are based on the retrieved information, which can significantly improve the accuracy and relevance of the writing content. In addition, the output of the LLM may be converted into a text output that meets the paradigm requirement based on a text structure and a generation rule indicated by a pre-defined template.

[0030] Figure 1 is a simple flowchart of a process of generating a text based on a template according to the conventional art. As shown in Figure 1, a text generation device may first acquire a template for generating a text, and obtain a natural language text generated based on configuration information in the template. If the text to be generated is a financial report review text, this template is for writing a financial report review. Each row of configuration information in the template may correspond to content that needs to be included in the generated text. The text generation device may sequentially rewrite query statements for each row of configuration information in the template, map pieces of configuration information into a series of query statements, which are also search conditions corresponding to the natural language model.

[0031] The text generation device may recall information from a material library based on the obtained query statement, to obtain multiple pieces of information associated with the query statement. For example, the information may be recalled by a vector recall method or a Best Matching 25 (BM25) algorithm. This material library may include a financial report to be reviewed, as well as historical data relevant to the financial report, such as data of annual reports and announcements belonging to the same company as the financial report. The text generation device may rank results of the recall, such as ranking according to the degree of matching with the query statement, to obtain multiple ranked materials.

[0032] After obtaining the template, the text generation device may further trigger a call instruction of the large language model to call the large language model. After obtaining multiple ranked materials, the large language model is used to generate a text based on the multiple materials, and a review text corresponding to each query statement is obtained. The

text generation device may further perform post-processing on each obtained review text to obtain a final review text. If the text generation device verifies the review text or determines whether the review text meets the requirements and regenerate a review text if the review text does not meet the requirements. Alternatively, the text generation device may further perform certain processing to eliminate potential issues with the large language model itself.

[0033] However, the above process of generating a text in the conventional art presents high linearity and rigidity, resulting in low text generation efficiency, that is, the efficiency of paradigm writing is low, and certain systematization is also lacking. A text generation method is provided according to the present disclosure, which uses a text generation process different from the conventional art, to improve the efficiency and quality of text generation. The present disclosure also relates to a text generation device, a text generation system, a computing device, a computer-readable storage medium, and a computer program product. The text generation device and the computing device each may be a server-side device or a terminal device with a strong computing power.

[0034] Figure 2 is a schematic diagram of the structure of a text generation system according to an embodiment of the present disclosure. As shown in Figure 2, the text generation system 10 may include a server 101 and a terminal 102, and the terminal 102 may establish a communication connection with the server 101. For example, the server 101 may be a cloud server or a server cluster, and the terminal 102 may be a smartphone, a desktop computer, a laptop, a tablet or a smart wearable device.

[0035] The text generation method according to the present disclosure may be applied to the server 101. The terminal 102 may interact with the server 101, for example, the terminal 102 may send data to the server 101 or obtain data from the server 101. For example, the terminal 102 may send target data to be analyzed and a text generation template for generating a text to the server 101. The server 101 may process the received target data, such as by performing the text generation method according to the embodiment of the present disclosure, to obtain a text that conforms to the text generation template, and feedback the generated text to the terminal 102. The terminal 102 may send a text generation request to the server 101, the request may carry the target data and the text generation template. The text generation template may exist in the form of a document, or the terminal 102 may be directly sent content in the template to the server 101 in the form of a dialogue.

[0036] In some embodiments, if the computing power of the terminal 102 is strong, the terminal 102 may not be connected to the server 101, and the terminal 102 may directly perform the text generation method according to the embodiment of the present disclosure.

[0037] Figure 3 is a flowchart of a text generation method according to an embodiment of the present disclosure. Description is made by taking the application of the method to a text generation device with the text generation device being a server as an example in the following. As shown in Figure 3, the method may include the following steps 302 to 308.

[0038] In step 302, target data to be analyzed is acquired, and a text generation template for generating a text is generated based on the target data.

[0039] In the embodiment of the present disclosure, the text generation device may analyze a certain target data to generate a required text based on the target data. The target data may be specified by the terminal, and a user may send the target data to be analyzed to the text generation device via the terminal. For example, the target data may be a financial report, and the text to be generated for this target data may be a review text for the financial report. The target data may also be data from other fields, such as relevant data during a project execution process of an enterprise, and the text to be generated may be a summary text for the project execution process.

[0040] The text generation device may generate a text for the target data according to a certain text generation template, which may be used to specify content included in the text to be generated. For example, the text generation template may include a topic of the text, an evaluation index, and a question to be answered. The text generation template may also be specified by the terminal, for example, the terminal may send the text generation template together with the target data. Preferably, the text generation template may also be stored in the text generation device, and the terminal may only indicate a type of the text to be generated. The text generation device determines a corresponding text generation template based on the type of the text.

[0041] Alternatively, the terminal may not directly send the target data and text generation template to the text generation device, but instead send indication information of the target data to the text generation device. The text generation device may acquire the target data and text generation template from other data storage device based on the indication information.

[0042] In step 304, multiple tasks to be executed for the target data are determined based on pieces of configuration information in the text generation template.

[0043] The different pieces of configuration information in the text generation template may correspond to different contents in the text to be generated. If each piece of configuration information may occupy one row of the text generation template, it may be the basic unit of the text generation template, corresponding to a basic content in the final generated text.

[0044] The text generation device may perform at least one task for each piece of configuration information to generate a text corresponding to the configuration information. Correspondingly, at least one task may be determined correspondingly based on each piece of configuration information, which may all be tasks to be executed for the target data.

[0045] In the embodiment of the present disclosure, the text generation device may also need to perform tasks irrelevant to the text generation template to generate a required text. The text generation device may also determine these tasks as tasks to be executed for the target data. For example, for various text generation processes, the text generation device may need to perform certain fixed tasks, such as a data formatting task and an index building task. The text generation device may also determine these tasks as tasks to be executed for the target data.

[0046] In step 306, based on a dependency relationship of the multiple tasks, topological sorting is performed on the multiple tasks to determine target tasks that support parallel execution from the multiple tasks.

[0047] For the tasks to be executed for the target data, there may be some tasks that have a dependency relationship there between, while others do not have a dependency relationship. The dependency relationship represents an order of execution. If the execution of a task A requires an execution result of a task B, the task A has a dependency relationship with the task B, and the task A depends on the task B.

[0048] After determining the tasks to be executed for the target data, the text generation device may further analyze the dependency relationship of the tasks, and further perform topological sorting on the tasks based on the dependency relationship, to generate a Directed Acyclic Graph (DAG) for the tasks. Based on a result of the topology sorting, the execution process of the tasks may be intuitively determined, and tasks that need to be executed in order

may be determined, as well as target tasks that do not have a specific order of execution and support parallel execution may also be determined.

[0049] For example, the task generation device may retrieve the target data only after formatting the target data, and the formatting task and the retrieval task have a sequential order. Formatting processes of different data do not have a sequential order of execution, and support parallel execution.

[0050] In step 308, the target tasks are executed in parallel for the target data, execution results of the target tasks are combined, to generate a text that conforms to the text generation template.

[0051] After performing topological sorting on the tasks and determining the execution order of the tasks, the text generation device may execute the tasks for the target data in that order. Tasks with a sequential order of execution are executed in that order, and target tasks that support parallel execution are executed in parallel to improve the task execution efficiency. For each task, the task generation device ensures that the task is executed only after a task it depends on has been completed.

[0052] Since the execution results of the target tasks executed in parallel are independent and ultimately require the generation of complete text, the text generation device may also combine the execution results of the target tasks according to the relationship between the target tasks and the relationship between the target tasks and the generation of the overall text, to generate a text that conforms to the text generation template.

[0053] In some embodiments, the step of obtaining a complete text during the text generation process relies solely on the multiple target tasks executed in parallel. By directly combining the execution results of the target tasks according to a set combination manner, the final generated text may be obtained. In other embodiments, after the multiple target tasks are executed in parallel, other tasks need to be executed. In this case, after combining the execution results of the multiple target tasks, further processing needs to be performed based on the execution results until the step of generating a complete text is performed to obtain a text that conforms to the text generation template. In the text generation process, there may be multiple groups of target tasks, each group of target tasks may include multiple target tasks, multiple target tasks in the same group may be executed in parallel, and target tasks in different groups may have a dependency relationship or be executed in parallel.

[0054] With the text generation method according to the embodiment of the present disclosure, the paradigm writing process is divided into multiple independent tasks. Based on the dependency relationship between the tasks, parallel processing is performed on the target tasks that support parallel processing, which can effectively optimize the processing procedure, improving the efficiency of paradigm writing.

[0055] In some embodiments, the executing the target tasks in parallel for the target data in the above step 308 includes: executing the target tasks in parallel by calling multiple coroutines using a coroutine pool for the target data. A coroutine runs on a thread, and when one coroutine is executed, this coroutine may choose to voluntarily give up and let another coroutine run on the current thread. For each target task, a coroutine may be called, and the thread running on the coroutine is specifically used to execute the target task. The use of coroutine may effectively control the parallel execution of multiple tasks. In the embodiments of the present disclosure, coroutine pool may also be used to restrict and reuse each coroutine, avoiding unrestricted development of coroutines and resource waste.

[0056] Coroutines do not increase the number of threads, and multiple coroutines are run in a time-sharing multiplexing manner on the basis of the threads. Moreover, the switching of coroutines is completed in a user mode, and the cost of switching is much lower than the cost of threads switching from the user mode to a kernel mode.

[0057] In some embodiments, a maximum number of concurrent tasks may be set. Correspondingly, the executing the target tasks in parallel for the target data in the step 308 may include: controlling, based on a set number of concurrent tasks, target tasks with a number less than or equal to the set number of concurrent tasks to be executed in parallel for the target data.

[0058] For example, if the number of target tasks that support parallel execution is greater than the set number m of concurrent tasks, the text generation device may first execute m target tasks in parallel, and then execute the remaining target tasks. Alternatively, the text generation device may also divide the target tasks into multiple groups of target tasks, with a small difference in the number of target tasks in each group and all less than or equal to m , and then execute the multiple groups of target tasks in sequence, with the target tasks in each group executed in parallel.

[0059] In the embodiments of the present disclosure, a natural language model (such as the

Large Language Model (LLM)) may be used to generate a text, which may correspondingly improve the efficiency of paradigm writing based on LLM. For example, in step 306, the target tasks that the natural language model may perform in parallel may be determined, and in step 308, the natural language model may be called to perform one or more tasks for the target data.

[0060] In summary, with the text generation method according to the embodiment of the present disclosure, target data to be analyzed and a text generation template for generating a text may be acquired. In a process of generating a text based on the target data to be analyzed according to the text generation template, multiple tasks to be executed for the target data may be determined based on pieces of configuration information in the text generation template, and the tasks to be executed may be topologically sorted based on a dependency relationship of the tasks to determine target tasks that support parallel execution and execute the target tasks in parallel. Then, execution results of the tasks may be combined to generate a text that conforms to the text generation template. In this way, the process of generating a template text (such as paradigm writing) can be performed flexibly, and the target tasks in the text generation process may be performed reasonably in parallel, the text generation efficiency can be improved.

[0061] In some embodiments, the determining, based on pieces of configuration information in the text generation template, multiple tasks to be executed for the target data in the above step 304 includes:

determining, based on the pieces of configuration information in the text generation template, at least one data query task corresponding to each of the pieces of configuration information, and a combination task for query results of the at least one data query task;

determining, for data sources in the target data, a data formatting task and at least one index building task corresponding to each of the data sources; and

obtaining multiple tasks to be executed for the target data based on the data query task, the combination task, the data formatting task and the index building task.

[0062] In the embodiment of the present disclosure, the tasks to be executed during the text generation process may include: a data formatting (Format Data) task and an index building (Build Index) task for the target data, as well as data query task for the target data and a

combination task for query results of the data query tasks. This combination task may belong to the post-processing process in text generation.

[0063] Each piece of configuration information in the text generation template may correspond to at least one data query task, which may be a task of querying required data from the target data and its relevant data (such as from the above material library). This data query task may be performed by a natural language model. The text generation device may combine execution results of various data query tasks to generate a required text, and accordingly determine the combination task that depends on the data query task.

[0064] The data in the target data to be analyzed may be quite complex, such as data from multiple sources, and there may be differences in formats of the data from different data sources. The data from different data sources described in the embodiment of the present disclosure may correspond to data collected from different devices, or data collected from different functional modules of the same device. The content, type, or form of data from different data sources may differ. In order to facilitate analysis (such as querying) of the target data, the text generation device may format the target data before executing the data query task, to convert the target data into a unified format. Correspondingly, the formatting process may correspond to a data formatting task. The text generation device may perform formatting processing for different data sources separately, and correspondingly, each data source may correspond to a data formatting task.

[0065] After formatting data from different data sources, the text generation device may further build an index for each data source to quickly determine the required data for analysis based on the indexes. Correspondingly, an index building task may be set for each data source. Preferably, only one index building task may be established for a data source. Alternatively, a data source may contain different types of data, and different index building tasks may be established for different types of data to directly determine the corresponding types of data based on the indexes.

[0066] The data formatting task, the index building task, the data query task and the combination task for the same data source have a sequential dependency relationship, so these tasks for the same data source do not support parallel execution. However, these tasks for different data sources may be executed in parallel. For example, the performing topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and

determining target tasks that support parallel execution from the multiple tasks in step 306 includes: performing topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determining the data formatting task, the index building task and the data query task for different data sources belong to the target tasks that support parallel execution.

[0067] Figure 4 is a schematic diagram of a result obtained by performing topology sorting on multiple tasks according to an embodiment of the present disclosure. As shown in Figure 4, the target data may include data from two data sources. For example, the data from one source may be financial data, while the data from the other source may be other descriptive information. The data formatting task may be performed for these two types of data to obtain financial data of a standard format and a pdf parsing file shown in Figure 4. The index building task may be performed separately for these two types of data. Figure 4 takes an example of executing one index building task for the financial data to obtain a financial data index, and executing two different index building tasks for the block parsing file to obtain a table data index and a text data index, respectively. Preferably, the text generation device may also obtain other indexes other than the target data, for example, the other indexes may be indexes of external data related to the target data.

[0068] The text generation device may call multiple data query instances to perform data query tasks based on the index of each data. Figure 4 is described by taking multiple data query instances corresponding to a financial data query task, a problem query task and a feature query task as an example. The data for different query tasks may intersect or overlap. As shown in Figure 4, the financial data query task may perform data query based on a financial data index and a table data index, the problem query task may perform data query based on a text data index and an external data index, and the feature query task may perform data query based on a table data index, a text data index and an external data index. After multiple data query tasks are performed, post-processing may be performed, such as combining the query results of the data query task.

[0069] For example, as shown in Figure 4, the data formatting tasks and the index building tasks for two types of data may be executed in parallel, and the data query tasks may be executed in parallel after the index building tasks are completed. The task of obtaining external data index may also be executed in parallel with the data formatting task and the

index building task.

5 [0070] In some embodiments, the performing topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determining target tasks that support parallel execution from the multiple tasks in the above step 306 includes: performing topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determining different index building tasks for the same data source belong to the target tasks that support parallel execution. For example, as shown in Figure 4, the two index building tasks for the block parsing file belong to the target tasks that support parallel execution. After formatting the data to obtain the block parsing file, the two index building tasks may be executed in parallel.

15 [0071] In the embodiments of the present disclosure, the tasks may support nesting with each other. For example, any task in the text generation process may include multiple subtasks, which may also be executed in parallel in a similar manner to the above tasks. Taking the target task as an example, the text generation device may perform topological sorting on the multiple subtasks included in the target task based on the dependency relationship there between, and determine target subtasks that support parallel execution from the multiple subtasks. Furthermore, the target subtasks may be executed in parallel, and execution results of the target subtasks may be combined to obtain an execution result of the target task.

20 [0072] For example, the target task is a data query task, which may be a task to be executed determined based on the configuration information in the text generated template. This data query task may also be used for other types of data query tasks, such as a direct set task that may directly query data that meets a certain feature condition. The target tasks may include a query statement rewriting subtask, a data recall subtask, an analysis subtask and a post-processing subtask that need to be executed sequentially.

25 [0073] The process of executing the target task for the target data in step 308 may include the following steps s2 to s8.

[0074] In step s2, for each piece of configuration information in the text generation template, the query statement rewriting subtask is executed to obtain prompt information of a natural language model.

30 [0075] Each piece of configuration information in the text generation template may indicate

content that needs to be included in the text to be generated. The text generation device may map the configuration information to prompt information required to generate a text by a natural language model. For example, this mapping may be realized by executing the query statement rewriting subtask.

5 **[0076]** In step s4, a recall subtask of relevant data of the prompt information is executed for the target data.

[0077] For any prompt information, the text generation device may recall the relevant data of the prompt information from the material library to obtain multiple materials. The relevant information may be determined based on a degree of matching with the prompt information.

10 Regarding the recall subtask, reference may be made to the relevant description of information recall in Figure 1, which will not be repeated here. For example, in a case that the target data includes multiple data sources, the recall subtasks of relevant data of the prompt information may be executed in parallel for different data sources.

[0078] Preferably, the text generation device may recall relevant data based on data indexes, and recall data from different storage locations for different data indexes. The recall subtasks may be executed in parallel for different data indexes.

[0079] In step S6, the analysis subtasks are performed in parallel for multiple pieces of relevant data by the natural language model, to obtain a correlation between each piece of relevant data and the prompt information.

20 **[0080]** After recalling multiple pieces of relevant data of the prompt information, the text generation device may analyze each piece of relevant data by the natural language model, to generate a text corresponding to the prompt information based on the relevant data. This process may refer to the description of Figure 1 regarding generating a review text. The text generation device may also determine the accuracy of the analysis result based on the analysis result of the relevant data (such as the generated text), to determine the correlation between the relevant data and the prompt information. This correlation may be used as a score for relevant data.

[0081] In step s8, based on various pieces of relevant data and a correlation between each piece of relevant data and the prompt information, the post-processing subtask is performed to obtain a result text corresponding to the prompt information, and the execution result of the

data query task is determined based on the result text.

[0082] The text generation device may perform the post-processing subtask, such as by combining analysis results based on various pieces of relevant data, or further processing or adjusting the analysis results to obtain the result text corresponding to the prompt information.

5 **[0083]** Figure 5 is a schematic diagram of a result obtained by topologically sorting multiple subtasks in a target task according to an embodiment of the present disclosure, and Figure 5 is described by taking the target task corresponding to the problem query task in Figure 4 as an example. As shown in Figure 5, the text generation device rewrites the configuration information to obtain corresponding prompt information. Afterwards, the recall
10 subtask may be executed based on both the text data index and the external data index, to recall prompt information from both the text data and the external data, such as recalling files 1 and 2 from the text data, and recalling files 3 and 4 from the external data. For each file, the Large Language Model (LLM) service may be used to perform the analysis subtask, to obtain an analysis result, and score each file based on the analysis result. Afterwards, post-processing
15 may be performed based on the analysis result of each file to obtain a result text generated by the target task.

[0084] For the target task shown in Figure 5, the processes of executing the recall subtask based on the text data index and the external data index may be executed in parallel, and the processes of calling the LLM service to analyze the recalled files may also be executed in
20 parallel.

[0085] In the embodiment of the present disclosure, a retry mechanism may be set for the text generation device, which allows for the re-execution of any task that fails. The above step s8 may include:

executing the post-processing subtask based on the multiple pieces of relevant data and
25 the correlation between each of the multiple pieces of relevant data and the prompt information;

if a correlation between a processing result of the post-processing subtask and the prompt information does not meet a correlation condition, re-executing the recall subtask, the analysis subtask and the post-processing subtask; and

30 if a correlation between a processing result of the post-processing subtask and the

prompt information meets a correlation condition, determining the processing result as the result text corresponding to the prompt information.

[0086] For the post-processing subtask, reference may be made to the relevant description of post-processing in Figure 1 described above. For example, during the execution of the post-processing subtask, the text generation device may combine the analysis results of relevant data with a correlation greater than a threshold to obtain a processing result of the post-processing subtask. For example, the text generation device may further adjust the analysis results of relevant data with a correlation greater than the threshold to obtain the processing result of the post-processing subtask.

[0087] The text generation device may also determine whether the processing result of the post-processing subtask meets a text generation requirement corresponding to the prompt information, for example, by determining whether the processing result of the post-processing subtask meets the text generation requirement by determining whether the correlation between the processing result and the prompt information meets the correlation condition (for example, whether greater than the correlation threshold). In a case that the correlation condition is not met, the recall subtask and the analysis subtask may be re-executed based on the prompt information. In a case that the correlation condition is met, the post-processing result may be determined as the result text corresponding to the prompt information, that is, determined as the execution result of the target task. As shown in Figure 5, in a case that the post-processing result is invalid, a retry may be performed to return to the process after executing the rewriting process.

[0088] Preferably, the retry mechanism may also be configured with retry times and a retry time interval. The text generation device may only re-execute a task when the number of retries for the task is less than a set threshold and the time interval from the last execution of the task is greater than a set retry time interval. For example, the recall subtask and the analysis subtask mentioned above are re-executed.

[0089] In the embodiment of the present disclosure, the text generation device may call the DAG engine to perform the steps 306 and 308 mentioned above. The DAG engine may be defined with a Runnable interface, a Task Base class, and a DAG class, which correspond to an executable entity.

[0090] The Runnable interface defines basic structures of all executable entities. The

Runnable interface, as an abstract base class, ensures that different types of executing entities follow a unified behavior mode. The basic task class inherits from the Runnable interface and may provide basic implementation and additional functions (such as retry mechanism) for task execution. The DAG class belongs to the core of the DAG engine, and extends the Runnable interface, making it not only an independent task execution unit, but also a collection of tasks. The DAG class supports nesting, allowing a DAG to be added as a subtask to a larger DAG. This nested function provides great flexibility and powerful organizational capability for building complex and multi-level task execution process.

[0091] For example, during the text generation process, the text generation device may first define subclass instances of the basic task class and determine a specific task logic implemented by each subclass instance. A DAG instance may be established, and subtasks to be executed are added to the DAG instance to achieve the DAG nesting. Afterwards, an asynchronous execution method of the DAG class may be called to perform topological sorting of tasks, to manage the execution order and concurrency of the subtasks in the DAG instance. Through the nesting capability of the DAG, a complex task structure may be constructed to optimize the organization and execution of the text generation process.

[0092] In the embodiment of the present disclosure, the text generation device may divide the entire paradigm writing process into several basic task instances and DAG instances. The basic task instance may include a data formatting task instance, an index creation task instances, and post-processing task instances. This DAG instance may include various data query task instances.

[0093] In some embodiments, corresponding properties and methods to be performed may be defined for the above interfaces and classes, and when calling a corresponding instance to execute a task, the execution should be performed based on the properties and methods. For example, the properties of the Runnable interface may include: an identifier (name, used to identify and track the corresponding instance), an execution result (output), a list of dependent tasks (dependencies), the number of other tasks unsolved that depend on this task (dependency_count), and a current status (status, including status such as not started, running, completed, and failed).

[0094] The method corresponding to the Runnable interface may include: initializing basic properties, which may be represented by a function *initialize ()*; adding dependent tasks,

which may be represented by a function *add_dependencies (tasks)*; checking a preparation status of the task, which may be represented by a function *is_dead()*; and defining an asynchronous execution logic, which may be represented by a function *async_run()*.

[0095] The properties of the basic task class may include: a task name, a task input parameter, a task output result, a list of dependent tasks, a list of other tasks that depend on this task, task retry times (*x_retry*) and a retry interval (retry interval), and a current task status. The methods corresponding to the basic task class may include: adding a set of dependent tasks (*add_dependencies*), checking if the tasks are ready to execute (*is_dead*), executing tasks asynchronously (including processing retry logic and updating dependent task status), and setting subclasses to implement specific task logic (*execute*).

[0096] The properties of a DAG class may include: the maximum number of concurrent tasks (*maxw_workers*), a list of all tasks in the DAG, and a property indicating whether the tasks have been topologically sorted. The methods corresponding to DAG class may include: adding a single task (*add_task*) or a list of tasks (*add_tasks*) to the DAG, and asynchronously executing all subtasks in the DAG.

[0097] In the embodiments of the present disclosure, a new text generation mode is adopted to optimize the generation chain of paradigm writing, which can fill the technical gap and make the text generation process more efficient and controllable. Preferably, the text generation device may also optimize the text generation process by using task scheduling algorithms or parallel processing frameworks other than DAG engine.

[0098] In summary, in the text generation method according to the embodiment of the present disclosure, target data to be analyzed and a text generation template for generating a text may be acquired. In a process of generating a text based on the target data to be analyzed according to the text generation template, multiple tasks to be executed for the target data may be determined based on pieces of configuration information in the text generation template, and the tasks to be executed may be topologically sorted based on a dependency relationship of the tasks to determine target tasks that support parallel execution and execute the target tasks in parallel. Then, execution results of the tasks may be combined to generate a text that conforms to the text generation template. In this way, the process of generating a template text (such as paradigm writing) can be performed flexibly, and the target tasks in the text generation process may be performed reasonably in parallel, the text generation efficiency can

be improved.

[0099] Corresponding to the above method embodiments, an embodiment of a text generation device is further provided according to the present disclosure, which may be a server. Figure 6 is a schematic diagram of a structure of a text generation device according to an embodiment of the present disclosure. As shown in Figure 6, the text generation device includes: an acquisition module 602, a first determination module 604, a second determination module 606 and a text generation module 608. The acquisition module 602 is configured to acquire target data to be analyzed, and generate, based on the target data, a text generation template for generating a text. The first determination module 604 is configured to determine, based on pieces of configuration information in the text generation template, multiple tasks to be executed for the target data. The second determination module 606 is configured to perform topological sorting on the multiple tasks based on a dependency relationship of the multiple tasks, and determine, from the multiple tasks, target tasks that support parallel execution. The text generation module 608 is configured to execute the target tasks in parallel for the target data, and combine execution results of the target tasks to generate a text that conforms to the text generation template.

[0100] Preferably, the first determination module 604 is further configured to:

determine, based on the pieces of configuration information in the text generation template, at least one data query task corresponding to each of the pieces of configuration information, and a combination task for query results of the at least one data query task;

determine, for data sources in the target data, a data formatting task and at least one index building task corresponding to each of the data sources; and

obtain multiple tasks to be executed for the target data based on the data query task, the combination task, the data formatting task and the index building task.

[0101] Preferably, the data formatting task, the index building task, the data query task and the combination task for the same data source have a sequential dependency relationship. The second determination module 606 is further configured to:

perform topological sorting on the multiple tasks based on the dependency relationship of the multiple tasks, and determine data formatting tasks, index building tasks and data query tasks for different data sources belong to the target tasks that support parallel execution.

[0102] Preferably, the second determination module 606 is further configured to:

perform topological sorting on the multiple tasks based on the dependency relationship of the multiple tasks, and determine different index building tasks for the same data source belong to the target tasks that support parallel execution.

5 **[0103]** Preferably, the target task includes a data query task, and the text generation module 608 is further configured to:

execute a query statement rewriting subtask for each of the pieces of configuration information in the text generation template to obtain prompt information of a natural language model;

10 execute a recall subtask of relevant data of the prompt information for the target data;

execute analysis subtasks in parallel for multiple pieces of relevant data by using the natural language model, to obtain a correlation between each of the multiple pieces of relevant data and the prompt information; and

15 execute a post-processing subtask based on the multiple pieces of relevant data and the correlation between each of the multiple pieces of relevant data and the prompt information, to obtain a result text corresponding to the prompt information, and determine an execution result of the data query task based on the result text.

[0104] Preferably, the text generation module 608 is further configured to:

20 execute, in a case that the target data includes multiple data sources, the recall subtask of the relevant data of the prompt information in parallel for different data sources.

[0105] Preferably, the text generation module 608 is further configured to:

execute the post-processing subtask based on the multiple pieces of relevant data and the correlation between each of the multiple pieces of relevant data and the prompt information;

25 if a correlation between a processing result of the post-processing subtask and the prompt information does not meet a correlation condition, re-execute the recall subtask, the analysis subtask and the post-processing subtask; and

if a correlation between a processing result of the post-processing subtask and the prompt information meets a correlation condition, determine the processing result as the result text corresponding to the prompt information.

[0106] Preferably, the text generation module 608 is further configured to:

control, based on a set number of concurrent tasks, target tasks with a number less than or equal to the set number of concurrent tasks to be executed in parallel for the target data.

[0107] Preferably, the text generation module 608 is further configured to:

5 execute the target tasks in parallel for the target data by calling multiple coroutines using a coroutine pool.

[0108] In summary, in the text generation device according to the embodiment of the present disclosure, the acquisition module may acquire target data to be analyzed and a text generation template for generating a text. In a process of generating a text based on the target data to be analyzed according to the text generation template, the first determination module may determine multiple tasks to be executed for the target data based on pieces of configuration information in the text generation template, and the second determination module may perform topological sorting on the tasks to be executed based on a dependency relationship of the tasks to determine target tasks that support parallel execution and execute the target tasks in parallel. Then, the text generation module may combine execution results of the tasks to generate a text that conforms to the text generation template. In this way, the process of generating a template text (such as paradigm writing) can be performed flexibly, and the target tasks in the text generation process may be performed reasonably in parallel, the text generation efficiency can be improved.

20 **[0109]** The various embodiments in the present disclosure are described in a progressive manner, and the same and similar parts between the embodiments may be referred to each other. Each embodiment focuses on the differences from other embodiments. Especially for text generation device, which is similar to the embodiments of text generation method, description is relatively simple. For relevant parts, reference may be made to the description of the corresponding parts of text generation method embodiment.

[0110] Figure 7 is a block diagram of a structure of a computing device according to an embodiment of the present disclosure. The components of the computing device 700 include, but are not limited to, a memory 710 and a processor 720. The processor 720 is connected to the memory 710 via a bus 730, and a database 750 is used to store data.

30 **[0111]** The computing device 700 further includes an access device 740, the access device

740 enables the computing device 700 to communicate via one or more networks 760. Examples of these networks include a public switched telephone network (PSTN), a local area network (LAN), a wide area network (WAN), a personal area network (PAN), or a combination of communication networks such as the Internet. The access device 740 may include one or more of any types of wired or wireless network interfaces (e.g., a network interface controller (NIC), such as an IEEE802.11 Wireless Local Area Network (WLAN) wireless interface, Worldwide Interoperability for Microwave Access (Wi-MAX) interface, Ethernet interface, Universal Serial Bus (USB) interface, cellular network interface, Bluetooth interface, Near Field Communication (NFC).

[0112] In an embodiment of the present disclosure, the above components of the computing device 700, as well as other components not shown in Figure 7, may also be connected to each other, such as via a bus. It should be understood that the block diagram of the structure of the computing device shown in Figure 7 is for exemplary purposes only and is not a limitation on the scope of the present disclosure. Those skilled in the art may add or replace other components as desired.

[0113] The computing device 700 may be any type of stationary or mobile computing device, including a mobile computer or a mobile computing device (e.g., a tablet computer, a personal digital assistant, a laptop computer, a notebook computer, a netbook, etc.), a cell phone (e.g., a smartphone), a wearable computing device (e.g., a smartwatch, smartglasses, etc.), or other types of mobile devices, or a stationary computing device such as desktop computer or a personal computer (PC). The computing device 700 may also be a mobile or stationary server.

[0114] The processor 720 is used to execute computer programs/instructions, which, when executed by the processor, perform the steps of the above-described text generation method.

[0115] For the embodiment of a computing device, which is similar to the embodiment of text generation method, the description is relatively simple. For relevant parts, reference may be made to the description of the corresponding parts of text generation method embodiment.

[0116] A computer-readable storage medium is further provided according to an embodiment of the present disclosure, which stores computer instructions, the computer instructions, when executed by a processor, perform the steps of the above-described text generation method. The computer instruction includes computer program codes, the computer

program code may be in the form of source code, in the form of object code, in the form of an executable file or in some intermediate form or the like. The computer readable storage medium may include: any entity or device capable of carrying the computer program code, a recording medium, a USB disk, a removable hard disk, a diskette, an optical disk, a computer memory, a Read-Only Memory (ROM), a Random Access Memory (RAM), an electric carrier signal, a telecommunications signal, and software distribution medium, etc. It should be noted that the content in the computer-readable storage medium may be appropriately added or removed according to the requirements of legislation and patent practice within the jurisdiction. For example, in some jurisdictions, according to legislation and patent practice, the computer-readable medium does not include carrier signals and telecommunication signals.

[0117] A computer program product is further provided according to an embodiment of the present disclosure, which includes computer programs/instructions, the computer programs/instructions, when executed by a processor, perform the steps of the above-described text generation method.

[0118] For the embodiments of the computer-readable storage medium and the computer program product, which are similar to the embodiment of text generation method, the description is relatively simple. For relevant parts, reference may be made to the description of the corresponding parts of text generation method embodiment.

[0119] Preferred embodiments of the present disclosure are described above. Other embodiments are within the scope of the claims. In some cases, the actions or steps described in the claims may be performed in a different order than in the embodiments and still achieve the desired results. In addition, the processes depicted in the drawings do not necessarily require the particular order or consecutive order shown to achieve the desired results. In some embodiments, multi-tasking and parallel processing is also possible or may be advantageous.

[0120] It should be noted that, preferred embodiments of the present disclosure are described above. Other embodiments are within the scope of the claims. In some cases, the actions or steps described in the claims may be performed in a different order than in the embodiments and still achieve the desired results. In addition, the processes depicted in the drawings do not necessarily require the particular order or consecutive order shown to achieve the desired results. In some embodiments, multi-tasking and parallel processing is also

possible or may be advantageous. Secondly, those skilled in the art should also be aware that the embodiments described in the present disclosure are all preferred embodiments, and the actions and modules involved may not necessarily be necessary for the embodiments of the present disclosure.

- 5 **[0121]** In the above-described embodiments, the description of each embodiment has its own focus, and the part not described in detail in a certain embodiment may be referred to the relevant description of other embodiments.

- 10 **[0122]** The above preferred embodiments of the present disclosure are intended only to aid in the description of the present disclosure. The preferred embodiments are not an exhaustive recitation of all details, nor do they limit the invention to the specific embodiments described only. Apparently, many modifications and variations may be made in accordance with the contents of the embodiments of the present disclosure. These embodiments are selected and specifically described in the present disclosure to better explain the principles and practice of the embodiments of the present disclosure so that those skilled in the art to which they belong
15 will be able to well understand and utilize the present disclosure.

CLAIMS

1. A text generation method, comprising:

acquiring target data to be analyzed, and generating, based on the target data, a text
5 generation template for generating a text;

determining, based on pieces of configuration information in the text generation template,
a plurality of tasks to be executed for the target data;

performing topological sorting on the plurality of tasks based on a dependency
relationship of the plurality of tasks, and determining, from the plurality of tasks, target tasks
10 that support parallel execution; and

executing the target tasks in parallel for the target data, and combining execution results
of the target tasks to generate a text that conforms to the text generation template.

2. The method according to claim 1, wherein the determining, based on pieces of
15 configuration information in the text generation template, a plurality of tasks to be executed
for the target data comprises:

determining, based on the pieces of configuration information in the text generation
template, at least one data query task corresponding to each of the pieces of configuration
information, and a combination task for query results of the at least one data query task;

20 determining, for data sources in the target data, a data formatting task and at least one
index building task corresponding to each of the data sources; and

obtaining a plurality of tasks to be executed for the target data based on the data query
task, the combination task, the data formatting task and the index building task.

25 3. The method according to claim 2, wherein the data formatting task, the index building
task, the data query task and the combination task for the same data source have a sequential
dependency relationship;

the performing topological sorting on the plurality of tasks based on a dependency
relationship of the plurality of tasks, and determining, from the plurality of tasks, target tasks

that support parallel execution comprises:

performing topological sorting on the plurality of tasks based on the dependency relationship of the plurality of tasks, and determining data formatting tasks, index building tasks and data query tasks for different data sources belong to the target tasks that support parallel execution.

4. The method according to claim 2, wherein the performing topological sorting on the plurality of tasks based on a dependency relationship of the plurality of tasks, and determining, from the plurality of tasks, target tasks that support parallel execution comprises:

performing topological sorting on the plurality of tasks based on the dependency relationship of the plurality of tasks, and determining different index building tasks for the same data source belong to the target tasks that support parallel execution.

5. The method according to any one of claims 1 to 4, wherein the target tasks comprise a data query task, and the executing the target tasks in parallel for the target data comprises:

executing a query statement rewriting subtask for each of the pieces of configuration information in the text generation template to obtain prompt information of a natural language model;

executing a recall subtask of relevant data of the prompt information for the target data;

executing analysis subtasks in parallel for a plurality of pieces of relevant data by using the natural language model, to obtain a correlation between each of the plurality of pieces of relevant data and the prompt information; and

executing a post-processing subtask based on the plurality of pieces of relevant data and the correlation between each of the plurality of pieces of relevant data and the prompt information, to obtain a result text corresponding to the prompt information, and determining an execution result of the data query task based on the result text.

6. The method according to claim 5, wherein the executing a recall subtask of relevant data of the prompt information for the target data comprises:

executing, in a case that the target data comprises a plurality of data sources, the recall subtask of the relevant data of the prompt information in parallel for different data sources.

7. The method according to claim 5, wherein the executing a post-processing subtask based on the plurality of pieces of relevant data and the correlation between each of the plurality of pieces of relevant data and the prompt information, to obtain a result text corresponding to the prompt information comprises:

executing the post-processing subtask based on the plurality of pieces of relevant data and the correlation between each of the plurality of pieces of relevant data and the prompt information;

if a correlation between a processing result of the post-processing subtask and the prompt information does not meet a correlation condition, re-executing the recall subtask, the analysis subtask and the post-processing subtask; and

if a correlation between a processing result of the post-processing subtask and the prompt information meets a correlation condition, determining the processing result as the result text corresponding to the prompt information.

8. The method according to any one of claims 1 to 4, wherein the executing the target tasks in parallel for the target data comprises:

controlling, based on a set number of concurrent tasks, target tasks with a number less than or equal to the set number of concurrent tasks to be executed in parallel for the target data.

9. The method according to any one of claims 1 to 4, wherein the executing the target tasks in parallel for the target data comprises:

executing the target tasks in parallel for the target data by calling a plurality of coroutines using a coroutine pool.

10. A text generation device, comprising:

an acquisition module, configured to acquire target data to be analyzed, and generate, based on the target data, a text generation template for generating a text;

a first determination module, configured to determine, based on pieces of configuration information in the text generation template, a plurality of tasks to be executed for the target data;

a second determination module, configured to perform topological sorting on the plurality of tasks based on a dependency relationship of the plurality of tasks, and determine, from the plurality of tasks, target tasks that support parallel execution; and

a text generation module, configured to execute the target tasks in parallel for the target data, and combine execution results of the target tasks to generate a text that conforms to the text generation template.

11. A text generation system, comprising a terminal and a server, wherein

the terminal is configured to send a text generation instruction for target data to the server and generate, based on the target data, a text generation template for generating a text; and

the server is configured to perform the method according to any one of claims 1 to 9 in response to the text generation instruction, and generate a text that conforms to the text generation template, and send the generated text to the terminal.

12. A computing device, comprising:

a memory, configured to store computer programs/instructions; and

a processor, configured to execute the computer programs/instructions, wherein the programs/instructions, when executed by the processor, perform the method according to any one of claims 1 to 9.

13. A computer-readable storage medium, comprising computer programs/instructions stored thereon, wherein the computer programs/instructions, when executed by a processor, perform the method according to any one of claims 1 to 9.

14. A computer program product, comprising computer programs/instructions, wherein the computer programs/instructions, when executed by a processor, perform the method according to any one of claims 1 to 9.