MODEL INFERENCE METHOD AND SYSTEM, AND COMPUTE DEVICE

TECHNICAL FIELD

[0001] This application relates to the field of artificial intelligence, and more specifically, to a model inference method and system, and a compute device.

BACKGROUND

[0002] In recent years, with the rapid development of large models, smart applications built based on large models have been widely implemented, influencing the ways in which people produce and use information. A representative feature of large models is their huge quantity of parameters. To continuously improve an application effect of large models, current large models are still developing rapidly along the direction of increasing the quantity of parameters. On one hand, the huge quantity of parameters makes large models successful; on the other hand, it also becomes a bottleneck hindering the widespread adoption of large models. One of the important reasons is high-cost service-based deployment of large models in an inference phase. Due to a huge quantity of parameters, large models require substantial compute resources during inference. However, due to the limitation of the auto-regressive token-by-token decoding generation method, utilization of the compute resources during large model inference is relatively low. How to enable models to make full use of compute resources during inference, increase a throughput rate of model inference, and reduce inference costs of large models is a difficult and important problem.

[0003] In solutions related to generative model inference, in two key phases of draft generation and draft verification, a draft generation model and a draft verification model have a strong coupling relationship in terms of a collaboration manner and a correspondence manner. The strong coupling relationship is fixed when a program is started. This results in poor flexibility of the draft generation model and the draft verification model in terms of the collaboration manner and the correspondence manner, thereby reducing model inference efficiency.

[0004] Therefore, how to improve the model inference efficiency becomes an urgent technical problem to be resolved.

SUMMARY

[0005] This application provides a model inference method and system, and a compute device. This can improve model inference efficiency.

[0006] According to a first aspect, a model inference method is provided. The method includes: determining at least one draft verification model from a draft verification model pool, where the draft verification model pool includes a plurality of draft verification models; determining, from a draft generation model pool, at least one draft generation model corresponding to the at least one draft verification model, where the draft generation model pool includes a plurality of draft generation models; receiving text prompt information input by a first user, where the text prompt information is used to generate text information; obtaining, from a draft pool, a target draft related to the text prompt information, where the draft pool stores a plurality of drafts, and the plurality of drafts include a draft generated by the at least one draft generation model; verifying the target draft by using the at least one draft verification model, and obtaining a verification result; and generating the text information based on the verification result, and feeding back the text information to the first user.

[0007] The text prompt information may be, for example, a prompt input by a user. The prompt generally refers to a piece of text or information that serves as an initial input or guidance for a generation model to guide the generation model to generate a draft.

[0008] The determined at least one draft verification model may be a draft verification model that is built in the draft verification model pool by a vendor, or may be a draft verification model obtained by the user fine-tuning, based on data of the user, the draft verification model that is built in the draft verification model pool by the vendor, or may be a draft verification model of the user (the user connects the draft verification model of the user to the draft verification model pool through a connection interface).

[0009] The determined at least one draft generation model may be a draft generation model that is built in the draft generation model pool by the vendor, or may be a draft generation model obtained by the user fine-tuning, based on data of the user, the draft generation model that is built in the draft generation model pool by the vendor, or may be a draft generation model of the user (the user connects the draft generation model of the user to the draft generation model pool through a connection interface).

[0010] In the foregoing technical solution, the draft generation model and the draft verification model may be fully decoupled in terms of a correspondence manner and a collaboration manner. The draft generation model and the draft verification model may be dynamically combined in any proportion. In addition, the correspondence manner and the collaboration manner between the draft generation model and the draft verification model are not fixed and are more flexible, so that model inference efficiency can be improved.

[0011] With reference to the first aspect, in some implementations of the first aspect, correspondence modes between the at least one draft verification model and the at least one draft

generation model include: one-to-one, one-to-many, many-to-one, and many-to-many.

[0012] In the foregoing technical solution, there are a plurality of correspondences, for example, one-to-one, one-to-many, many-to-one, and many-to-many, between the at least one draft verification model and the at least one draft generation model, and the correspondences may be switched in real time and dynamically combined.

[0013] With reference to the first aspect, in some implementations of the first aspect, the at least one draft generation model includes a draft synchronous generation model, and the method further includes: obtaining draft generation request information based on the text prompt information; sending the draft generation request information to the draft synchronous generation model, where the draft synchronous generation model is configured to generate a draft based on the draft generation request information; and storing, in the draft pool, the draft generated by the draft synchronous generation model.

[0014] In the foregoing technical solution, the draft pool may record and manage a draft generated by a draft generation model in a running process of a system, so that the draft recorded in the draft pool can be used for subsequent inference, thereby enabling the draft to be reused and avoiding a situation that the generated draft is used only for inference verification of a current request. This can reduce computational power and time required for draft generation to some extent, thereby further improving model inference efficiency and reducing model inference costs.

[0015] With reference to the first aspect, in some implementations of the first aspect, the draft pool further stores attribute information of the drafts.

[0016] In the foregoing technical solution, the attribute information of the drafts may be further recorded in the draft pool. In this way, an effective draft screening mechanism is provided for subsequent inference, thereby improving model inference efficiency and reducing model inference costs.

[0017] With reference to the first aspect, in some implementations of the first aspect, the target draft related to the text prompt information is obtained from the draft pool based on the attribute information of the drafts stored in the draft pool.

[0018] In the foregoing technical solution, more effective drafts may be screened and retrieved for different text prompt information (for example, prompts) based on the attribute information of the drafts stored in the draft pool.

[0019] With reference to the first aspect, in some implementations of the first aspect, the method further includes: retrieving and/or updating the drafts stored in the draft pool based on the attribute information of the drafts stored in the draft pool.

[0020] In the foregoing technical solution, self-maintenance and automatic update may be performed on the draft pool based on the attribute information of the drafts stored in the draft pool.

[0021] With reference to the first aspect, in some implementations of the first aspect, the method further includes: performing operations on the draft generation models included in the draft generation model pool based on the attribute information of the drafts stored in the draft pool, where the operations include: activation, hibernation, and release.

[0022] In the foregoing technical solution, operations such as activation, hibernation, and release may be performed on the draft generation models included in the draft generation model pool based on the attribute information of the drafts stored in the draft pool, thereby improving model inference efficiency.

[0023] With reference to the first aspect, in some implementations of the first aspect, the attribute information of the draft includes at least one of the following: a field of the draft, content of the draft, a label of the draft generation model corresponding to the draft, a quantity of times that the draft is invoked, a quantity of times that the draft is hit, and a list of hit text prompt information.

[0024] In the foregoing technical solution, multi-dimensional attribute information of the draft may be used, so that draft recall is more accurate, and a higher hit rate is achieved in a draft verification phase.

[0025] With reference to the first aspect, in some implementations of the first aspect, the draft generation request information further includes a quantity of to-be-generated drafts, the quantity of to-be-generated drafts is determined based on usage of a model inference resource, and the model inference resource is used by a draft generation model in the draft generation model pool to generate a draft and/or by a draft verification model in the draft verification model pool to perform a verification process.

[0026] In the foregoing technical solution, a quantity of drafts may be adaptively adjusted based on a current inference resource status, a resource occupation status, and the like.

[0027] With reference to the first aspect, in some implementations of the first aspect, the at least one draft generation model includes a draft asynchronous generation model, and the method further includes: when inference load is less than a preset threshold, activating the draft asynchronous generation model to generate a draft; and storing, in the draft pool, the draft generated by the draft asynchronous generation model.

[0028] In the foregoing technical solution, when the inference load is less than the preset threshold, the draft asynchronous generation model may be activated to generate the draft, so as to avoid a situation that the draft generation model and the draft verification model wait for each other in a synchronous mode, thereby improving utilization of inference resources in a model inference process, improving model inference efficiency, and reducing model inference costs.

[0029] With reference to the first aspect, in some implementations of the first aspect, the

method further includes: automatically requesting/releasing the model inference resource based on usage of the model inference resource.

[0030] In the foregoing technical solution, an inference compute resource may be applied for/released in real time based on a current inference system status.

[0031] With reference to the first aspect, in some implementations of the first aspect, the draft pool includes a shared draft pool and a private draft pool of the first user, and the target draft related to the text prompt information is obtained from the shared draft pool and/or the private draft pool of the first user.

[0032] In the foregoing technical solution, the draft pool supports a multi-tenant mode. The draft pool mainly includes two parts: a shared draft pool and a private draft pool of each tenant , where the shared draft pool may be shared by multiple tenants, and draft pools of the tenants are isolated from each other. When configuring respective inference services, users may choose to use the shared draft pool for cold start.

[0033] With reference to the first aspect, in some implementations of the first aspect, the draft generated by the draft synchronous generation model is stored in the private draft pool of the first user.

[0034] In the foregoing technical solution, after the inference service is started, the system allocates an independent draft pool for each user, and the draft generated in a running process of the inference service of each user is written to the private draft pool of the user. The drafts in a running process of the inference services of different users can be maintained independently.

[0035] With reference to the first aspect, in some implementations of the first aspect, the draft generated by the draft asynchronous generation model is stored in the private draft pool of the first user.

[0036] According to a second aspect, a model inference system is provided. The system includes a draft verification management unit, a draft generation management unit, and at least one draft verification model. The draft verification management unit is configured to determine at least one draft verification model from a draft verification model pool, where the draft verification model pool includes a plurality of draft verification models; the draft generation management unit is configured to determine, from a draft generation model pool, at least one draft generation model corresponding to the at least one draft verification model, where the draft generation model pool includes a plurality of draft generation models; the at least one draft verification model is configured to receive text prompt information input by a first user, where the text prompt information is used to generate text information; the draft verification management unit is further configured to obtain, from a draft pool, a target draft related to the text prompt information, where the draft pool stores a plurality of drafts, and the plurality of drafts include a draft generated by the

at least one draft generation model; the at least one draft verification model is further configured to verify the target draft, and obtain a verification result; and the at least one draft verification model is further configured to generate the text information based on the verification result, and feed back the text information to the first user.

[0037] With reference to the second aspect, in some implementations of the second aspect, the at least one draft generation model includes a draft synchronous generation model, and the at least one draft verification model is further configured to: obtain draft generation request information based on the text prompt information, and send the draft generation request information to the draft generation management unit through the draft verification management unit; the draft generation management unit is further configured to send the draft generation request information to the draft synchronous generation model; the draft synchronous generation model is configured to generate a draft based on the draft generation request information, and send the draft to the draft generation management unit; and the draft generation management unit is further configured to store, in the draft pool, the draft generated by the draft synchronous generation model.

[0038] With reference to the second aspect, in some implementations of the second aspect, the at least one draft generation model includes a draft asynchronous generation model, and the draft verification management unit is further configured to: when inference load is less than a preset threshold, activate the draft asynchronous generation model to generate a draft; the draft asynchronous generation model is configured to generate a draft, and send the draft to the draft verification management unit; and the draft verification management unit is further configured to store, in the draft pool, the draft generated by the draft asynchronous generation model.

[0039] With reference to the second aspect, in some implementations of the second aspect, the draft pool further stores attribute information of the drafts.

[0040] With reference to the second aspect, in some implementations of the second aspect, the draft verification management unit is specifically configured to: obtain, from the draft pool, the target draft related to the text prompt information based on the attribute information of the drafts stored in the draft pool.

[0041] With reference to the second aspect, in some implementations of the second aspect, the system further includes: a draft analysis unit, configured to retrieve and/or update the drafts stored in the draft pool based on the attribute information of the drafts stored in the draft pool.

[0042] With reference to the second aspect, in some implementations of the second aspect, the draft generation management unit is further configured to perform operations on the draft generation models included in the draft generation model pool based on the attribute information of the drafts stored in the draft pool, where the operations include: activation, hibernation, and release.

[0043] With reference to the second aspect, in some implementations of the second aspect, the attribute information of the draft includes at least one of the following: a field of the draft, content of the draft, a label of the draft generation model corresponding to the draft, a quantity of times that the draft is invoked, a quantity of times that the draft is hit, and a list of hit text prompt information.

[0044] With reference to the second aspect, in some implementations of the second aspect, the draft generation request information further includes a quantity of to-be-generated drafts, the quantity of to-be-generated drafts is determined based on usage of a model inference resource, and the model inference resource is used by a draft generation model in the draft generation model pool to generate a draft and/or by a draft verification model in the draft verification model pool to perform a verification process.

[0045] With reference to the second aspect, in some implementations of the second aspect, the system further includes: an inference resource analysis unit, configured to automatically request/release the model inference resource based on usage of the model inference resource.

[0046] With reference to the second aspect, in some implementations of the second aspect, the draft pool includes a shared draft pool and a private draft pool of the first user, and the draft verification management unit is specifically configured to: obtain, from the shared draft pool and/or the private draft pool of the first user, the target draft related to the text prompt information.

[0047] With reference to the second aspect, in some implementations of the second aspect, the draft generation management unit is specifically configured to store, in the private draft pool of the first user, the draft generated by the draft synchronous generation model.

[0048] With reference to the second aspect, in some implementations of the second aspect, the draft generation management unit is specifically configured to store, in the private draft pool of the first user, the draft generated by the draft asynchronous generation model.

[0049] With reference to the second aspect, in some implementations of the second aspect, correspondence modes between the at least one draft verification model and the at least one draft generation model include: one-to-one, one-to-many, many-to-one, and many-to-many.

[0050] It should be understood that beneficial effects of the second aspect and the implementations of the second aspect are similar to those of the first aspect and the implementations of the first aspect. For details, refer to the beneficial effects of the first aspect and the implementations of the first aspect. Details are not described herein again.

[0051] According to a third aspect, a compute device is provided, including a processor and a memory, and optionally, further including an input/output interface. The processor is configured to control the input/output interface to send and receive information, the memory is configured to store a computer program, and the processor is configured to invoke the computer program from

the memory and run the computer program, to cause the compute device to perform the method according to any one of the first aspect or the possible implementations of the first aspect.

[0052] Optionally, the processor may be a general-purpose processor, and may be implemented by hardware or software. When the processor is implemented by the hardware, the processor may be a logic circuit, an integrated circuit, or the like. When the processor is implemented by the software, the processor may be a general-purpose processor, and is implemented by reading software code stored in the memory. The memory may be integrated into the processor, or may be located outside the processor and exist independently.

[0053] According to a fourth aspect, a compute device cluster is provided, including at least one compute device. Each compute device includes a processor and a memory. The processor of the at least one compute device is configured to execute instructions stored in the memory of the at least one compute device, to enable the compute device cluster to perform the method according to any one of the first aspect or the possible implementations of the first aspect.

[0054] According to a fifth aspect, a chip is provided. The chip obtains instructions and executes the instructions to implement the method according to any one of the first aspect or the implementations of the first aspect.

[0055] Optionally, in an implementation, the chip includes a processor and a data interface. The processor reads, through the data interface, instructions stored in a memory, to perform the method according to any one of the first aspect and the implementations of the first aspect.

[0056] Optionally, in an implementation, the chip may further include the memory. The memory stores the instructions. The processor is configured to execute the instructions stored in the memory. When the instructions are executed, the processor is configured to perform the method according to any one of the first aspect and the implementations of the first aspect.

[0057] According to a sixth aspect, a computer program product including instructions is provided. When the instructions are run by a compute device, the compute device is enabled to perform the method according to any one of the first aspect or the implementations of the first aspect.

[0058] According to a seventh aspect, a computer program product including instructions is provided. When the instructions are run by a compute device cluster, the compute device cluster is enabled to perform the method according to any one of the first aspect or the implementations of the first aspect.

[0059] According to an eighth aspect, a computer-readable storage medium is provided, including computer program instructions. When the computer program instructions are executed by a compute device, the compute device performs the method according to any one of the first aspect or the implementations of the first aspect.

[0060] For example, the computer-readable storage media include but are not limited to one or more of the following: a read-only memory (read-only memory, ROM), a programmable ROM (programmable ROM, PROM), an erasable PROM (erasable PROM, EPROM), a flash memory, an electrically EPROM (electrically EPROM, EEPROM), and a hard disk drive (hard disk drive).

[0061] Optionally, in an implementation, the foregoing storage medium may be specifically a non-volatile storage medium.

[0062] According to a ninth aspect, a computer-readable storage medium is provided, and includes computer program instructions. When the computer program instructions are executed by a compute device cluster, the compute device cluster performs the method according to any one of the first aspect or the implementations of the first aspect.

[0063] For example, the computer-readable storage media include but are not limited to one or more of the following: a read-only memory (read-only memory, ROM), a programmable ROM (programmable ROM, PROM), an erasable PROM (erasable PROM, EPROM), a flash memory, an electrically EPROM (electrically EPROM, EEPROM), and a hard disk drive (hard disk drive).

[0064] Optionally, in an implementation, the foregoing storage medium may be specifically a non-volatile storage medium.

BRIEF DESCRIPTION OF DRAWINGS

[0065] FIG. 1 is a block diagram of a cloud scenario to which an embodiment of this application is applicable;

[0066] FIG. 2 is a block diagram of a model inference system according to an embodiment of this application;

[0067] FIG. 3 is a block diagram of another model inference system according to an embodiment of this application;

[0068] FIG. 4 is a schematic flowchart of a model inference method according to an embodiment of this application;

[0069] FIG. 5 is a block diagram of a user setting of a draft generation model and a draft verification model according to an embodiment of this application;

[0070] FIG. 6 is a block diagram of another model inference system according to an embodiment of this application;

[0071] FIG. 7 is a block diagram of another model inference system according to an embodiment of this application;

[0072] FIG. 8 is a schematic flowchart of another model inference method according to an embodiment of this application;

[0073] FIG. 9 is a block diagram of another user setting of a draft generation model and a draft verification model according to an embodiment of this application;

[0074] FIG. 10 is a block diagram of another model inference system according to an embodiment of this application;

[0075] FIG. 11 is a block diagram of another model inference system according to an embodiment of this application;

[0076] FIG. 12 is a schematic flowchart of another model inference method according to an embodiment of this application;

[0077] FIG. 13 is a block diagram of another user setting of a draft generation model and a draft verification model according to an embodiment of this application;

[0078] FIG. 14 is a block diagram of another model inference system according to an embodiment of this application;

[0079] FIG. 15 is a block diagram of another model inference system according to an embodiment of this application;

[0080] FIG. 16 is a schematic flowchart of another model inference method according to an embodiment of this application;

[0081] FIG. 17 is a block diagram of user A setting of a draft generation model and a draft verification model according to an embodiment of this application;

[0082] FIG. 18 is a block diagram of another model inference system according to an embodiment of this application;

[0083] FIG. 19 is a block diagram of user B setting of a draft generation model and a draft verification model according to an embodiment of this application;

[0084] FIG. 20 is a block diagram of another model inference system according to an embodiment of this application;

[0085] FIG. 21 is a block diagram of a model inference apparatus 2100 according to an embodiment of this application;

[0086] FIG. 22 is a diagram of an architecture of a compute device 1500 according to an embodiment of this application;

[0087] FIG. 23 is a diagram of an architecture of a compute device cluster according to an embodiment of this application; and

[0088] FIG. 24 is a diagram of connecting compute devices 1500A and 1500B through a network according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0089] The following describes technical solutions of this application with reference to accompanying drawings.

[0090] Each aspect, embodiment, or feature is presented in this application with reference to a system including a plurality of devices, components, modules, and the like. It should be appreciated and understood that, each system may include another device, component, module, and the like, and/or may not include all devices, components, modules, and the like discussed with reference to the accompanying drawings. In addition, a combination of these solutions may be used.

[0091] In addition, in embodiments of this application, the terms such as "example" or "for example" are for representing giving an example, an illustration, or a description. Any embodiment or design scheme described as an "example" in this application should not be explained as being more preferred or having more advantages than another embodiment or design scheme. To be precise, the term "example" is for presenting a concept in a specific manner.

[0092] In embodiments of this application, "relevant (corresponding, relevant)" and "corresponding (corresponding)" may sometimes be mixed. It should be noted that meanings to be expressed by the two are consistent when a difference between them is not emphasized.

[0093] A service scenario described in embodiments of this application is intended to describe the technical solutions in embodiments of this application more clearly, and does not constitute a limitation on the technical solutions provided in embodiments of this application. A person of ordinary skill in the art may know that with the evolution of the network architecture and the emergence of a new service scenario, the technical solutions provided in embodiments of this application are also applicable to similar technical problems.

[0094] Reference to "an embodiment", "some embodiments", or the like described in this specification means that one or more embodiments of this application include a specific feature, structure, or characteristic described with reference to embodiments. Therefore, statements such as "in an embodiment", "in some embodiments", "in some other embodiments", and "in other embodiments" that appear at different places in this specification do not necessarily mean referring to a same embodiment. Instead, the statements mean "one or more but not all of embodiments", unless otherwise specifically emphasized in another manner. The terms "include", "contain", "have", and their variants all mean "include but are not limited to", unless otherwise specifically emphasized in another manner.

[0095] In this application, "at least one" means one or more, and "a plurality of" means two or more. The term "and/or" describes an association relationship for describing associated objects

and represents that three relationships may exist. For example, A and/or B may represent the following cases: Only A exists, both A and B exist, and only B exists, where A and B may be singular or plural. The character "/" generally indicates an "or" relationship between the associated objects. "At least one of the following items (pieces)" or a similar expression thereof means any combination of these items, including a single item (piece) or any combination of a plurality of items (pieces). For example, at least one item (piece) of a, b, or c may indicate: a, b, c, a and b, a and c, b and c, or a, b, and c, where a, b, and c may be singular or plural.

[0096] For ease of description, the following first describes concepts in embodiments of this application.

[0097] 1. Generation model

[0098] A generation model, which may also be referred to as a generative model, is a machine learning model. An objective of the generation model is to generate new data similar to training data. Specifically, the generation model is a method for generating data by using a probabilistic model, and may consider data generation as a process of extracting samples from prior distribution, thereby generating new data rather than merely classifying existing data.

[0099] 2. Autoregressive decoding

[00100] Autoregressive decoding is an important decoding policy in a generation model. Specifically, the generation model generates tokens in a sequence one by one, and an autoregressive decoding manner is token by token. That is, in an autoregressive decoding process, a probability distribution of a next token is calculated based on an existing context (that is, previously generated tokens), and a token is selected as a next output based on the probability distribution. This process is recursive, which means that each newly generated token becomes a condition for subsequent prediction. For example, in a text generation task, a first token of a sequence is generated first, and then a second token is predicted based on the token. This process is repeated until the complete sequence is generated.

[00101] It should be understood that the token may be a text, a character, a word, an image block, or a data unit of another type, which specifically depends on an application field and a task of the model.

[00102] 3. Draft

[00103] A draft includes a plurality of tokens having a context relationship, that is, a plurality of tokens having a context relationship are referred to as a draft. In a context of a generation model, a draft generally means a preliminary result generated by the generation model, for example, a preliminary version or an unfinished draft, and the result may require further revision or verification.

[00104] 4. Prompt

[00105] A prompt generally refers to a piece of text or information that serves as an initial input or guidance for a generation model to guide the generation model to generate a draft. That is, a prompt may be considered as a start point or trigger factor for generating a draft. By providing an appropriate prompt, the generation model can generate a draft related to content of the prompt.

[00106] 5. Verification model

[00107] A verification model verifies, based on an input prompt, a plurality of tokens included in a generated draft, and determines at least one token that passes the verification.

[00108] In recent years, with the rapid development of large models, smart applications built based on large models have been widely implemented, influencing the ways in which people produce and use information, for example, ChatGPT, Sora, and the like. A representative feature of large models is their huge quantity of parameters. To continuously improve an application effect of large models, current large models are still developing rapidly along the direction of increasing the quantity of parameters. In 2023, OpenAI released ChatGPT with hundreds of billions of parameters, and just a few months later, released GPT4 with trillions of parameters. A dialog effect of the model was significantly improved, which caused a continuous sensation in the industry. On one hand, the huge quantity of parameters makes large models successful; on the other hand, it also becomes a bottleneck hindering the widespread adoption of large models. One of the important reasons is high-cost service-based deployment of large models in an inference phase. Due to a huge quantity of parameters, large models require substantial compute resources during inference. However, due to the limitation of the auto-regressive token-by-token decoding generation method, utilization of the compute resources during large model inference is relatively low. How to enable models to make full use of compute resources during inference, increase a throughput rate of model inference, and reduce inference costs of large models is a difficult and important problem.

[00109] In solutions related to generative model inference, in two key phases of draft generation and draft verification, a draft generation model and a draft verification model have a strong coupling relationship in terms of a collaboration manner and a correspondence manner. The strong coupling relationship is fixed when a program is started. This results in poor flexibility of the draft generation model and the draft verification model in terms of the collaboration manner and the correspondence manner, thereby reducing model inference efficiency.

[00110] In view of this, an embodiment of this application provides a model inference method. The method can improve model inference efficiency.

[00111] In a possible implementation, the method provided in this embodiment of this application may be applied to a cloud service scenario, and a cloud management platform in the cloud service scenario performs the method. For ease of description, the following first describes

the cloud service scenario in detail with reference to FIG. 1.

[00112] FIG. 1 is a block diagram of a cloud scenario to which an embodiment of this application is applicable. As shown in FIG. 1, the cloud scenario may include a cloud management platform 110, an Internet 120, and a client 130.

[00113] As shown in FIG. 1, the cloud management platform 110 is configured to manage an infrastructure that provides a plurality of cloud services. The infrastructure includes a plurality of cloud data centers. Each cloud data center includes a plurality of servers. Each server includes a cloud service resource, to provide a corresponding cloud service for a tenant.

[00114] The cloud management platform 110 may be located in the cloud data center, and may provide an access interface (for example, an interface or an application programming interface (application programming interface, API)). The tenant may operate the client 130 to remotely access the access interface, to register a cloud account and a password on the cloud management platform 110 and log in to the cloud management platform 110. After the cloud management platform 110 successfully authenticates the cloud account and the password, the tenant may further pay on the cloud management platform 110 to select and purchase a virtual machine with a specific specification (processor, memory, and disk). After the payment for purchase succeeds, the cloud management platform 110 provides a remote login account and password of the purchased virtual machine, and the client 130 may remotely log in to the virtual machine, and install and run an application of the tenant in the virtual machine. Therefore, the tenant may create, manage, log in to, and operate the virtual machine in the cloud data center through the cloud management platform 110. The virtual machine may also be referred to as a cloud server (elastic compute service, ECS) or an elastic instance (different cloud service providers have different names).

[00115] It should be understood that the tenant of the cloud service may be an individual, an enterprise, a school, a hospital, an administrative agency, or the like.

[00116] Functions of the cloud management platform 110 include but are not limited to a user console, a computing management service, a network management service, a storage management service, an authentication service, and an image management service. The user console provides an interface or an API to interact with the tenant. The computing management service is used to manage servers on which a virtual machine and a container are run, and a bare-metal server. The network management service is used to manage a network service (for example, a gateway or a firewall). The storage management service is used to manage a storage service (for example, a data bucket service). The authentication service is used to manage a tenant account and password. The image management service is used to manage a virtual machine image. The tenant may use the client 130 to log in to the cloud management platform 110 through the Internet 120 to manage a rented cloud service.

[00117] FIG. 2 is a block diagram of a model inference system according to an embodiment of this application. As shown in FIG. 2, before inference of the model inference system (which may also be referred to as an efficient decoding inference system) is started, a system administrator needs to complete configurations for a draft generation model, a draft verification model, compute resources, a draft pool, and another related component. The model inference system loads configuration information and starts a service. During the entire service, the model inference system receives an external request, generates text through collaborative work of the draft generation model, the draft verification model, and the draft pool, and feeds back the text to a user. In addition, the model inference system collects statistics on and analyzes a large amount of running information in an inference process, and sends related results to the system administrator and an automatic scheduling module. The system administrator can manually or automatically update a system configuration. The updated configuration may be sent to the model inference system in real time and take effect in real time.

[00118] FIG. 3 is a block diagram of another model inference system according to an embodiment of this application. As shown in FIG. 3, the model inference system includes a draft generation model pool 310, a draft generation management unit 320, a draft verification model pool 330, and a draft verification management unit 340. The following separately describes parts included in the model inference system in detail.

[00119] 1. Draft generation model pool 310

[00120] The draft generation model pool 310 includes a plurality of draft generation models, for example, a draft generation model 1 to a draft generation model M. Specifically, the plurality of draft generation models may be connected to the draft generation model pool 310 in a hot swapping manner.

[00121] A type of the draft generation model in the draft generation model pool 310 is not specifically limited in this embodiment of this application. The type of the draft generation model may include but is not limited to a statistics-based natural language processing (natural language processing, NLP) model, a conventional deep learning model (for example, a recurrent neural network (recurrent neural network, RNN) or a long short-term memory (long short-Term memory, LSTM) network), a pre-trained model (for example, bidirectional encoder representations from transformers (bidirectional encoder representations from transformers, BERT) or a generative pre-trained transformer (generative pre-trained transformer, GPT)), a general large model (for example, Llama or Pangu), a domain-specific large model, and a user-defined model.

[00122] Each draft generation model transmits information to the draft generation management unit 320. The information includes but is not limited to: a label of the draft generation model, resource occupation of the draft generation model for inference (generating a draft), and a draft

generated by the draft generation model.

[00123] A form of the draft generated by the draft generation model is not specifically limited in this embodiment of this application. The form of the draft may include but is not limited to: an N-Gram, a phrase, a sentence, a paragraph, and the like.

[00124] 2. Draft generation management unit 320

[00125] The draft generation management unit 320 is configured to manage the draft generation models in the draft generation model pool 310. Specifically, the draft generation management unit 320 may send information to the draft generation model pool 310 based on determining of a current requirement of the model inference system for the draft and current usage of an inference resource. The information is used to control one or more draft generation models in the draft generation model pool 310.

[00126] In this embodiment of this application, the information sent by the draft generation management unit 320 to the draft generation model pool 310 may include but is not limited to: a label of the draft generation model, an instruction for performing operations on the draft generation model, and a prompt for the draft generation model to generate a draft. The operations performed on the draft generation model include but are not limited to: activating, hibernating, and releasing the draft generation model, and the like.

[00127] 3. Draft verification model pool 330

[00128] The draft verification model pool 330 includes a plurality of draft verification models, for example, a draft verification model 1 to a draft verification model N. Specifically, the plurality of draft verification models may be connected to the draft verification model pool 330 in a hot swapping manner. The draft verification model is configured to receive an external request (including prompts), generate a response (for example, generated content), and send the response to a user.

[00129] A type of the draft verification model in the draft verification model pool 330 is not specifically limited in this embodiment of this application, and the draft verification model may be an LLM generation model of any scale.

[00130] Each draft verification model transmits information to the draft verification management unit 340. The information includes but is not limited to: a label of the draft verification model, resource occupation of the draft verification model for inference (performing verification), a current prompt, a verification result generated by the draft verification model, and the like.

[00131] In this embodiment of this application, there are a plurality of correspondence manners between the draft verification model and the draft generation model. This is not specifically limited herein.

[00132] The following provides examples of several possible correspondence manners:

one draft generation model corresponds to one draft verification model;

multiple draft generation models correspond to one draft verification model;

one draft generation model corresponds to multiple draft verification models; and

multiple draft generation models correspond to multiple draft verification models.

[00133] It should be noted that the foregoing correspondence manners (correspondence modes) between the draft verification model and the draft generation model may be freely switched in real time, to complete dynamic combination.

[00134] 4. Draft verification management unit 340

[00135] The draft verification management unit 340 is configured to manage the draft verification models in the draft verification model pool 330. For example, the draft verification management unit 340 is mainly configured to distribute the draft received from the draft generation management unit 320 to the corresponding draft verification model.

[00136] The following uses the model inference system shown in FIG. 3 as an example to describe the model inference method provided in an embodiment of this application in detail with reference to FIG. 4. It should be understood that an example in FIG. 4 is merely intended to help a person skilled in the art understand embodiments of this application, but is not intended to limit embodiments of this application to a specific value or a specific scenario in the example in FIG. 4. It is clear that a person skilled in the art can make various equivalent modifications or changes based on the following example provided in FIG. 4, and such modifications and changes also fall within the scope of embodiments of this application.

[00137] FIG. 4 is a schematic flowchart of a model inference method according to an embodiment of this application. As shown in FIG. 4, the method may include steps 410 to 480. The following separately describes steps 410 to 480 in detail.

[00138] Step 410: A user completes setting of a draft generation model and a draft verification model.

[00139] In this embodiment of this application, the user may set a draft verification model and a draft generation model corresponding to the draft verification model.

[00140] It should be understood that the draft generation model corresponding to the draft verification model may be set by the user, or may be automatically recommended to the user by the system based on a draft verification model input by the user.

[00141] For example, as shown in FIG. 5, the draft verification model selected by the user is Llama 13B, and the selected draft generation models are the draft generation model 1 and the draft generation model 2. In this case, the model inference system is shown in FIG. 6.

[00142] Step 420: During running, after receiving prompts, the model inference system sends

the prompts to the draft verification model Llama 13B in the draft verification model pool 330.

[00143] In this embodiment of this application, during running, after receiving prompts, the model inference system sends the prompts to a draft verification model in the draft verification model pool 330, for example, sends the prompts to the Llama 13B in the draft verification model pool 330.

[00144] Step 430: The draft verification model Llama 13B generates a draft generation request based on the received prompts, and sends the draft generation request to the draft verification management unit 340.

[00145] In this embodiment of this application, after receiving the prompts, the draft verification model Llama 13B generates a draft generation request based on the received prompts, and sends the draft generation request to the draft verification management unit 340.

[00146] Step 440: The draft verification management unit 340 generates a new draft generation request based on the received draft generation request, and sends the new draft generation request to the draft generation management unit 320.

[00147] In this embodiment of this application, after receiving the draft generation request sent by the draft verification model Llama 13B, the draft verification management unit 340 generates a new draft generation request after adding information such as a quantity of to-be-generated drafts to the draft generation request, and sends the new draft generation request to the draft generation management unit 320.

[00148] Step 450: The draft generation management unit 320 sends the new draft generation request to a draft generation model that is in the draft generation model pool 310 and that corresponds to the draft verification model Llama 13B.

[00149] In this embodiment of this application, after receiving the new draft generation request from the draft verification management unit 340, the draft generation management unit 320 may send the new draft generation request to the draft generation model that is in the draft generation model pool 310 and that corresponds to the draft verification model Llama 13B, for example, the draft generation model 1 and the draft generation model 2 in the draft generation model pool 310.

[00150] Step 460: The draft generation model 1 and the draft generation model 2 each generate a draft based on the received new draft generation request, and feed back the drafts to the draft generation management unit 320.

[00151] In this embodiment of this application, after receiving the new draft generation request sent by the draft generation management unit 320, the draft generation model 1 and the draft generation model 2 each generate a draft based on the new draft generation request, and feed back the drafts generated by the draft generation model 1 and the draft generation model 2 to the draft generation management unit 320.

[00152] Step 470: After collecting the drafts, the draft generation management unit 320 sends the collected drafts to the draft verification model Llama 13B through the draft verification management unit 340.

[00153] In this embodiment of this application, after separately receiving the drafts sent by the draft generation model 1 and the draft generation model 2, the draft generation management unit 320 may send the received drafts to the draft verification management unit 340, and the draft verification management unit 340 sends the received drafts to the draft verification model Llama 13B.

[00154] Step 480: The draft verification model Llama 13B verifies one or more drafts corresponding to each prompt to form a new prompt.

[00155] In this embodiment of this application, the draft verification model Llama 13B may verify one or more drafts corresponding to each prompt. Specifically, the draft verification model Llama 13B may confirm, based on the one or more drafts corresponding to each prompt, at least one generated token, and concatenate the confirmed token after a corresponding prompt (the received prompt) to form a new prompt.

[00156] In this embodiment of this application, the draft verification model Llama 13B may further send a new prompt to the draft verification model Llama 13B, and repeat steps 430 to 480 for a plurality of iterations until the draft verification model Llama 13B completes content generation based on the input prompts, and feeds back the content to the user.

[00157] In the foregoing technical solution, correspondence manners between the draft generation model and the draft verification model are decoupled, and correspondence modes between the models may be dynamically combined in any proportion and may be adjusted in real time based on a status of the model inference system.

[00158] FIG. 7 is a block diagram of another model inference system according to an embodiment of this application. As shown in FIG. 7, the model inference system includes a draft generation model pool 310, a draft generation management unit 320, a draft verification model pool 330, a draft verification management unit 340, and a draft pool 350.

[00159] It should be understood that for descriptions of the draft generation model pool 310, the draft generation management unit 320, the draft verification model pool 330, and the draft verification management unit 340, refer to the descriptions in FIG. 3. Details are not described herein again. The following describes the functions of the draft pool 350 in detail.

[00160] The draft pool 350 is configured to store drafts generated by draft generation models and attribute label information of each draft.

[00161] The attribute label information of the draft is not specifically limited in this embodiment of this application. The attribute label information of the draft includes but is not

limited to: information such as a content field corresponding to the draft, a label of a generation model corresponding to the draft, a quantity of times that the draft is invoked, a quantity of times that the draft is hit, and a list of prompts that are hit by the draft.

[00162] It should be understood that in the model inference system shown in FIG. 7, the draft pool 350 is added on the basis of the model inference system in FIG. 3. Based on the model inference system in FIG. 7, the draft generation model and the draft verification model may collaborate in the following manners:

[00163] (1) Synchronous collaboration

[00164] The draft verification model sends a draft generation request and activates the corresponding draft generation model to generate a draft, and the generated draft is recorded in the draft pool 350 for subsequent reuse by the model or request.

[00165] (2) Asynchronous collaboration

[00166] The draft generation model generates a draft and stores the draft in the draft pool 350. The draft verification model sends a draft generation request and obtains the related draft from the draft pool 350.

[00167] (3) Hybrid collaboration

[00168] Some draft generation models and draft verification models collaborate synchronously, and some draft generation models and draft verification models collaborate asynchronously.

[00169] The following uses the model inference system shown in FIG. 7 as an example to describe another model inference method provided in an embodiment of this application in detail with reference to FIG. 8. It should be understood that an example in FIG. 8 is merely intended to help a person skilled in the art understand embodiments of this application, but is not intended to limit embodiments of this application to a specific value or a specific scenario in the example in FIG. 8. It is clear that a person skilled in the art can make various equivalent modifications or changes based on the following example provided in FIG. 8, and such modifications and changes also fall within the scope of embodiments of this application.

[00170] FIG. 8 is a schematic flowchart of another model inference method according to an embodiment of this application. As shown in FIG. 8, the method may include steps 810 to 880. The following separately describes steps 810 to 880 in detail.

[00171] Step 810: A user completes setting of a draft generation model and a draft verification model.

[00172] In this embodiment of this application, the user may set a draft verification model and a draft generation model corresponding to the draft verification model.

[00173] For example, as shown in FIG. 9, the draft verification model selected by the user is Llama 13B, and in the selected draft generation models, synchronous models are a draft generation

model 1 and a draft generation model 2, and an asynchronous model is a draft generation model 3. In this case, the model inference system is shown in FIG. 10.

[00174] Step 820: During running, after receiving prompts, the model inference system sends the prompts to the draft verification model Llama 13B in the draft verification model pool 330.

[00175] This step is similar to step 420. For details, refer to the descriptions in step 420. Details are not described herein again.

[00176] Step 830: The draft verification model Llama 13B generates a draft generation request based on the received prompts, and sends the draft generation request to the draft verification management unit 340.

[00177] Step 840: The draft verification management unit 340 generates a new draft generation request based on the received draft generation request, and sends the new draft generation request to the draft pool 350.

[00178] In this embodiment of this application, after receiving the draft generation request sent by the draft verification model Llama 13B, the draft verification management unit 340 generates a new draft generation request after adding information such as a quantity of to-be-generated drafts to the draft generation request, and sends the new draft generation request to the draft pool 350.

[00179] Step 850: After receiving the new draft generation request, the draft pool 350 obtains a corresponding Draft.

[00180] In this embodiment of this application, after receiving the new draft generation request sent by the draft verification management unit 340, the draft pool 350 may obtain a corresponding draft.

[00181] Specifically, the draft pool 350 simultaneously performs the following two steps:

[00182] 1. The draft pool 350 continues to pass the new draft generation request to the draft generation management unit 320. The draft generation management unit 320 sends the new draft generation request to synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B, for example, the draft generation model 1 and the draft generation model 2 in the draft generation model pool 310. The draft generation model 1 and the draft generation model 2 each generate a draft based on the received new draft generation request, and store, in the draft pool 350, the drafts generated by the draft generation model 1 and the draft generation model 2 through the draft generation management unit 320.

[00183] 2. The draft pool 350 searches the pool for a draft related to the new draft generation request.

[00184] Step 860: The draft pool 350 sends the obtained draft to the draft verification model Llama 13B through the draft verification management unit 340.

[00185] Step 870: The draft verification model Llama 13B verifies one or more drafts

corresponding to each prompt to form a new prompt.

[00186] In this embodiment of this application, the draft verification model Llama 13B may verify one or more drafts corresponding to each prompt. Specifically, the draft verification model Llama 13B may confirm, based on the one or more drafts corresponding to each prompt, at least one generated token, and concatenate the confirmed token after a corresponding prompt (the received prompt) to form a new prompt.

[00187] In this embodiment of this application, the draft verification model Llama 13B may further send verification information such as a hit rate of the draft to the draft pool 350 through the draft verification management unit 340, and the draft pool 350 records the verification information.

[00188] In this embodiment of this application, the draft verification model Llama 13B may further send a new prompt to the draft verification management unit 340, and repeat steps 830 to 870 for a plurality of iterations until the draft verification model Llama 13B completes content generation based on the input prompts, and feeds back the content to the user.

[00189] Step 880: When inference service load is low, the draft pool 350 activates, based on the verification information such as the hit rate of the collected draft, the synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B.

[00190] In this embodiment of this application, when the inference service load is low, the draft pool 350 forms a draft generation request based on the verification information such as the hit rate of the collected draft. The draft generation management unit 320 activates a synchronous model that is in the draft generation model pool 310 and that corresponds to the draft verification model Llama 13B, for example, the draft generation model 3. The draft generation model 3 generates a corresponding draft based on the draft generation request and stores the draft in the draft pool 350 for use in subsequent model inference.

[00191] In the foregoing technical solution, the draft generation model and the draft verification model may be fully decoupled. Correspondence modes and collaboration manners between the models may be dynamically combined in any proportion and may be adjusted in real time based on a status of the model inference system. In addition, the draft pool stores the drafts generated in a running process of the system, so that the drafts can be reused. This reduces computational power and time required for draft generation to some extent.

[00192] FIG. 11 is a block diagram of another model inference system according to an embodiment of this application. As shown in FIG. 11, the model inference system includes a draft generation model pool 310, a draft generation management unit 320, a draft verification model pool 330, a draft verification management unit 340, a draft pool 350, a draft analysis unit 360, and

an inference resource analysis unit 370.

[00193] It should be understood that, in the model inference system shown in FIG. 11, the draft analysis unit 360 and the inference resource analysis unit 370 are added on the basis of the model inference system in FIG. 7. Therefore, for descriptions of the draft generation model pool 310, the draft generation management unit 320, the draft verification model pool 330, and the draft verification management unit 340, refer to the descriptions in FIG. 3. For descriptions of the draft pool 350, refer to the descriptions in FIG. 7. Details are not described herein again.

[00194] 1. Draft analysis unit 360:

[00195] The draft analysis unit 360 collects statistics on, analyzes, and manages usage of drafts in the draft pool 350.

[00196] As an example, main functions of the draft analysis unit 360 include but are not limited to:

[00197] (a) Analyze existing drafts in the draft pool 350, and notify, based on an analysis result, the draft generation management unit 320 to generate a related draft.

[00198] (b) Analyze a prompt from the draft verification management unit 340, retrieve a related draft from the draft pool 350 through multi-dimensional information based on an analysis result, and if a related draft is missing, notify, based on the analysis result, the draft generation management unit 320 to generate a related draft.

[00199] (c) Receive draft information from the draft generation management unit 320 and verification information from the draft verification management unit 340, and complete operations such as screening, retrieval, modification, deletion, and addition on the drafts in the draft pool 350 based on the foregoing information.

[00200] 2. Inference resource analysis unit 370:

[00201] The inference resource analysis unit 370 is mainly configured to analyze current usage of an inference resource.

[00202] As an example, main functions of the inference resource analysis unit 370 include but are not limited to:

[00203] (a) Receive request information from the draft verification model sent by the draft verification management unit 340, evaluate how much computational power and video memory are currently used for model inference, and implement adaptive adjustment of a quantity of drafts based on a current inference resource status, model resource occupation, and a request status.

[00204] (b) Receive information about the draft generation model sent by the draft generation management unit 320, evaluate whether the draft generation model can be activated/hibernated, and update resource information after the draft generation model is released.

[00205] (c) Determine whether currently applied inference compute resources need to be

partially released and whether additional inference compute resources need to be applied for.

[00206] The following uses the model inference system shown in FIG. 11 as an example to describe another model inference method provided in an embodiment of this application in detail with reference to FIG. 12. It should be understood that an example in FIG. 12 is merely intended to help a person skilled in the art understand embodiments of this application, but is not intended to limit embodiments of this application to a specific value or a specific scenario in the example in FIG. 12. It is clear that a person skilled in the art can make various equivalent modifications or changes based on the following example provided in FIG. 12, and such modifications and changes also fall within the scope of embodiments of this application.

[00207] FIG. 12 is a schematic flowchart of another model inference method according to an embodiment of this application. As shown in FIG. 12, the method may include steps 1210 to 1280. The following separately describes steps 1210 to 1280 in detail.

[00208] Step 1210: A user completes setting of a draft generation model and a draft verification model.

[00209] In this embodiment of this application, the user may set a draft verification model and a draft generation model corresponding to the draft verification model.

[00210] For example, as shown in FIG. 13, the draft verification model selected by the user is Llama 13B, and in the selected draft generation models, synchronous models are a draft generation model 1 and a draft generation model 2, and an asynchronous model is a draft generation model 3. In addition, in FIG. 13, the user further chooses to enable a data adaptation function and an automatic scheduling function of a draft generation model. In this case, the model inference system is shown in FIG. 14.

[00211] Step 1220: During running, after receiving prompts, the model inference system sends the prompts to the draft verification model Llama 13B in the draft verification model pool 330.

[00212] This step is similar to step 420. For details, refer to the descriptions in step 420. Details are not described herein again.

[00213] Step 1230: The draft verification model Llama 13B generates a draft generation request based on the received prompts, and sends the draft generation request to the draft verification management unit 340.

[00214] Step 1240: The draft verification management unit 340 generates a new draft generation request based on the received draft generation request, and sends the new draft generation request to the draft analysis unit 360.

[00215] In this embodiment of this application, after receiving the draft generation request sent by the draft verification model Llama 13B, the draft verification management unit 340 automatically sets additional information (data adaptation) such as a quantity of to-be-generated

drafts for each prompt based on information such as resource utilization provided by the inference resource analysis unit 370, generates a new draft generation request, and sends the new draft generation request to the draft analysis unit 360.

[00216] Step 1250: After receiving the new draft generation request, the draft analysis unit 360 obtains a corresponding draft.

[00217] In this embodiment of this application, after receiving the new draft generation request sent by the draft verification management unit 340, the draft analysis unit 360 may obtain a corresponding draft.

[00218] Specifically, the draft analysis unit 360 simultaneously performs the following two steps:

[00219] 1. The draft analysis unit 360 continues to pass the new draft generation request to the draft generation management unit 320. The draft generation management unit 320 sends the new draft generation request to synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B, for example, the draft generation model 1 and the draft generation model 2 in the draft generation model pool 310. The draft generation model 1 and the draft generation model 2 each generate a draft based on the received new draft generation request, and send the drafts generated by the draft generation model 1 and the draft generation model 2 to the draft analysis unit 360 through the draft generation management unit 320.

[00220] 2. The draft analysis unit 360 generates multi-dimensional analysis information of prompts from multiple perspectives such as fields and semantics, and searches the draft pool 350 for a draft related to the new draft generation request based on the multi-dimensional analysis information of the prompts.

[00221] Step 1260: The draft analysis unit 360 sends the obtained draft to the draft verification model Llama 13B through the draft verification management unit 340.

[00222] Step 1270: The draft verification model Llama 13B verifies one or more drafts corresponding to each prompt to form a new prompt.

[00223] In this embodiment of this application, the draft verification model Llama 13B may verify one or more drafts corresponding to each prompt. Specifically, the draft verification model Llama 13B may confirm, based on the one or more drafts corresponding to each prompt, at least one generated token, and concatenate the confirmed token after a corresponding prompt (the received prompt) to form a new prompt.

[00224] In this embodiment of this application, the draft verification model Llama 13B may further send verification information such as a hit rate of the draft to the draft analysis unit 360 through the draft verification management unit 340, and the draft analysis unit 360 records the

verification information.

[00225] In this embodiment of this application, the draft verification model Llama 13B may further send a new prompt to the draft verification model Llama 13B, and repeat steps 1230 to 1270 for a plurality of iterations until the draft verification model Llama 13B completes content generation based on the input prompts, and feeds back the content to the user.

[00226] Step 1280: When inference service load is low, the draft analysis unit 360 activates, based on the verification information such as the hit rate of the collected draft, the synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B.

[00227] In this embodiment of this application, when the inference service load is low, the draft analysis unit 360 forms a draft generation request based on the verification information such as the hit rate of the collected draft. The draft generation management unit 320 activates a synchronous model that is in the draft generation model pool 310 and that corresponds to the draft verification model Llama 13B, for example, the draft generation model 3, based on information such as current inference resource utilization provided by the inference resource analysis unit 370. The draft generation model 3 generates a corresponding draft based on the draft generation request and stores the draft in the draft pool 350 for use in subsequent model inference.

[00228] In this embodiment of this application, if the draft generation management unit 320 requires activation of the draft generation model 3, but currently there is no sufficient inference resource, the inference resource analysis unit 370 provides a new inference resource for the draft generation model 3 to generate the draft.

[00229] In this embodiment of this application, if inference resource usage information provided by the inference resource analysis unit 370 indicates that there is an unused inference resource currently, and the draft generation model 3 is not activated, the inference resource analysis unit 370 automatically releases the inference resource.

[00230] In the foregoing technical solution, a quantity of to-be-generated drafts may be adaptively adjusted based on a current resource status. The model inference system may complete self-adjustment based on different statuses, including adjustment of a correspondence manner and a collaboration manner between a draft generation model and a draft verification model, self-maintenance of a draft pool, and application and release of an inference resource. In addition, the draft analysis unit 360 generates multi-dimensional analysis information of prompts from multiple perspectives such as fields and semantics, and searches the draft pool 350 for a draft related to the draft generation request based on the multi-dimensional analysis information of the prompts, so that draft recall is more accurate, and a higher draft hit rate is achieved in a verification phase.

[00231] FIG. 15 is a block diagram of another model inference system according to an

embodiment of this application. As shown in FIG. 15, the model inference system includes a draft generation model pool 310, a draft generation management unit 320, a draft verification model pool 330, a draft verification management unit 340, a draft pool 350, a draft analysis unit 360, and an inference resource analysis unit 370.

[00232] It should be understood that the model inference system shown in FIG. 15 is based on the model inference system shown in FIG. 11, so that the draft pool 350 supports a multi-tenant mode. The draft pool 350 mainly includes two parts. One is a preset draft pool (which may also be referred to as a shared draft pool), where the shared draft pool may be shared among multiple tenants. The other is a private draft pool of each tenant, and draft pools of the tenants are isolated from each other. When configuring respective inference services, different users may choose whether to use the shared draft pool for cold start. After the service is started, the system allocates an independent draft pool for each user, and drafts generated in a running process of the inference service of each user are written to the draft pool of the user.

[00233] The following uses the model inference system shown in FIG. 15 as an example to describe another model inference method provided in an embodiment of this application in detail with reference to FIG. 16. It should be understood that an example in FIG. 16 is merely intended to help a person skilled in the art understand embodiments of this application, but is not intended to limit embodiments of this application to a specific value or specific scenario in the example in FIG. 16. It is clear that a person skilled in the art can make various equivalent modifications or changes based on the following example provided in FIG. 16, and such modifications and changes also fall within the scope of embodiments of this application.

[00234] FIG. 16 is a schematic flowchart of a model inference method according to an embodiment of this application. As shown in FIG. 16, the method may include steps 1610 to 1680. The following separately describes steps 1610 to 1680 in detail.

[00235] Step 1610: A user A completes setting of a draft generation model and a draft verification model.

[00236] For example, as shown in FIG. 17, the draft verification model selected by the user A is Llama 13B, and in the selected draft generation models, synchronous models are a draft generation model 1 and a draft generation model 2, and an asynchronous model is a draft generation model 3. In addition, in FIG. 17, the user A further chooses to enable a data adaptation function, enable an automatic scheduling function of a draft generation model, and enable use of the shared draft pool in the draft pool 350. In this case, the model inference system is shown in FIG. 18. The draft pool 350 in FIG. 18 includes a shared draft pool and a private draft pool of the user A.

[00237] Step 1620: During running, after receiving prompts, the model inference system sends

the prompts to the draft verification model Llama 13B in the draft verification model pool 330.

[00238] This step is similar to step 420. For details, refer to the descriptions in step 420. Details are not described herein again.

[00239] Step 1630: The draft verification model Llama 13B generates a draft generation request based on the received prompts, and sends the draft generation request to the draft verification management unit 340.

[00240] Step 1640: The draft verification management unit 340 generates a new draft generation request based on the received draft generation request, and sends the new draft generation request to the draft analysis unit 360.

[00241] Step 1650: After receiving the new draft generation request, the draft analysis unit 360 obtains a corresponding draft.

[00242] In this embodiment of this application, after receiving the new draft generation request sent by the draft verification management unit 340, the draft analysis unit 360 may obtain a corresponding draft.

[00243] Specifically, the draft analysis unit 360 simultaneously performs the following three steps:

[00244] 1. The draft analysis unit 360 continues to pass the new draft generation request to the draft generation management unit 320. The draft generation management unit 320 sends the new draft generation request to synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B, for example, the draft generation model 1 and the draft generation model 2 in the draft generation model pool 310. The draft generation model 1 and the draft generation model 2 each generate a draft based on the received new draft generation request, and send the drafts generated by the draft generation model 1 and the draft generation model 2 to the draft analysis unit 360 through the draft generation management unit 320.

[00245] 2. The draft analysis unit 360 generates multi-dimensional analysis information of prompts from multiple perspectives such as fields and semantics, and searches the shared draft pool in the draft pool 350 for a draft related to the new draft generation request based on the multi-dimensional analysis information of the prompts.

[00246] 3. The draft analysis unit 360 generates multi-dimensional analysis information of prompts from multiple perspectives such as fields and semantics, and searches the private draft pool of the user A in the draft pool 350 for a draft related to the new draft generation request based on the multi-dimensional analysis information of the prompts.

[00247] Step 1660: The draft analysis unit 360 sends the obtained draft to the draft verification model Llama 13B through the draft verification management unit 340.

[00248] Step 1670: The draft verification model Llama 13B verifies one or more drafts corresponding to each prompt to form a new prompt.

[00249] In this embodiment of this application, the draft verification model Llama 13B may verify one or more drafts corresponding to each prompt. Specifically, the draft verification model Llama 13B may confirm, based on the one or more drafts corresponding to each prompt, at least one generated token, and concatenate the confirmed token after a corresponding prompt (the received prompt) to form a new prompt.

[00250] In this embodiment of this application, the draft verification model Llama 13B may further send verification information such as a hit rate of the draft to the draft analysis unit 360 through the draft verification management unit 340, and the draft analysis unit 360 records the verification information. The draft analysis unit 360 analyzes the currently obtained draft based on the verification information, and completes operations on a related draft in the private draft pool of the user A based on an analysis result.

[00251] In this embodiment of this application, the draft verification model Llama 13B may further send a new prompt to the draft verification model Llama 13B, and repeat steps 1630 to 1670 for a plurality of iterations until the draft verification model Llama 13B completes content generation based on the input prompts, and feeds back the content to the user A.

[00252] Step 1680: When inference service load is low, the draft analysis unit 360 activates, based on the verification information such as the hit rate of the collected draft, the synchronous models that are in the draft generation model pool 310 and that correspond to the draft verification model Llama 13B.

[00253] In this embodiment of this application, when the inference service load is low, the draft analysis unit 360 forms a draft generation request based on the verification information such as the hit rate of the collected draft. The draft generation management unit 320 activates a synchronous model that is in the draft generation model pool 310 and that corresponds to the draft verification model Llama 13B, for example, the draft generation model 3, based on information such as current inference resource utilization provided by the inference resource analysis unit 370. The draft generation model 3 generates a corresponding draft based on the draft generation request and stores the draft in the private draft pool of the user A in the draft pool 350 for use in subsequent model inference.

[00254] Similarly, if the draft generation management unit 320 requires activation of the draft generation model 3, but currently there is no sufficient inference resource, the inference resource analysis unit 370 provides a new inference resource for the draft generation model 3 to generate the draft. If inference resource usage information provided by the inference resource analysis unit 370 indicates that there is an unused inference resource currently, and the draft generation model

3 is not activated, the inference resource analysis unit 370 automatically releases the inference resource.

[00255] Optionally, in some embodiments, it is assumed that a user B also completes system configuration. For example, as shown in FIG. 19, the draft verification model selected by the user B is Llama 7B, and in the selected draft generation models, synchronous models are a draft generation model 1 and a draft generation model 4, and an asynchronous model is a draft generation model 5. In addition, in FIG. 19, the user B also chooses to enable a data adaptation function, enable an automatic scheduling function of a draft generation model, and enable use of the shared draft pool in the draft pool 350. In this case, the model inference system is shown in FIG. 20. The draft pool 350 in FIG. 20 includes a shared draft pool, a private draft pool of the user A, and a private draft pool of the user B.

[00256] It should be noted that the user B may also complete the model inference process based on steps 1610 to 1680 performed by the user A. A difference lies in that the draft verification model of the user B is Llama 7B, synchronous models are a draft generation model 1 and a draft generation model 4, the draft generation model 1 is shared with the user A, and an asynchronous model is a draft generation model 5. In an inference process of the user B, the draft analysis unit 360 can read the shared draft pool in the draft pool 350, but can perform operations such as reading and writing only on the private draft pool of the user B in the draft pool 350.

[00257] In the foregoing technical solution, the shared draft pool in the draft pool may be shared by multiple tenants to help a user complete inference acceleration during cold start. Private draft pools of different users in the draft pool are independent of each other, so that drafts generated by different users in an inference service running process can be independently maintained.

[00258] The foregoing describes the model inference method provided in embodiments of this application in detail with reference to FIG. 1 to FIG. 20. The following describes apparatus embodiments in this application in detail with reference to FIG. 21 to FIG. 24. It should be understood that the descriptions of the method embodiments correspond to descriptions of the apparatus embodiments. Therefore, for a part that is not described in detail, refer to the foregoing method embodiments.

[00259] FIG. 21 is a block diagram of a model inference apparatus 2100 according to an embodiment of this application. The apparatus 2100 may be implemented by software, hardware, or a combination of software and hardware. The apparatus 2100 provided in this embodiment of this application may implement the method procedure shown in the embodiments of this application. The apparatus 2100 includes a receiving module 2110, a processing module 2120, and a feedback module 2130. The receiving module 2110 is configured to receive text prompt information input by a first user, where the text prompt information is used to generate text

information. The processing module 2120 is configured to determine at least one draft verification model from a draft verification model pool, where the draft verification model pool includes a plurality of draft verification models; determine, from a draft generation model pool, at least one draft generation model corresponding to the at least one draft verification model, where the draft generation model pool includes a plurality of draft generation models, and the at least one draft generation model includes a draft synchronous generation model and a draft asynchronous generation model; obtain draft generation request information based on the text prompt information, and send the draft generation request information to the draft synchronous generation model, where the draft synchronous generation model is configured to generate a draft based on the draft generation request information; and/or obtain, from a draft pool, a draft related to the text prompt information, where the draft pool includes a draft generated by the draft asynchronous generation model; and verify, by using the at least one draft verification model, the draft generated by the draft synchronous generation model and/or the draft obtained from the draft pool, to obtain a verification result. The feedback module 2130 is configured to generate the text information based on the verification result, and feed back the text information to the first user.

[00260] Optionally, correspondence modes between the at least one draft verification model and the at least one draft generation model include: one-to-one, one-to-many, many-to-one, and many-to-many.

[00261] Optionally, the apparatus 2100 further includes: a storage module, configured to store, in the draft pool, the draft generated by the draft synchronous generation model.

[00262] Optionally, the draft pool further stores attribute information of the drafts.

[00263] Optionally, the processing module 2120 is specifically configured to: obtain, from the draft pool, the draft related to the text prompt information based on the attribute information of the drafts stored in the draft pool.

[00264] Optionally, the processing module 2120 is further configured to retrieve and/or update the drafts stored in the draft pool based on the attribute information of the drafts stored in the draft pool.

[00265] Optionally, the processing module 2120 is further configured to perform operations on the draft generation models included in the draft generation model pool based on the attribute information of the drafts stored in the draft pool, where the operations include: activation, hibernation, and release.

[00266] Optionally, the attribute information of the draft includes at least one of the following: a field of the draft, content of the draft, a label of the draft generation model corresponding to the draft, a quantity of times that the draft is invoked, a quantity of times that the draft is hit, and a list of hit text prompt information.

[00267] Optionally, the draft generation request information further includes a quantity of to-be-generated drafts, the quantity of to-be-generated drafts is determined based on usage of a model inference resource, and the model inference resource is used by a draft generation model in the draft generation model pool to generate a draft and/or by a draft verification model in the draft verification model pool to perform a verification process.

[00268] Optionally, the processing module 2120 is further configured to: when inference load is less than a preset threshold, activate the draft asynchronous generation model to generate a draft. The storage module is further configured to store, in the draft pool, the draft generated by the draft asynchronous generation model.

[00269] Optionally, the processing module 2120 is further configured to automatically request/release the model inference resource based on usage of the model inference resource.

[00270] Optionally, the draft pool includes a shared draft pool and a private draft pool of the first user, and the processing module 2120 is specifically configured to: obtain the draft related to the text prompt information from the shared draft pool and/or the private draft pool of the first user.

[00271] Optionally, the storage module is specifically configured to: store, in the private draft pool of the first user, the draft generated by the draft synchronous generation model.

[00272] Optionally, the storage module is further configured to store, in the private draft pool of the first user, the draft generated by the draft asynchronous generation model.

[00273] The apparatus 2100 herein may be embodied in a form of functional module. The term "module" herein may be implemented in a form of software and/or hardware, and this is not specifically limited.

[00274] For example, the "module" may be a software program, a hardware circuit, or a combination thereof that implements the foregoing functions. For example, the following uses the receiving module 2110 as an example to describe an implementation of the receiving module 2110. Similarly, for an implementation of another module, for example, the processing module 2120, the feedback module 2130, or the storage module, refer to an implementation of the receiving module 2110.

[00275] The receiving module 2110 is used as an example of a software functional unit, and the receiving module 2110 may include code running on a computing instance. The computing instance may include at least one of a physical host (compute device), a virtual machine, and a container. Further, there may be one or more computing instances. For example, the receiving module 2110 may include code running on a plurality of hosts/virtual machines/containers. It should be noted that the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same region (region), or may be distributed in different regions. Further, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in

a same availability zone (availability zone, AZ), or may be distributed in different AZs. Each AZ includes one data center or a plurality of data centers that are geographically close to each other. Generally, one region may include a plurality of AZs.

[00276] Similarly, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same virtual private cloud (virtual private cloud, VPC), or may be distributed in a plurality of VPCs. Generally, one VPC is disposed in one region. A communication gateway needs to be disposed in each VPC for communication between two VPCs in a same region and cross-region communication between VPCs in different regions. The VPCs are interconnected through the communication gateway.

[00277] The receiving module 2110 is used as an example of a hardware functional unit, and the receiving module 2110 may include at least one compute device such as a server. Alternatively, the receiving module 2110 may be a device implemented by using an application-specific integrated circuit (application-specific integrated circuit, ASIC) or a programmable logic device (programmable logic device, PLD), or the like. The PLD may be implemented by a complex programmable logic device (complex programmable logic device, CPLD), a field-programmable gate array (field-programmable gate array, FPGA), generic array logic (generic array logic, GAL), or any combination thereof.

[00278] A plurality of compute devices included in the receiving module 2110 may be distributed in a same region, or may be distributed in different regions. The plurality of compute devices included in the receiving module 2110 may be distributed in a same AZ, or may be distributed in different AZs. Similarly, the plurality of compute devices included in the receiving module 2110 may be distributed in a same VPC, or may be distributed in a plurality of VPCs. The plurality of compute devices may be any combination of compute devices such as a server, an ASIC, a PLD, a CPLD, an FPGA, and GAL.

[00279] Therefore, modules in examples described in embodiments of this application can be implemented by electronic hardware or a combination of computer software and electronic hardware. Whether the functions are performed by hardware or software depends on particular applications and design constraint conditions of the technical solutions. A person skilled in the art may use different methods to implement the described functions for each particular application, but it should not be considered that the implementation goes beyond the scope of this application.

[00280] It should be noted that when the apparatus provided in the foregoing embodiment performs the foregoing method, division into the foregoing functional modules is merely used as an example for description. During actual application, the foregoing functions may be allocated as needed to different functional modules for implementation, that is, an internal structure of the apparatus is divided into different functional modules to implement all or some of the functions

described above. For example, the receiving module 2110 may be configured to perform any step in the foregoing method, the processing module 2120 may be configured to perform any step in the foregoing method, the feedback module 2130 may be configured to perform any step in the foregoing method, and the storage module may be configured to perform any step in the foregoing method. Steps implemented by the receiving module 2110, the processing module 2120, the feedback module 2130, and the storage module may be specified as required. The receiving module 2110, the processing module 2120, the feedback module 2130, and the storage module respectively implement different steps in the foregoing method to implement all functions of the foregoing apparatus.

[00281] In addition, the apparatus embodiments and the method embodiments provided in the foregoing embodiments belong to a same concept. For specific implementation processes thereof, refer to the method embodiments. Details are not described herein again.

[00282] The method provided in embodiments of this application may be performed by a compute device, and the compute device may also be referred to as a computer system. The compute device may include a hardware layer, an operating system layer running above the hardware layer, and an application layer running above the operating system layer. The hardware layer includes hardware, for example, a processing unit, a memory, and a memory control unit. Subsequently, functions and structures of the hardware are described in detail. The operating system is any one or more computer operating systems through a process (process), for example, a Linux operating system, a Unix operating system, an Android operating system, an iOS operating system, or a Windows operating system, that implement service processing. The application layer includes applications such as a browser, an address book, word processing software, and instant messaging software. In addition, optionally, the computer system is a handheld device, for example, a smartphone, or a terminal device, for example, a personal computer. This is not particularly limited in this application, provided that the method according to embodiments of this application can be implemented. The method provided in embodiments of this application may be performed by the compute device or a functional module that is in the compute device and that can invoke and execute a program.

[00283] The following describes a compute device provided in an embodiment of this application in detail with reference to FIG. 22.

[00284] FIG. 22 is a diagram of an architecture of a compute device 1500 according to an embodiment of this application. The compute device 1500 may be a server, a computer, or another device with a computing capability. The compute device 1500 shown in FIG. 22 includes at least one processor 1510 and a memory 1520.

[00285] It should be understood that a quantity of processors and a quantity of memories in the

compute device 1500 are not limited in this application.

[00286] The processor 1510 executes instructions in the memory 1520, so that the compute device 1500 implements the method provided in this application. Alternatively, the processor 1510 executes the instructions in the memory 1520, so that the compute device 1500 implements the functional modules provided in this application, to implement the method provided in this application.

[00287] Optionally, the compute device 1500 further includes a communication interface 1530. The communication interface 1530 uses a transceiver module such as but not limited to a network interface card or a transceiver, to implement communication between the compute device 1500 and another device or a communication network.

[00288] Optionally, the compute device 1500 further includes a system bus 1540. The processor 1510, the memory 1520, and the communication interface 1530 are separately connected to the system bus 1540. The processor 1510 can access the memory 1520 through the system bus 1540. For example, the processor 1510 can read and write data or execute code in the memory 1520 through the system bus 1540. The system bus 1540 is a peripheral component interconnect express (peripheral component interconnect express, PCI) bus, an extended industry standard architecture (extended industry standard architecture, EISA) bus, or the like. The system bus 1540 is classified into an address bus, a data bus, a control bus, or the like. For ease of representation, the bus is represented by only one bold line in FIG. 22. However, it does not mean that there is only one bus or one type of bus.

[00289] In a possible implementation, a function of the processor 1510 is mainly to interpret instructions (or code) of a computer program and process data in computer software. The instructions of the computer program and the data in the computer software may be stored in the memory 1520 or a cache 1516.

[00290] Optionally, the processor 1510 may be an integrated circuit chip and has a signal processing capability. By way of example and not limitation, the processor 1510 is a general-purpose processor, a digital signal processor (digital signal processor, DSP), an application-specific integrated circuit (application-specific integrated circuit, ASIC), a field programmable gate array (field programmable gate array, FPGA) or another programmable logic device, a discrete gate or transistor logic device, or a discrete hardware assembly. The general-purpose processor is a microprocessor or the like. For example, the processor 1510 is a central processing unit (central processing unit, CPU).

[00291] Optionally, each processor 1510 includes at least one processing unit 1512 and a memory control unit 1514.

[00292] Optionally, the processing unit 1512 is also referred to as a core (core) or a kernel, and

is the most important component of the processor. The processing unit 1512 is made of monocrystalline silicon through a specific production process. All computation, accept commands, storage commands, and data processing of the processor are executed by the core. The processing unit independently runs program instructions, and increases a running speed of a program through a parallel computing capability. Various processing units have fixed logical structures. For example, the processing unit includes logical units such as a level 1 cache, a level 2 cache, an execution unit, an instruction level unit, and a bus interface.https://baike.baidu.com/item/CPU corehttps://baike.baidu.com/item/logical structurehttps://baike.baidu.com/item/L1 cachehttps://baike.baidu.com/item/L2 cachehttps://baike.baidu.com/item/bus interfacehttps://baike.baidu.com/item/logical unit

[00293] In an implementation example, the memory control unit 1514 is configured to control data exchange between the memory 1520 and the processing unit 1512. Specifically, the memory control unit 1514 receives a memory access request from the processing unit 1512, and controls access to the memory based on the memory access request. By way of example and not limitation, the memory control unit is a device, for example, a memory management unit (memory management unit, MMU).

[00294] In an implementation example, each memory control unit 1514 performs addressing for the memory 1520 through the system bus. In addition, an arbiter (not shown in FIG. 22) is configured in the system bus, and the arbiter is responsible for processing and coordinating contention-based access of a plurality of processing units 1512.

[00295] In an implementation example, the processing unit 1512 is in communication connection with the memory control unit 1514 through a connection line inside a chip, for example, an address line, to implement communication between the processing unit 1512 and the memory control unit 1514.

[00296] Optionally, each processor 1510 further includes a cache 1516, and the cache is a data exchange buffer (referred to as a cache). When the processing unit 1512 needs to read data, the processing unit 1512 first searches the cache for required data. If the required data is found, the processing unit 1512 directly reads the data. If the required data is not found, the processing unit 1512 searches the memory for the required data. Because the cache runs much faster than the memory, a function of the cache is to help the processing unit 1512 run faster.

[00297] The memory 1520 can provide running space for a process in the compute device 1500. For example, the memory 1520 stores a computer program (specifically, code of the program) for generating the process. After the computer program is run by the processor to generate the process, the processor allocates corresponding storage space to the process in the memory 1520. Further, the storage space further includes a text segment, an initial data segment, an uninitialized data

segment, a stack segment, a heap segment, and the like. The memory 1520 stores, in the storage space corresponding to the process, data generated during running of the process, for example, intermediate data or process data.

[00298] Optionally, the memory is also referred to as an internal memory, and a function of the memory is to temporarily store operation data in the processor 1510 and data exchanged with an external storage such as a hard disk. Provided that the computer runs, the processor 1510 loads, to the memory for an operation, data on which the operation needs to be performed, and the processing unit 1512 sends a result after the operation is completed.

[00299] By way of example and not limitation, the memory 1520 is a volatile memory or a non-volatile memory, or may include both a volatile memory and a non-volatile memory. The non-volatile memory is a read-only memory (read-only memory, ROM), a programmable read-only memory (programmable ROM, PROM), an erasable programmable read-only memory (erasable PROM, EPROM), an electrically erasable programmable read-only memory (electrically EPROM, EEPROM), or a flash memory. The volatile memory is a random access memory (random access memory, RAM) and serves as an external cache. Through example but not limitative description, many forms of RAMs may be used, for example, a static random access memory (static RAM, SRAM), a dynamic random access memory (dynamic RAM, DRAM), a synchronous dynamic random access memory (synchronous DRAM, SDRAM), a double data rate synchronous dynamic random access memory (double data rate SDRAM, DDR SDRAM), an enhanced synchronous dynamic random access memory (enhanced SDRAM, ESDRAM), a synchronous link dynamic random access memory (synchlink DRAM, SLDRAM), and a direct rambus random access memory (direct rambus RAM, DR RAM). It should be noted that the memory 1520 in the system and method described in this specification is intended to include but is not limited to these memories and any memory of another proper type.

[00300] The listed structure of the compute device 1500 is merely an example for description, and this application is not limited thereto. The compute device 1500 in this embodiment of this application includes various types of hardware in a computer system in the conventional technology. For example, the compute device 1500 further includes a memory other than the memory 1520, for example, a magnetic disk memory. A person skilled in the art should understand that the compute device 1500 may further include another component required for implementing normal running. In addition, according to a specific requirement, a person skilled in the art should understand that the compute device 1500 may further include a hardware component for implementing another additional function. In addition, a person skilled in the art should understand that the compute device 1500 may alternatively include only a component required for implementing embodiments of this application, and does not need to include all the components

shown in FIG. 22.

[00301] An embodiment of this application further provides a compute device cluster. The compute device cluster includes at least one compute device. The compute device may be a server. In some embodiments, the compute device may alternatively be a terminal device, for example, a desktop computer, a notebook computer, or a smartphone.

[00302] As shown in FIG. 23, the compute device cluster includes at least one compute device 1500. The memory 1520 in one or more compute devices 1500 in the compute device cluster may store same instructions used to perform the foregoing method.

[00303] In some possible implementations, the memory 1520 in the one or more compute devices 1500 in the compute device cluster may also separately store some instructions used to perform the foregoing method. In other words, a combination of one or more compute devices 1500 may jointly execute instructions used to perform the foregoing method.

[00304] It should be noted that the memories 1520 in different compute devices 1500 in the compute device cluster may store different instructions, which are respectively used to perform some functions of the foregoing apparatus. In other words, instructions stored in the memories 1520 in different compute devices 1500 may implement functions of one or more modules in the foregoing apparatus.

[00305] In some possible implementations, the one or more compute devices in the compute device cluster may be connected through a network. The network may be a wide area network, a local area network, or the like. FIG. 24 shows a possible implementation. As shown in FIG. 24, two compute devices 1500A and 1500B are connected through a network. Specifically, each compute device is connected to the network through a communication interface in the compute device.

[00306] It should be understood that functions of the compute device 1500A shown in FIG. 24 may also be completed by a plurality of compute devices 1500. Similarly, functions of the compute device 1500B may be alternatively completed by a plurality of compute devices 1500.

[00307] In this embodiment, a computer program product including instructions is further provided. The computer program product may be software or a program product that includes instructions and that can run on a compute device or that is stored in any usable medium. When the computer program product runs on a compute device, the compute device is enabled to perform the method provided above, or the compute device is enabled to implement functions of the apparatus provided above.

[00308] In this embodiment, a computer-readable storage medium is further provided. The computer-readable storage medium may be any usable medium that can be stored by a compute device, or a data storage device such as a data center, including one or more usable media. The

usable medium may be a magnetic medium (for example, a floppy disk, a hard disk, or a magnetic tape), an optical medium (for example, a DVD), a semiconductor medium (for example, a solid-state drive), or the like. The computer-readable storage medium includes instructions. When the instructions in the computer-readable storage medium are executed on the compute device, the compute device is enabled to perform the method provided above.

[00309] It should be understood that sequence numbers of the foregoing processes do not mean execution sequences in various embodiments of this application. The execution sequences of the processes should be determined according to functions and internal logic of the processes, and should not be construed as any limitation on the implementation processes of embodiments of this application.

[00310] A person of ordinary skill in the art may be aware that, units and algorithm steps in the examples described with reference to embodiments disclosed in this specification may be implemented by electronic hardware or a combination of computer software and electronic hardware. Whether the functions are performed by hardware or software depends on particular applications and design constraint conditions of the technical solutions. A person skilled in the art may use different methods to implement the described functions for each particular application, but it should not be considered that the implementation goes beyond the scope of this application.

[00311] It may be clearly understood by a person skilled in the art that, for the purpose of convenient and brief description, for a detailed working process of the foregoing system, apparatus, and unit, refer to a corresponding process in the foregoing method embodiments. Details are not described herein again.

[00312] In the several embodiments provided in this application, it should be understood that the disclosed system, apparatus, and method may be implemented in other manners. For example, the apparatus embodiments described above are merely examples. For example, the unit division is merely logical function division and may be other division in actual implementation. For example, a plurality of units or components may be combined or integrated into another system, or some features may be ignored or not performed. In addition, the displayed or discussed mutual couplings or direct couplings or communication connections may be implemented through some interfaces. The indirect couplings or communication connections between the apparatuses or units may be implemented in an electrical form, a mechanical form, or another form.

[00313] The units described as separate parts may or may not be physically separate, and parts displayed as units may or may not be physical units, may be located in one position, or may be distributed on a plurality of network units. Some or all of the units may be selected based on actual requirements to achieve the objectives of the solutions of embodiments.

[00314] In addition, functional units in embodiments of this application may be integrated into

one processing unit, each of the units may exist alone physically, or two or more units are integrated into one unit.

[00315] When the functions are implemented in the form of a software functional unit and sold or used as an independent product, the functions may be stored in a computer-readable storage medium. Based on such an understanding, the technical solutions of this application essentially, the part contributing to the conventional technology, or some of the technical solutions may be implemented in a form of a software product. The computer software product is stored in a storage medium, and includes several instructions for enabling a computer device (which may be a personal computer, a server, a network device, or the like) to perform all or some of the steps of the methods described in embodiments of this application. The foregoing storage medium includes any medium that can store program code, such as a USB flash drive, a removable hard disk, a read-only memory (read-only memory, ROM), a random access memory (random access memory, RAM), a magnetic disk, or an optical disc.

[00316] The foregoing descriptions are merely specific implementations of this application, but are not intended to limit the protection scope of this application. Any variation or replacement readily figured out by a person skilled in the art within the technical scope disclosed in this application shall fall within the protection scope of this application. Therefore, the protection scope of this application shall be subject to the protection scope of the claims.

CLAIMS

1. A model inference method, wherein the method comprises:

determining at least one draft verification model from a draft verification model pool, wherein the draft verification model pool comprises a plurality of draft verification models;

determining, from a draft generation model pool, at least one draft generation model corresponding to the at least one draft verification model, wherein the draft generation model pool comprises a plurality of draft generation models;

receiving text prompt information input by a first user, wherein the text prompt information is used to generate text information;

obtaining, from a draft pool, a target draft related to the text prompt information, wherein the draft pool stores a plurality of drafts, and the plurality of drafts comprise a draft generated by the at least one draft generation model;

verifying the target draft by using the at least one draft verification model, and obtaining a verification result; and

generating the text information based on the verification result, and feeding back the text information to the first user.

2. The method according to claim 1, wherein the at least one draft generation model comprises a draft synchronous generation model, and the method further comprises:

obtaining draft generation request information based on the text prompt information;

sending the draft generation request information to the draft synchronous generation model, wherein the draft synchronous generation model is configured to generate a draft based on the draft generation request information; and

storing, in the draft pool, the draft generated by the draft synchronous generation model.

3. The method according to claim 1 or 2, wherein the at least one draft generation model comprises a draft asynchronous generation model, and the method further comprises:

when inference load is less than a preset threshold, activating the draft asynchronous generation model to generate a draft; and

storing, in the draft pool, the draft generated by the draft asynchronous generation model.

4. The method according to any one of claims 1 to 3, wherein the draft pool further stores attribute information of the drafts.

5. The method according to claim 4, wherein the obtaining, from the draft pool, the target draft related to the text prompt information comprises:

obtaining, from the draft pool, the target draft related to the text prompt information based on the attribute information of the drafts stored in the draft pool.

6. The method according to claim 4 or 5, wherein the method further comprises:

retrieving and/or updating the drafts stored in the draft pool based on the attribute information of the drafts stored in the draft pool.

7. The method according to any one of claims 1 to 6, wherein the draft pool comprises a shared draft pool and a private draft pool of the first user; and

the obtaining, from the draft pool, the target draft related to the text prompt information comprises:

obtaining, from the shared draft pool and/or the private draft pool of the first user, the target draft related to the text prompt information.

8. A model inference system, wherein the system comprises:

a draft verification management unit, configured to determine at least one draft verification model from a draft verification model pool, wherein the draft verification model pool comprises a plurality of draft verification models;

a draft generation management unit, configured to determine, from a draft generation model pool, at least one draft generation model corresponding to the at least one draft verification model, wherein the draft generation model pool comprises a plurality of draft generation models; and

the at least one draft verification model, configured to receive text prompt information input by a first user, wherein the text prompt information is used to generate text information; wherein

the draft verification management unit is further configured to obtain, from a draft pool, a target draft related to the text prompt information, wherein the draft pool stores a plurality of drafts, and the plurality of drafts comprise a draft generated by the at least one draft generation model;

the at least one draft verification model is further configured to verify the target draft, and obtain a verification result; and

the at least one draft verification model is further configured to generate the text information based on the verification result, and feed back the text information to the first user.

9. The system according to claim 8, wherein the at least one draft generation model comprises a draft synchronous generation model;

the at least one draft verification model is further configured to obtain draft generation request information based on the text prompt information, and send the draft generation request information to the draft generation management unit through the draft verification management unit;

the draft generation management unit is further configured to send the draft generation request information to the draft synchronous generation model;

the draft synchronous generation model is configured to generate a draft based on the draft generation request information, and send the draft to the draft generation management unit; and

the draft generation management unit is further configured to store, in the draft pool, the draft generated by the draft synchronous generation model.

10. The system according to claim 8 or 9, wherein the at least one draft generation model comprises a draft asynchronous generation model;

the draft verification management unit is further configured to: when inference load is less than a preset threshold, activate the draft asynchronous generation model to generate a draft;

the draft asynchronous generation model is configured to generate a draft, and send the draft to the draft verification management unit; and

the draft verification management unit is further configured to store, in the draft pool, the draft generated by the draft asynchronous generation model.

11. The system according to any one of claims 8 to 10, wherein the draft pool further stores attribute information of the drafts.

12. The system according to claim 11, wherein the draft verification management unit is specifically configured to:

obtain, from the draft pool, the target draft related to the text prompt information based on the attribute information of the drafts stored in the draft pool.

13. The system according to claim 11 or 12, wherein the system further comprises:

a draft analysis unit, configured to retrieve and/or update the drafts stored in the draft pool based on the attribute information of the drafts stored in the draft pool.

14. The system according to any one of claims 8 to 13, wherein the draft pool comprises a shared draft pool and a private draft pool of the first user, and the draft verification management unit is specifically configured to:

obtain, from the shared draft pool and/or the private draft pool of the first user, the target draft related to the text prompt information.

15. A computer-readable storage medium, comprising computer program instructions, wherein when the computer program instructions are executed by a compute device cluster, the compute device cluster performs the method according to any one of claims 1 to 7.