MODEL TRAINING METHOD AND APPARATUS

[0001] This application claims priorities to Chinese Patent Application No. 202410299301.2, filed with the China National Intellectual Property Administration on March 14, 2024 and entitled "DATA PROCESSING METHOD, APPARATUS, COMPUTE DEVICE CLUSTER, AND STORAGE MEDIUM", and to Chinese Patent Application No. 202410798950.7, filed with the China National Intellectual Property Administration on June 18, 2024 and entitled "MODEL TRAINING METHOD AND APPARATUS", both of which are incorporated herein by reference in their entireties.

TECHNICAL FIELD

[0002] This application relates to the field of cloud computing technologies, and in particular, to a model training method and apparatus.

BACKGROUND

[0003] Model training refers to inputting training data to a model for processing to obtain a processing result. Based on a difference between the processing result and a label of the training data, model parameters are updated by using a backpropagation algorithm and an optimizer, so that the foregoing difference is gradually reduced, thereby obtaining a trained model. To prevent a case in which a fault of a training device in a training process causes a loss of previous training results (model parameters) and necessitates model re-training, a checkpoint (checkpoint) file is commonly saved to store the model parameters obtained in the training process of the model. Currently, data-parallel model training is often performed by using multiple groups of training devices, and parameters of models in the multiple groups of training devices remain consistent through parameter synchronization. Therefore, one group of training devices may be used to store the checkpoint file.

[0004] However, saving the checkpoint file by using one group of training devices leads to a long saving time due to a large quantity of model parameters, which in turn reduces model training efficiency.

SUMMARY

[0005] This application provides a model training method and apparatus, to resolve the

following problem that saving a checkpoint file by using one group of training devices leads to a long saving time due to a large quantity of model parameters leads to long saving time, thereby reducing low model training efficiency.

[0006] The following technical solutions are used in this application.

[0007] According to a first aspect, this application provides a model training method. The model training method may be applied to a computer system or a compute device that supports the computer system in implementing the model training method. The compute device may be a server, a terminal, or the like. In a possible example, the model training method is applied to a first worker node, the first worker node is deployed on a cloud computing platform (for example, the foregoing computer system), at least one second worker node is further deployed on the cloud computing platform, and the first worker node and the at least one second worker node use different training data to collaboratively train a model or a part of network layers in the model. The model training method includes: The first worker node receives a storage request, and splits, based on a first field that is in the storage request and that is used for splitting data used by a model or a part of network layers in the model, the data that is in the first worker node and that needs to be used by the model or the part of network layers in the model into N pieces of sub-data, to store A pieces of sub-data of the N pieces of sub-data as a checkpoint file. The first worker node obtains a part or all of B pieces of sub-data if data loss occurs in a process in which the first worker node trains the model, where $N \geq 2$, $1 \leq A \leq N$, a checkpoint file stored by the at least one second worker node includes the B pieces of sub-data of the N pieces of sub-data, and B is a difference between N and A.

[0008] In a possible case, the data used by the model includes a parameter and/or training data of the model.

[0009] In a possible case, the data used by the part of network layers in the model includes a parameter and/or training data of the part of network layers in the model.

[0010] In this application, the checkpoint file stored by the first worker node is only a part of the complete data used by the model, and the checkpoint files stored by the first worker node and the second worker node may be integrated to obtain the complete data used by the model. Therefore, when the checkpoint file stored by the first worker node is lost, the first worker node may obtain the checkpoint file stored by the second worker node, to resume training of the model or the part of network layers in the model, and avoid obtaining the complete parameter and/or training data of the model or the part of network layers in the model from a far storage node to resume training. This helps shorten time for resuming model training, and further improves model training efficiency.

[0011] For content of the checkpoint file stored by at least one check point file, three possible examples are provided below.

[0012] In a first possible example, the checkpoint file stored by the at least one second worker node includes the B pieces of sub-data.

[0013] In a second possible example, the checkpoint file stored by the at least one second worker node includes the N pieces of sub-data, and the N pieces of sub-data include the B pieces of sub-data.

[0014] In a third possible example, the checkpoint file stored by the at least one second worker node includes any quantity of pieces of sub-data from B to N, and the any quantity of pieces of sub-data from B to N includes the B pieces of sub-data.

[0015] In a possible case, the data loss indicates one or more of the following: a fault of the first worker node, and an interruption of the first worker node.

[0016] In a possible case, a value of the first field is N, and the first worker node and (N–1) second worker nodes use different training data to collaboratively train the model or the part of network layers in the model.

[0017] In a possible example, the first worker node and the (N–1) second worker nodes use different training data to collaboratively train the model. That is, the first worker node and the (N–1) second worker nodes are separately deployed with a complete model. Each time one or more rounds of training are performed, parameters of models in the first worker node and the (N–1) second worker nodes are synchronized.

[0018] In another possible example, the first worker node and the (N–1) worker nodes use different training data to collaboratively train the part of network layers in the model. That is, the first worker node and the (N–1) worker nodes are separately deployed with same one or more network layers in the model, and the first worker node and the (N–1) worker nodes use different training data to train the same one or more network layers. Each time one or more rounds of training are performed, parameters of network layers in the first worker node and the (N–1) second worker nodes are synchronized.

[0019] In a possible implementation, the cloud computing platform further includes a cloud management platform, and the cloud management platform is configured to manage the first worker node and the second worker node. The cloud management platform provides a storage request configuration interface, where the storage request configuration interface is configured to obtain configuration information entered by a user for data to be stored during model training, the configuration information is used to determine to split the data that is in the first worker node and that needs to be used by the model or the part of network layers in the model into the N pieces of sub-data, and the first field in the storage request is determined based on the configuration information. That the first worker node receives the storage request includes: The first worker node receives the storage request from the cloud management platform.

[0020] In this application, the cloud management platform receives the configuration information entered by the user, to store, based on a requirement of the user, the data used by the model or the part of network layers in the model, so as to ensure that when the data loss occurs in the model training process, the worker node can find a closest checkpoint file to resume model training. In this way, model training efficiency is improved, and the user customizes details (for example, the first field) of the checkpoint file stored by the first worker node.

[0021] In a possible case, the storage request configuration interface provided by the cloud management platform may be an application programming interface (application programming interface, API).

[0022] For example, the user may access, through an API provided by a compute device 111 and by using a terminal, a user interface (for example, a web page interface or a program interface) provided by the cloud management platform. The user interface includes various control components, for example, a control component corresponding to the configuration information.

[0023] In a possible case, the cloud management platform may also be referred to as a management node.

[0024] In a possible implementation, if the data used by the model includes the training data, the checkpoint file in the first worker node includes first sub-data and compressed second sub-data, where the first sub-data is the A pieces of sub-data, the second sub-data is training data used by the at least one second worker node in a process of training the model, and the B pieces of sub-data include the second sub-data. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model includes: The first worker node obtains the part of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model.

[0025] In this application, the first worker node does not compress training data used by the first worker node for model training, and compresses training data used by the at least one second worker node for model training, to ensure that the training data occupies small storage space. In addition, when the data loss occurs in the first worker node, the part of the B pieces of sub-data may be obtained, to avoid obtaining the training data from the far storage node to resume training. This helps shorten the time for resuming model training, and further improves model training efficiency.

[0026] In a possible example, the first worker node obtains the part of the B pieces of sub-data from the at least one second worker node.

[0027] In a possible example, the part of the B pieces of sub-data is sub-data required by the first worker node for model training, for example, the first sub-data.

[0028] In a possible implementation, if the data used by the model includes the parameter, the

checkpoint file in the first worker node includes the A pieces of sub-data, where $1 \leq A < N$. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model includes: The first worker node obtains the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model.

[0029] In this application, the first worker node stores only a part of the N pieces of sub-data, which reduces a data volume of the checkpoint file stored by the worker node, reduces a saving time, and improves model training efficiency.

[0030] In a possible implementation, that the first worker node stores the A pieces of sub-data of the N pieces of sub-data as the checkpoint file includes: The first worker node stores, locally in the first worker node, the checkpoint file including the A pieces of sub-data. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model includes: The first worker node obtains the part or all of the B pieces of sub-data from the at least one second worker node if the data loss occurs in the process in which the first worker node trains the model.

[0031] In this application, the first worker node locally stores the checkpoint file in the first worker node, which can improve efficiency of storing the checkpoint file, reduce a saving time, and improve model training efficiency. In addition, compared with obtaining the complete checkpoint file (including the data used by the model or the part of network layers in the model) from the storage node when the data loss occurs in the first worker node, the first worker node may obtain all or a part of the complete checkpoint file from the at least one second worker node. Because a data transmission rate between the first worker node and the second worker node is far greater than a data transmission rate between the first worker node and the storage node, efficiency of obtaining the complete checkpoint file to resume model training is accelerated.

[0032] In a possible implementation, the cloud computing platform further includes the storage node, and the model training method further includes: The first worker node sends, to the storage node, the checkpoint file including the A pieces of sub-data. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model includes: If the data loss occurs in the process in which the first worker node trains the model, the first worker node obtains the B pieces of sub-data from the at least one second worker node, and obtains the A pieces of sub-data from the storage node.

[0033] In this application, the first worker node sends the A pieces of sub-data to the storage node for persistent storage. This avoids a problem that model training in the first worker node cannot be resumed when the at least one second worker node does not store the sub-data required by the first worker node, and ensures that model training can be resumed.

[0034] In a possible case, the at least one second worker node sends, to the storage node, the checkpoint file including the B pieces of sub-data, to ensure that the storage node stores the complete checkpoint file.

[0035] In a possible implementation, the storage request further carries a status field indicating storage of the checkpoint file by the first worker node, the status field indicates a target network layer in the model, and the first worker node is configured to train the target network layer.

[0036] In this application, the first worker node stores the checkpoint file for the target network layer in the model based on the status field, thereby achieving storage of the checkpoint file for the specified network layer in the model. This avoids a problem that data loss of the target network layer requires re-training of the target network layer and results in low model training efficiency, thereby improving model training efficiency.

[0037] In a possible implementation, the storage request further carries a second field indicating storage frequency, and the storage frequency includes time frequency or training epoch frequency. That the first worker node stores the A pieces of sub-data of the N pieces of sub-data as the checkpoint file includes: if the time frequency or the training epoch frequency is satisfied in a process in which the first worker node trains the model or the part of network layers in the model, storing the A pieces of sub-data of the N pieces of sub-data as the checkpoint file.

[0038] In this application, the first worker node stores the checkpoint file based on the specified time frequency or training epoch frequency, to ensure that a time node or a training epoch node for storing the checkpoint file by the first worker node meets a requirement, and ensure that a process of storing the checkpoint file does not affect model training efficiency.

[0039] According to a second aspect, this application provides a model training apparatus. The model training apparatus is used in a computer system or a compute device that supports the computer system in implementing a model training method, and the model training apparatus includes modules configured to perform the model training method in the first aspect or any one of the optional implementations of the first aspect. In a possible example, the model training apparatus is used in a first worker node, the first worker node is deployed on a cloud computing platform, at least one second worker node is further deployed on the cloud computing platform, and the first worker node and the at least one second worker node use different training data to collaboratively train a model or a part of network layers in the model. For example, the model training apparatus includes a receiving module, a splitting module, a storage module, and an obtaining module. The receiving module is configured to receive a storage request, where the storage request carries a first field used for splitting data used by the model or the part of network layers in the model, and the data used by the model or the part of network layers in the model includes a parameter and/or training data of the model or the part of network layers in the model.

[0040] The splitting module is configured to split, based on the first field in the storage request, the data that needs to be used by the model or the part of network layers in the model into N pieces of sub-data, where $N \geq 2$.

[0041] The storage module is configured to store A pieces of sub-data of the N pieces of sub-data as a checkpoint file, where $1 \leq A \leq N$, a checkpoint file stored by the at least one second worker node includes B pieces of sub-data of the N pieces of sub-data, and B is a difference between N and A.

[0042] The obtaining module is configured to: obtain a part or all of the B pieces of sub-data if data loss occurs in a process of training the model.

[0043] In a possible implementation, a value of the first field is N, and the first worker node and (N–1) second worker nodes use different training data to collaboratively train the model or the part of network layers in the model.

[0044] In a possible implementation, the cloud computing platform further includes a cloud management platform, the cloud management platform is configured to manage the first worker node and the second worker node, the cloud management platform provides a storage request configuration interface, the storage request configuration interface is configured to obtain configuration information entered by a user for data to be stored during model training, the configuration information is used to determine to split the data that is in the first worker node and that needs to be used by the model into the N pieces of sub-data, the first field in the storage request is determined based on the configuration information, and the receiving module is specifically configured to receive the storage request from the cloud management platform.

[0045] In a possible implementation, if the data used by the model includes the training data, the checkpoint file in the first worker node includes first sub-data and compressed second sub-data, where the first sub-data is the A pieces of sub-data, the second sub-data is training data used by the at least one second worker node in a process of training the model, the B pieces of sub-data include the second sub-data, and the obtaining module is specifically configured to: obtain the part of the B pieces of sub-data if the data loss occurs in the process of training the model.

[0046] In a possible implementation, if the data used by the model includes the parameter, the checkpoint file in the first worker node includes the A pieces of sub-data, where $1 \leq A < N$, and the obtaining module is specifically configured to: obtain the B pieces of sub-data if the data loss occurs in the process of training the model.

[0047] In a possible implementation, the storage module is specifically configured to store, locally in the first worker node, the checkpoint file including the A pieces of sub-data, and the obtaining module is specifically configured to: obtain the part or all of the B pieces of sub-data from the at least one second worker node if the data loss occurs in the process of training the model.

[0048] In a possible implementation, the cloud computing platform further includes a storage node, and the apparatus further includes a sending module. The sending module is configured to send, to the storage node, the checkpoint file including the A pieces of sub-data. The obtaining module is specifically configured to: if the data loss occurs in the process of training the model, obtain the B pieces of sub-data from the at least one second worker node, and obtain the A pieces of sub-data from the storage node.

[0049] In a possible implementation, the storage request further carries a status field indicating storage of the checkpoint file by the first worker node, the status field indicates a target network layer in the model, and the first worker node is configured to train the target network layer.

[0050] In a possible implementation, the storage request further carries a second field indicating storage frequency, and the storage frequency includes time frequency or training epoch frequency. The storage module is specifically configured to: if the time frequency or the training epoch frequency is satisfied in a process of training the model or the part of network layers in the model, store the A pieces of sub-data of the N pieces of sub-data as the checkpoint file.

[0051] In a possible implementation, the data loss indicates one or more of the following: a fault of the first worker node, and an interruption of the first worker node.

[0052] For more detailed implementation content of the model training apparatus, refer to the descriptions of the first aspect and any one of the implementations of the first aspect and content of the following specific implementations. Details are not described herein again.

[0053] According to a third aspect, this application provides a chip, including a processor and a power supply circuit, where the power supply circuit is configured to supply power to the processor, and the processor is configured to perform the method in the first aspect or any one of the possible implementations of the first aspect.

[0054] According to a fourth aspect, this application provides a compute device cluster. The compute device cluster includes at least one compute device. The compute device includes a memory and a processor. The memory is configured to store computer instructions. When executing the computer instructions, the processor implements the method in the first aspect or any one of the possible implementations of the first aspect.

[0055] According to a fifth aspect, this application provides a computer-readable storage medium. The storage medium stores a computer program or instructions. When the computer program or the instructions are executed by a processing device, the method in the first aspect or any one of the possible implementations of the first aspect is implemented.

[0056] According to a sixth aspect, this application provides a computer program product. The computer program product includes a computer program or instructions. When the computer program or the instructions are executed by a processing device, the method in the first aspect or

any one of the possible implementations of the first aspect is implemented.

[0057] For beneficial effects of the second aspect to the sixth aspect, refer to the first aspect or any one of the possible implementations of the first aspect. Details are not described herein. In this application, based on the implementations provided in the foregoing aspects, the implementations may be further combined to provide more implementations.

BRIEF DESCRIPTION OF DRAWINGS

[0058] FIG. 1 is a diagram of a computer system according to this application;

[0059] FIG. 2 is a schematic flowchart of a model training method according to this application;

[0060] FIG. 3 is a diagram of a user interface according to this application;

[0061] FIG. 4 is a diagram 1 of storage performed by a worker node according to this application;

[0062] FIG. 5 is a diagram 2 of storage performed by a worker node according to this application;

[0063] FIG. 6 is a diagram 3 of storage performed by a worker node according to this application;

[0064] FIG. 7 is a diagram 4 of storage performed by a worker node according to this application;

[0065] FIG. 8a is a diagram 1 of multi-level storage of a checkpoint file according to this application;

[0066] FIG. 8b is a diagram 2 of multi-level storage of a checkpoint file according to this application;

[0067] FIG. 9 is a diagram 1 of a structure of a model training apparatus according to this application;

[0068] FIG. 10 is a diagram 2 of a structure of a model training apparatus according to this application;

[0069] FIG. 11 is a diagram of a structure of a compute device according to this application;

[0070] FIG. 12 is a diagram of a structure of a compute device cluster according to this application; and

[0071] FIG. 13 is a diagram of connection between compute devices according to this application.

DESCRIPTION OF EMBODIMENTS

[0072] Embodiments provide a model training method, to resolve the following problem that

saving a checkpoint file by using one group of training devices leads to a long saving time due to a large quantity of model parameters, thereby reducing model training efficiency. In a process in which a first worker node and a second worker node use different training data to collaboratively train a model or a part of network layers in the model, a checkpoint file stored by the first worker node is only a part of complete data used by the model or the part of network layers in the model, and checkpoint files stored by the first worker node and the second worker node are integrated to obtain the complete data used by the model or the part of network layers in the model. In this way, the first worker node stores only the part of the complete data used by the model, which reduces a volume of stored data, reduces a saving time, and improves model training efficiency. In addition, if data loss occurs in a process in which the first worker node trains the model, the first worker node may obtain a checkpoint file stored by the second worker node, to resume training of the model or the part of network layers in the model, and avoid obtaining a complete parameter and/or training data of the model or the part of network layers in the model from a far storage node to resume training. This helps shorten time for resuming model training, and further improves model training efficiency.

[0073] For ease of understanding, technical terms in this application are first described.

[0074] Artificial intelligence models, usually based on machine learning or deep learning technologies, are used to simulate some or all functions of human intelligence. By learning and training data, these models may perform various tasks, such as image recognition, speech recognition, natural language processing, prediction, and decision making.

[0075] Centralized training or distributed training is usually used for model training.

[0076] Distributed training: A model usually needs to undergo distributed training on a plurality of training devices (such as a server, a graphics processing unit (graphics processing unit, GPU), or a neural network processing unit (neural network processing unit, NPU)), to accelerate a training speed. In addition to having an advantage of accelerating the training speed, distributed training also allows remaining training devices to continue training when some of the training devices are faulty.

[0077] In a distributed training process, a distributed consistency algorithm is usually used to synchronize parameters of models in the plurality of training devices, to ensure parameter consistency. The distributed consistency algorithm may include distributed gradient descent, parameter server synchronization, or the like.

[0078] In the distributed training process, the following three parallel training modes are usually used.

[0079] Model parallelism: When a quantity of parameters of a model is large, the complete model cannot be trained on a single training device. Therefore, the complete model is split to obtain

a plurality of sub-models, and the plurality of sub-models are deployed in different training devices for training. In this way, a quantity of parameters of a sub-model trained on each training device is reduced, to implement training of the complete model, and accelerate a training speed of the complete model.

[0080] Pipeline parallelism: When a quantity of parameters of a model is large, the complete model cannot be trained on a single training device. Therefore, different network layers of the complete model are placed on different training devices for training, to reduce video memory consumption of a single training device, and implement training of the complete model.

[0081] Data parallelism: A training data set is split into a plurality of training data sub-sets, and the training data sub-sets are allocated to different training devices or training device groups, to train a model deployed in the training device or the training device group. The training device or the training device group is deployed with the complete model. That is, a training process is parallelized along a batch (batch) dimension.

[0082] A checkpoint file is a snapshot of a model stored in a training process, and may capture a status of the model, that is, a parameter and training data of the model. The parameter of the model includes a model weight, an optimizer status (such as a learning rate or momentum), a scheduler status (such as a historical learning rate, momentum, or weight decay), or an epoch hyperparameter (such as a training batch size or a quantity of training steps).

[0083] It should be noted that, in the distributed training process, training may be performed by using one or more of the foregoing parallel training modes.

[0084] The following describes in detail the model training method provided in this application with reference to the accompanying drawings.

[0085] First, FIG. 1 is a diagram of a computer system according to this application. As shown in FIG. 1, the computer system includes a management node 110 and a worker node cluster 120. The management node 110 and the worker node cluster 120 may communicate with each other in a wired manner, or may communicate with each other in a wireless manner.

[0086] In this application, the management node 110 provides a storage request configuration interface (for example, an API) externally. The storage request configuration interface is configured to obtain configuration information (namely, a training requirement) entered by a user for data to be stored during model training.

[0087] For detailed content of the configuration information, refer to the following description of the configuration information. Details are not described herein again.

[0088] In a possible example, the management node 110 may send, based on the received configuration information, a storage request to worker nodes included in the worker node cluster 120, to control the worker node cluster 120 to train a model. The storage request carries the content

of the configuration information.

[0089] The worker node cluster 120 is configured to collaboratively perform model training.

[0090] The worker node cluster 120 may include a plurality of worker nodes. For example, the plurality of worker nodes include a worker node 121, a worker node 122, a worker node 123, and a worker node 124. The worker node may also be referred to as a compute device or a training device, and the compute device or the training device may be configured to perform model training. The management node 110 may also be referred to as a compute device, a control device, or a cloud management platform. The compute device or the control device may be configured to: interact with the user, and control, based on the training requirement of the user, the worker node cluster 120 to train the model.

[0091] In a possible example, the worker node 121, the worker node 122, the worker node 123, and the worker node 124 use different training data to collaboratively train the model or a part of network layers in the model.

[0092] In a possible case, the worker node may include one or more acceleration cards. The accelerator card may be one or more of the following: a GPU, an NPU, a field programmable gate array (field programmable gate array, FPGA), an application processor (application processor, AP), a tensor processing unit (tensor processing unit, TPU), a controller, an application-specific integrated circuit (Application-Specific Integrated Circuit, ASIC), a discrete gate, a transistor logic device, a discrete hardware component, and the like.

[0093] The management node may include one or more processing units. The processing unit may be a GPU, an NPU, an FPGA, an AP, a modem processor, a GPU, an image signal processor (image signal processor, ISP), a controller, a video codec, a digital signal processor (digital signal processor, DSP), an ASIC, an FPGA, a discrete gate, a transistor logic device, a discrete hardware component, a baseband processor, an NPU, and/or the like, or may be a processing device including any one or more of the foregoing processing units, such as a terminal or a server.

[0094] The foregoing wired communication manner may be the Ethernet, an optical fiber, and various peripheral component interconnect express (peripheral component interconnect express, PCIe) buses, universal serial buses (universal serial buses, USBs) or unified buses (unified buses, Ubuses or UBs), compute express links (compute express links, CXLs), cache coherent interconnect for accelerators (cache coherent interconnect for accelerators, CCIX), and the like that are disposed inside the computer system and configured to connect the management node 110 and the worker node cluster 120.

[0095] The foregoing wireless communication manner may be the Internet, wireless communication (wireless fidelity, Wi-Fi), an ultra-wideband (ultra-wideband, UWB) technology, and the like.

[0096] In a possible case, a cloud computing platform may include the management node 110 and the worker node cluster 120. The cloud computing platform is configured to train the model, and the management node 110 receives the configuration information from the user for the data to be stored during model training. The configuration information may be a quantity of worker nodes that perform model training this time, a quantity of worker node groups, a storage frequency, a storage location, and the like, so that the management node 110 controls, based on the configuration information, the worker node cluster 120 to train the model, and stores a checkpoint file in a training process.

[0097] In a possible example, the computer system may further include a terminal 130, a terminal 140, and a terminal 150.

[0098] The terminal 130, the terminal 140, and the terminal 150 each may be a mobile phone, a tablet computer, a handheld computer, a personal computer (personal computer, PC), a cellular phone, a personal digital assistant (personal digital assistant, PDA), a wearable device (for example, a smartwatch), a smart home device (for example, a television set), a vehicle-mounted computer, a game console, an augmented reality (augmented reality, AR)/virtual reality (virtual reality, VR) device, or the like. Specific device forms of the terminal 130, the terminal 140, and the terminal 150 are not specially limited in this application.

[0099] In another possible example, the terminal 130, the terminal 140, and the terminal 150 are independently disposed, and are communicatively connected to the computer system, namely, the cloud computing platform.

[00100] It should be noted that an architecture of the computer system shown in FIG. 1 is merely an example, and a device type or a quantity of devices in the system may be configured according to an actual requirement. This is not limited in embodiments of this application. For example, the computer system may further include more management nodes or worker nodes.

[00101] In a possible embodiment, the computer system may further include a storage node 160, and the storage node 160 is managed by the management node 110. The storage node 160 may include one or more storage devices, for example, a mechanical hard disk (hard disk drive, HDD), a solid-state disk (solid-state drive, SSD), an optical disc, a memory card, and a magnetic tape. The storage node 160 may be configured to store a model parameter generated by the worker node in the model training process, training data, and/or the like. A quantity of storage devices included in the storage node 160 may be 10 or 30. This is not limited in this application.

[00102] In a possible case, the storage node 160 may store the checkpoint file sent by the worker node, and all checkpoint files stored in the storage node 160 are integrated to include all data used by the model.

[00103] The following describes in detail implementations of the model training method

provided in embodiments of this application with reference to the accompanying drawings.

[00104] FIG. 2 is a schematic flowchart of a model training method according to this application. The method provided in this embodiment may be applied to the computer system shown in FIG. 1. Herein, an example in which the model training method in embodiments of this application is performed by a first worker node 210 is used for description. The first worker node 210 may be any one of the plurality of worker nodes shown in FIG. 1, and worker nodes, other than the first worker node 210, in the plurality of worker nodes may be referred to as second worker nodes. In the following, the worker node 121 is the first worker node 210, and the worker node 122, the worker node 123, and the worker node 124 may all be referred to as second worker nodes. The first worker node 210 and the second worker node use different training data to collaboratively train a model or a part of network layers in the model.

[00105] In a possible scenario, the management node 110 in FIG. 1 may be configured to manage the first worker node 210 and the second worker node, and the management node 110 provides an API externally. A user may access, through the API provided by the management node 110 and by using a terminal 130, a user interface (for example, a web page interface or a program interface) provided by a cloud management platform. The user interface includes various control components, for example, a control component corresponding to configuration information. The management node 110 sends, to the first worker node 210 based on the received configuration information, a storage request determined based on the configuration information, so that the first worker node stores a checkpoint file in a model training process, and resumes model training based on the checkpoint file after data loss occurs.

[00106] In a second possible scenario, the user may directly send the storage request to the first worker node 210 and at least one second worker node based on APIs provided by the first worker node 210 and the at least one second worker node. For example, the user may send the storage request to the first worker node 210 and the at least one second worker node on the terminal through a command line interface.

[00107] The foregoing two possible application scenarios are merely examples provided in this embodiment, and should not be construed as a limitation on this application.

[00108] It should be noted that the model training method provided in this application may be applied to a public cloud, a private cloud, or a hybrid cloud.

[00109] For example, the model training method provided in this application is applied to a public cloud. To be specific, a cloud service vendor provides all of the management node and the worker nodes shown in FIG. 1 and software services on the management node and the worker nodes. The user only needs to purchase a corresponding node from the cloud service vendor for use.

[00110] For another example, the model training method provided in this application is applied to a private cloud. To be specific, the user purchases all of the management node and the worker nodes shown in FIG. 1 and establishes a cloud service.

[00111] For still another example, the model training method provided in this application is applied to a hybrid cloud. To be specific, the management node shown in FIG. 1 is located on a public cloud, and the management node is configured to manage a worker node located on a private cloud.

[00112] Still refer to FIG. 2. The method includes the following steps S210 to S240.

[00113] S210: The first worker node 210 receives the storage request.

[00114] The storage request carries a first field used for splitting data used by the model or the part of network layers in the model.

[00115] In a possible case, the data used by the model includes a parameter and/or training data of the model.

[00116] In a possible case, the data used by the part of network layers in the model includes a parameter and/or training data of the part of network layers in the model.

[00117] In a possible example, a value of the first field is N, and N is an integer greater than or equal to 2.

[00118] The first worker node and (N–1) second worker nodes use different training data to collaboratively train the model or the part of network layers in the model.

[00119] For example, a first network layer in the model is deployed in all of the first worker node and the (N–1) second worker nodes, so that the first worker node and the (N–1) second worker nodes use different training data to train the first network layer.

[00120] For another example, the complete model is deployed in the first worker node and the (N–1) second worker nodes, so that the first worker node and the (N–1) second worker nodes use different training data to train the model.

[00121] The following provides three possible cases for the data used by the model.

[00122] In a first possible case, the data used by the model includes the parameter of the model. The parameter of the model includes a weight of each network layer included in the model, an optimizer status, a scheduler status, an epoch hyperparameter, or the like.

[00123] In a second possible case, the data used by the model includes the training data. The training data is voice data, text data, image data, or the like.

[00124] In a third possible case, the data used by the model includes the parameter and the training data of the model. For content of this case, refer to the descriptions of the foregoing two cases. Details are not described herein again.

[00125] For detailed content of the data used by the part of network layers in the model, refer

to the foregoing descriptions of the data used by the model. Details are not described herein again.

[00126] In a possible implementation, that the first worker node 210 receives the storage request includes: The first worker node 210 receives the storage request from the management node 110.

[00127] For example, the management node 110 receives the configuration information from the user, and then sends, to the first worker node 210 through the Ethernet, an optical fiber, or a bus, the storage request including the configuration information. The first worker node 210 receives the storage request sent by the management node 110.

[00128] The configuration information includes the first field.

[00129] For content of receiving the configuration information from the user by the management node 110, the following provides a possible embodiment. FIG. 3 is a diagram of a user interface according to this application. Content shown in FIG. 3 may include the following steps S310 and S320.

[00130] S310: The management node 110 receives a trigger operation performed by the user on a control component on the user interface.

[00131] The user interface includes control components for the configuration information (such as a parameter 1 and a parameter 2), for example, a control component corresponding to the first field (such as the parameter 1).

[00132] As shown in FIG. 3, the control components for the configuration information on the user interface may be in a plurality of forms, for example, in a square shape or a switch shape. This is not limited in this application. For example, the control component corresponding to the parameter 1 is configured to indicate to fill in the value of the first field, for example, N.

[00133] In a possible example, the management node 110 may display the user interface at a front end. The front end herein may be a display connected to the management node 110, or a compute device (for example, the terminal 130) connected to the management node 110, where the compute device has a display or the like. This is not limited in this application.

[00134] For example, the user accesses a storage request configuration interface provided by the management node 110, to display, on the terminal 130, the user interface provided by the management node 110, so that the management node 110 obtains, through the storage configuration interface, the configuration information entered by the user on the user interface for data to be stored during model training.

[00135] For example, the configuration information is used to determine to split the data that is in the first worker node and that needs to be used by the model into N pieces of sub-data. That is, the configuration information includes the first field, so that the first field can be determined from the configuration information.

[00136] In a possible implementation, that the management node 110 receives the trigger

operation performed by the user on the control component includes: The management node 110 obtains trigger operations performed by the user on the control component on the user interface by using various input devices (such as a keyboard, a mouse, and a touchscreen).

[00137] For specific implementations of the trigger operation, the following provides three possible examples.

[00138] Example 1: The trigger operation may be confirmation of the control component for the configuration information by the user by using the keyboard. For example, the trigger operation is an enter (enter) key triggered by the user.

[00139] Example 2: The trigger operation may be a click performed by the user on the control component for the configuration information by using the mouse.

[00140] Example 3: The trigger operation may be an input operation performed by the user on the control component for the configuration information by using the keyboard, for example, inputting a corresponding value.

[00141] The foregoing examples are merely optional implementations provided in this embodiment, and should not be understood as a limitation on this application. In another embodiment of this application, the trigger operation may alternatively be an air gesture operation, voice control, knuckle tapping, or the like.

[00142] S320: The management node 110 obtains the configuration information in response to the trigger operation performed by the user on the control component.

[00143] The management node 110 obtains the configuration information in response to the trigger operations performed by the user on the control component by using various input devices.

[00144] In this application, the management node 110 interacts with the user through the user interface, and obtains the configuration information to be configured by the user, so that visualization is implemented, and the checkpoint file is stored according to a user requirement, to improve a degree of processing freedom.

[00145] In a possible embodiment, the storage request further carries a status field indicating the first worker node 210 to store the checkpoint file. The status field indicates a target network layer in the model, and the first worker node 210 is configured to train the target network layer.

[00146] The status field may alternatively be included in the configuration information, and may also be obtained by using the method shown in FIG. 3. Details are not described in this application.

[00147] The status field may be an identifier of the network layer, and the identifier may be expressed in a form of character, such as a letter, a value, or a punctuation mark.

[00148] The target network layer may be any network layer included in the model, for example, an embedded layer, a fully connected layer, a convolutional layer, or a normalization layer.

[00149] In a possible case, after the status field identifies the target network layer in the model, in a process in which the first worker node and the second worker node collaboratively perform model training, the first worker node 210 and the second worker node store checkpoint files once before or after the target network layer is trained.

[00150] For example, in the process in which the first worker node 210 and the second worker node collaboratively perform model training, the first worker node 210 stores the checkpoint file once after the first worker node 210 trains a convolutional layer a included in the model.

[00151] When the model includes P network layers, the convolutional layer a is any one of a 1^{st} network layer to a $(P-1)^{th}$ network layer in the model.

[00152] In the process in which the first worker node 210 and the second worker node collaboratively perform model training, the first worker node 210 stores the checkpoint file once after the first worker node trains a convolutional layer b included in the model. The convolutional layer b is the P^{th} network layer. That is, the checkpoint file is stored only after a round of training is performed on each layer in the model.

[00153] In a possible case, the status field further indicates steps in data preprocessing, for example, data denoising, missing value filling, duplicate value removal, and standardization.

[00154] In a possible embodiment, the storage request further carries a second field indicating a storage frequency.

[00155] The second field may alternatively be included in the configuration information, and may also be obtained by using the method shown in FIG. 3. Details are not described in this application.

[00156] The second field is a value, for example, 30 or 10. This is not limited in this application.

[00157] In a possible case, the storage frequency indicates a time frequency or a training epoch frequency. If the time frequency or the training epoch frequency is satisfied in a process in which the first worker node 210 trains the model or the part of network layers in the model, A pieces of sub-data of the N pieces of sub-data are stored as the checkpoint file.

[00158] For example, when the storage frequency indicates the time frequency, and the second field is 30, it indicates that in the process in which the first worker node 210 and the second worker node collaboratively perform model training, the first worker node 210 and the second worker node store the checkpoint files once every 30 minutes.

[00159] When the storage frequency indicates the training epoch frequency, and the second field is 10, it indicates that in the process in which the first worker node 210 and the second worker node collaboratively perform model training, the first worker node 210 and the second worker node store the checkpoint files once every 10 rounds of training.

[00160] It should be noted that each time the first worker node 210 and the second worker node

store the checkpoint files, previously stored checkpoint files are overwritten.

[00161] In a possible embodiment, the storage request further carries the status field and the second field. For content of this embodiment, refer to the content of the foregoing embodiment. Details are not described herein again.

[00162] For content of storing the checkpoint file by the first worker node 210, refer to the following description of S230. Details are not described herein again.

[00163] In a possible embodiment, the storage request further carries a specified field. The specified field indicates to store only a model parameter in a worker node that trains a specific network layer, that is, store only a parameter of a feature network layer in the model.

[00164] In another possible implementation, the first worker node 210 receives the storage request sent by the terminal 130. That is, the user directly performs information configuration on the first worker node 210 and the second worker node.

[00165] For example, the user may directly send the storage request to the first worker node 210 and the at least one second worker node based on the APIs provided by the first worker node 210 and the at least one second worker node. For example, the user may send the storage request to the first worker node 210 and the at least one second worker node on the terminal through the command line interface.

[00166] For content of the storage request, refer to the content of the storage request described above in reception of the storage request by the first worker node 210 from the management node 110. Details are not described herein again.

[00167] Still refer to FIG. 2. The model training method provided in this embodiment further includes step S220.

[00168] S220: The first worker node 210 splits, based on the first field in the storage request, the data that is in the first worker node 210 and that needs to be used by the model or the part of network layers in the model into the N pieces of sub-data.

[00169] N is an integer greater than or equal to 2.

[00170] In a possible example, the value of the first field is N, for example, 4, and four pieces of sub-data are sub-data 1 including a weight 1, sub-data 2 including a weight 2, sub-data 3 including a weight 3, and sub-data 4 including a weight 4.

[00171] For content of S220, the following provides two possible implementations.

[00172] In a first possible implementation, the first worker node 210 evenly splits, based on the first field in the storage request, the data that is in the first worker node and that is used by the model into the N pieces of sub-data.

[00173] For example, the first worker node 210 evenly splits, based on a data volume of the data used by the model, the data into the N pieces of sub-data.

[00174] In a second possible implementation, the first worker node 210 evenly splits, based on the first field in the storage request and a type of the data used by the model, the data into the N pieces of sub-data.

[00175] For example, because the data used by the model includes a model weight, the optimizer status, the scheduler status, and the epoch hyperparameter, and the first field is 4, the first worker node 210 may split the data used by the model into four pieces of sub-data: the model weight, the optimizer status, the scheduler status, and the epoch hyperparameter.

[00176] It should be noted that the foregoing two possible implementations are merely possible examples provided in this application for splitting the data used by the model. In another possible implementation of this application, the first worker node 210 only needs to split the data used by the model into the N pieces of sub-data. A specific splitting manner is not limited in this application.

[00177] S230: The first worker node 210 stores the A pieces of sub-data of the N pieces of sub-data as the checkpoint file.

[00178] A is an integer greater than or equal to 1, and A is less than or equal to N.

[00179] For example, the first worker node 210 stores the A pieces of sub-data of the N pieces of sub-data as the checkpoint file locally in the first worker node 210, that is, in a memory included in the first worker node 210.

[00180] In a possible case, S210 and S220 may alternatively be performed by the second worker node, so that the second worker node stores B pieces of sub-data of the N pieces of sub-data as a checkpoint file, where B is a difference between N and A. In other words, the checkpoint file stored by the second worker node includes the B pieces of sub-data of the N pieces of sub-data. For example, the first worker node 210 stores the sub-data 4 as the checkpoint file.

[00181] It should be noted that, before the first worker node 210 stores the checkpoint file, the first worker node 210 and the second worker node perform parameter synchronization based on a distributed consistency algorithm. In other words, before the first worker node 210 stores the checkpoint file, parameters of models in the first worker node 210 and the second worker node are consistent.

[00182] S240: The first worker node 210 obtains a part or all of the B pieces of sub-data if data loss occurs in the process in which the first worker node 210 trains the model.

[00183] For example, if the data used by the model includes the parameter of the model, the first worker node 210 obtains all of the B pieces of sub-data.

[00184] For example, if the data used by the model includes the training data, the first worker node 210 obtains the part of the B pieces of sub-data.

[00185] For example, the part of the B pieces of sub-data is training data used by the first worker node 210 to train the model.

[00186] In a possible case, the data loss may be loss of the data used by the model or the part of network layers in the model.

[00187] For the data loss, the following provides three possible cases.

[00188] In a first possible case, the data loss indicates a fault of the first worker node 210.

[00189] For example, the fault of the first worker node 210 may be one or more of the following: a fault of a storage medium (such as an HDD or an SSD), a communication delay, a software crash or error in the first worker node 210, overload, and the like.

[00190] In a second possible case, the data loss indicates an interruption of the first worker node.

[00191] For example, the fault of the first worker node 210 may be one or more of the following: a communication interruption, a power-off interruption, a computer crash, and the like.

[00192] In a third possible case, the data loss indicates a fault and an interruption of the first worker node 210.

[00193] For content of this example, refer to the content of the first possible case and the second possible case. Details are not described herein again.

[00194] For detailed content of S240, refer to the following two possible cases. Details are not described herein again.

[00195] For content of S230, the following provides three possible implementations.

[00196] In a possible implementation, if the data used by the model includes the parameter of the model, the first worker node 210 stores the A pieces of sub-data of the N pieces of sub-data as the checkpoint file, where $1 \leq A < N$.

[00197] If A is 1, and the second worker nodes include the worker node 122, the worker node 123, and the worker node 124, the following provides a possible example. FIG. 4 is a diagram 1 of storage performed by a worker node according to this application.

[00198] The first worker node 210 splits the data used by the model into four pieces of sub-data, for example, A1-1, A1-2, A1-3, and A1-4, and the first worker node 210 stores A1-1 as a checkpoint file.

[00199] In addition, the worker node 122 stores A1-2 as a checkpoint file, the worker node 123 stores A1-3 as a checkpoint file, and the worker node 124 stores A1-4 as a checkpoint file.

[00200] If A is 2, and the second worker nodes include the worker node 122, the worker node 123, and the worker node 124, the following provides another possible example. FIG. 5 is a diagram 2 of storage performed by a worker node according to this application.

[00201] The first worker node 210 splits the data used by the model into four pieces of sub-data, for example, B1-1, B1-2, B1-3, and B1-4, and the first worker node 210 stores B1-1 and B1-2 as a checkpoint file.

[00202] In addition, the worker node 122 stores B1-2 and B1-3 as a checkpoint file, the worker

node 123 stores B1-3 and B1-4 as a checkpoint file, and the worker node 124 stores B1-4 and B1-1 as a checkpoint file.

[00203] In other words, each worker node stores two pieces of sub-data, so that the first worker node 210 can obtain complete model parameters only from the second worker nodes when performing fault recovery, to avoid obtaining the model parameters from a far storage node 160. This shortens time for obtaining the model parameters, and improves efficiency of resuming model training by the first worker node 210.

[00204] It should be noted that, a checkpoint file stored by one worker node may further include more sub-data. For content that the checkpoint file may further include more sub-data, refer to the foregoing content that the checkpoint file includes two pieces of sub-data. Details are not described herein again.

[00205] It should be noted that for content of a case in which the data used by the part of network layers in the model includes the parameter of the part of network layers in the model, refer to the foregoing implementations. Details are not described herein again.

[00206] In another possible implementation, if the data used by the model includes the training data, and the first worker node 210 and the second worker node are two worker nodes in data-parallel training, that is, the first worker node 210 and the second worker node use different training data to train the model, the first worker node 210 stores first sub-data and compressed second sub-data as the checkpoint file.

[00207] The first sub-data is first training data used by the first worker node 210 in the process of training the model, the second sub-data is second training data used by the second worker node in a process of training the model, and the B pieces of sub-data stored by the second worker node include the second sub-data.

[00208] The N pieces of sub-data are equal to the A pieces of sub-data, and the A pieces of sub-data include the first sub-data and the second sub-data.

[00209] If A is 4, and the second worker nodes include the worker node 122, the worker node 123, and the worker node 124, the following provides another possible example. FIG. 6 is a diagram 3 of storage performed by a worker node according to this application.

[00210] The first worker node 210 splits the training data into four pieces of sub-data, for example, C1-1, C1-2, C1-3, and C1-4. Because only C1-1 is first training sub-data used by the first worker node 210, the first worker node 210 compresses C1-2, C1-3, and C1-4 to obtain compressed C1-2, compressed C1-3, and compressed C1-4. Then, the first worker node 210 stores C1-1, the compressed C1-2, the compressed C1-3, and the compressed C1-4 as a checkpoint file.

[00211] In addition, because only C1-2 is second training sub-data used by the worker node 122, the worker node 122 compresses C1-1, C1-3, and C1-4 to obtain compressed C1-1, the

compressed C1-3, and the compressed C1-4. Then, the worker node 122 stores C1-2, the compressed C1-1, the compressed C1-3, and the compressed C1-4 as a checkpoint file.

[00212] Because only C1-3 is second training sub-data used by the worker node 123, the worker node 123 compresses C1-1, C1-2, and C1-4 to obtain the compressed C1-1, the compressed C1-2, and the compressed C1-4. Then, the worker node 123 stores C1-3, the compressed C1-1, the compressed C1-2, and the compressed C1-4 as a checkpoint file.

[00213] Because only C1-4 is second training sub-data used by the worker node 124, the worker node 122 compresses C1-1, C1-3, and C1-3 to obtain the compressed C1-1, the compressed C1-2, and the compressed C1-3. Then, the worker node 124 stores C1-4, the compressed C1-1, the compressed C1-2, and the compressed C1-3 as a checkpoint file.

[00214] It should be noted that, in this implementation, the first worker node 210 and the second worker node are two worker nodes in data-parallel training, and the first worker node 210 and the second worker node may be configured to train a 1st network layer of the model, or may be configured to perform data preprocessing.

[00215] It should be noted that for content of a case in which the data used by the part of network layers in the model includes the training data of the part of network layers in the model, refer to the foregoing implementations. Details are not described herein again.

[00216] In still another possible implementation, if the data that needs to be used by the model includes the parameter and the training data of the model, for content of this implementation, refer to the descriptions of the foregoing two possible implementations. Details are not described herein again.

[00217] In a possible embodiment, the first worker node 210 and the second worker node are configured to train a same network layer in the model.

[00218] For example, when the model has eight network layers, the first worker node 210 and the second worker node are configured to train any one of the eight network layers. The following uses an example in which both the first worker node 210 and the second worker node train a network layer A of the eight network layers for description.

[00219] If M is 1, and the second worker nodes include the worker node 122, the worker node 123, and the worker node 124. FIG. 7 is a diagram 4 of storage performed by a worker node according to this application.

[00220] The first worker node 210 splits data used by the network layer D1 into four pieces of sub-data, for example, D1-1, D1-2, D1-3, and D1-4, and the first worker node 210 stores D1-1 as a checkpoint file. The data used by the network layer D includes a weight of the network layer D, and the like.

[00221] In addition, the worker node 122 stores D1-2 as a checkpoint file, the worker node 123

stores D1-3 as a checkpoint file, and the worker node 124 stores D1-4 as a checkpoint file.

[00222] It should be noted that data used by network layers D1 in the first worker node 210, the worker node 122, the worker node 123, and the worker node 124 is consistent.

[00223] In addition, for network layers, other than the network layer D1, in the eight network layers, data used by the network layers is stored in a manner the same as that of the first worker node 210 or the second worker node.

[00224] For example, a worker node 711, a worker node 712, a worker node 713, and a worker node 714 are configured to train a same network layer in the model, such as a network layer D2. Similarly, the worker node 711, the worker node 712, the worker node 713, and the worker node 714 all split the data used by the network layer D2 into four pieces of sub-data, for example, D2-1, D2-2, D2-3, and D2-4. The worker node 711 stores D2-1 as a checkpoint file, the worker node 712 stores D2-2 as a checkpoint file, the worker node 713 stores D2-3 as a checkpoint file, and the worker node 714 stores D2-4 as a checkpoint file.

[00225] Similarly, a worker node 715 stores D3-1 as a checkpoint file, a worker node 716 stores D3-2 as a checkpoint file, a worker node 717 stores D3-3 as a checkpoint file, and a worker node 718 stores D3-4 as a checkpoint file.

[00226] If M is 2 or more, for an example of storing sub-data by a worker node, refer to the content shown in FIG. 5 or FIG. 6. Details are not described herein again.

[00227] In a possible embodiment, the cloud computing platform shown in FIG. 1 further includes the storage node 160. After the first worker node 210 shown in FIG. 2 stores the checkpoint file, the model training method further includes: The first worker node 210 sends, to the storage node 160, the checkpoint file including the A pieces of sub-data.

[00228] FIG. 8a is a diagram 1 of multi-level storage of a checkpoint file according to this application. As shown in FIG. 8a, after storing the checkpoint file, the first worker node 210 further sends the checkpoint file to the storage node 160 for persistent storage.

[00229] In a possible case, the second worker node also sends, to the storage node 160, the checkpoint file including the B pieces of sub-data.

[00230] For example, the first worker node 210 sends, to the storage node 160 for persistent storage, A1-1 stored by the first worker node 210, the worker node 122 sends, to the storage node 160 for persistent storage, A1-2 stored by the worker node 122, the worker node 123 sends, to the storage node 160 for persistent storage, A1-3 stored by the worker node 123, and the worker node 124 sends, to the storage node 160 for persistent storage, A1-4 stored by the worker node 124.

[00231] In a possible case, FIG. 8b is a diagram 2 of multi-level storage of a checkpoint file according to this application. As shown in a in FIG. 8b, the first worker node 210 may send a checkpoint file to the storage node 160 in a communication gap or a computing gap after an n^{th}

round of training.

[00232] As shown in b in FIG. 8b, the first worker node 210 may send a checkpoint file to the storage node 160 in a process of performing an $(n+1)^{th}$ round of training (for example, in a communication gap or a computing gap in one round of training).

[00233] Further, after receiving a checkpoint file sent by each worker node, the storage node 160 performs persistent storage.

[00234] For content of S240, the following provides two possible cases.

[00235] In a first possible case, the data used by the model includes the training data. That the first worker node 210 obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node 210 trains the model includes: The first worker node 210 obtains the part of the B pieces of sub-data if the data loss occurs in the process in which the first worker node 210 trains the model.

[00236] In this case, both the first worker node 210 and the second worker node store the complete training data, and the complete training data includes compressed sub-data and uncompressed sub-data. The uncompressed sub-data is training data required when a worker node a trains the model. The compressed sub-data is training data required when a worker node, other than the worker node a, in a plurality of worker nodes trains the model, where the worker node may be any one of the first worker node 210 and the second worker node.

[00237] As shown in FIG. 6, the first worker node 210 stores C1-1, the compressed C1-2, the compressed C1-3, and the compressed C1-4 as the checkpoint file, and the worker node 122 stores C1-2, the compressed C1-1, the compressed C1-3, and the compressed C1-4 as the checkpoint file. Therefore, when the data loss occurs in the first worker node 210, the first worker node 210 may obtain the compressed C1-1 (training data that needs to be used by the first worker node 210 to train the model) locally from the worker node 122.

[00238] The first worker node 210 decompresses the compressed C1-1, to resume model training by using C1-1.

[00239] In a possible example, the first worker node 210 may obtain the complete checkpoint file in the worker node 122, to be specific, C1-2, the compressed C1-1, the compressed C1-3, and the compressed C1-4; and after resuming model training by using C1-1, compress C1-2 to obtain the compressed C1-2. Further, the first worker node 210 locally stores C1-1, the compressed C1-2, the compressed C1-3, and the compressed C1-4 as a checkpoint file again.

[00240] In a second possible scenario, the data used by the model includes the parameter of the model. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node 210 trains the model includes: The first worker node obtains the B pieces of sub-data if the data loss occurs in the process in which the first worker

node trains the model.

[00241] In a possible example, if the checkpoint file locally stored by the second worker node includes the N pieces of sub-data, the first worker node 210 may obtain the complete N pieces of sub-data from the second worker node, to resume model training.

[00242] For example, as shown in FIG. 5, when A is 2, the checkpoint file stored by the first worker node 210 includes B1-1 and B1-2, the checkpoint file stored by the worker node 122 includes B1-2 and B1-3, the checkpoint file stored by the worker node 123 includes B1-3 and B1-4, and the checkpoint file stored by the worker node 124 includes B1-4 and B1-1.

[00243] In this way, when the data loss occurs in the first worker node 210, the first worker node 210 can obtain B1-2 and B1-3 from the worker node 122, and obtain B1-4 and B1-1 from the worker node 124, so that the model resumes model training by using complete B1.

[00244] It should be noted that the data loss not only means loss of the stored checkpoint file, but also includes loss of a model parameter stored in a cache.

[00245] In another possible example, if the checkpoint file locally stored by the second worker node does not include the N pieces of sub-data, the first worker node 210 may obtain the B pieces of sub-data from the second worker node, and obtain the A pieces of sub-data from the storage node 160, that is, obtain, from the storage node 160, sub-data, of the N pieces of sub-data, other than the sub-data in the checkpoint file stored by the second worker node.

[00246] As shown in FIG. 4, when A is 1, the first worker node 210 stores A1-1 as the checkpoint file, the worker node 122 stores A1-2 as the checkpoint file, the worker node 123 stores A1-3 as the checkpoint file, the worker node 124 stores A1-4 as the checkpoint file, and the storage node 160 stores complete A1.

[00247] In this way, to quickly resume model training, the first worker node 210 can obtain A1-2 from the worker node 122, obtain A1-3 from the worker node 123, and obtain A1-4 from the worker node 124. In addition, because A1-1 is still absent from complete model parameters, and the storage node 160 stores the complete N pieces of data, the first worker node 210 may obtain A1-1 from the storage node 160, so that the first worker node 210 resumes model training by using the obtained complete A1 (A1-1, A1-2, A1-3, and A1-4).

[00248] For example, the data used by the model includes the parameter of the model. That the first worker node obtains the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node 210 trains the model includes: The first worker node obtains the part of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model.

[00249] As shown in FIG. 4, after the data loss occurs in the first worker node 210, A1-2 may be obtained from the worker node 122, and A1-3 may be obtained from the worker node 123. In

addition, because A1-1 and A1-4 are still absent from the complete model parameters, and the storage node 160 stores the complete N pieces of data, the first worker node 210 may obtain A1-1 and A1-4 from the storage node 160, so that the first worker node 210 resumes model training by using the obtained complete A1 (A1-1, A1-2, A1-3, and A1-4). It should be noted that for content of a case in which the data used by the part of network layers in the model includes the parameter and/or the training data, refer to the descriptions of the foregoing two possible cases. Details are not described herein again.

[00250] It may be understood that, to implement functions in the foregoing embodiments, the first worker node 210 includes corresponding hardware structures and/or software modules for performing the functions. A person skilled in the art should be easily aware that, in this application, the units and method steps in the examples described with reference to embodiments disclosed in this application can be implemented by hardware or a combination of hardware and computer software. Whether a function is performed by hardware or hardware driven by computer software depends on particular application scenarios and design constraint conditions of the technical solutions.

[00251] The foregoing describes in detail the model training method provided in this application with reference to FIG. 2 to FIG. 8b. The following describes a model training apparatus provided in this application with reference to FIG. 9. FIG. 9 is a diagram 1 of a structure of a model training apparatus according to this application. The model training apparatus 900 may be configured to implement functions of the first worker node 210 in the foregoing method embodiments, and therefore can also implement beneficial effects of the foregoing method embodiments. In a possible example, the model training apparatus is used in a first worker node, the first worker node is deployed on a cloud computing platform, at least one second worker node is further deployed on the cloud computing platform, and the first worker node and the at least one second worker node use different training data to collaboratively train a model or a part of network layers in the model.

[00252] As shown in FIG. 9, the model training apparatus 900 includes a receiving module 910, a splitting module 920, a storage module 930, and an obtaining module 940. The model training apparatus 900 is configured to implement functions of the first worker node 210 in the method embodiments corresponding to FIG. 2 to FIG. 8b. In a possible example, a specific process in which the model training apparatus 900 is configured to implement the foregoing model training method includes the following process.

[00253] The receiving module 910 is configured to receive a storage request, where the storage request carries a first field used for splitting data used by the model or the part of network layers in the model, and the data used by the model or the part of network layers in the model includes a

parameter and/or training data of the model or the part of network layers in the model.

[00254] The splitting module 920 is configured to split, based on the first field in the storage request, the data that needs to be used by the model or the part of network layers in the model into N pieces of sub-data, where $N \geq 2$.

[00255] The storage module 930 is configured to store A pieces of sub-data of the N pieces of sub-data as a checkpoint file, where $1 \leq A \leq N$, a checkpoint file stored by the at least one second worker node includes B pieces of sub-data of the N pieces of sub-data, and B is a difference between N and A.

[00256] The obtaining module 940 is configured to: obtain a part or all of the B pieces of sub-data if data loss occurs in a process of training the model.

[00257] To further implement functions in the method embodiments shown in FIG. 2 to FIG. 8b, this application further provides a model training apparatus. FIG. 10 is a diagram 2 of a structure of a model training apparatus according to this application. The model training apparatus 900 further includes a sending module 950.

[00258] The sending module 950 is configured to send, to a storage node, a checkpoint file including A pieces of sub-data.

[00259] A receiving module 910, a splitting module 920, a storage module 930, and an obtaining module 940 may all be implemented by software, or may be implemented by hardware. For example, the following uses the receiving module 910 as an example to describe an implementation of the receiving module 910. Similarly, for implementations of the splitting module 920, the storage module 930, and the obtaining module 940, refer to the implementation of the receiving module 910.

[00260] The module is used as an example of a software functional unit, and the receiving module 910 may include code that is run on a compute instance. The compute 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 compute instances. For example, the receiving module 910 may include code that is run on a plurality of hosts/virtual machines/containers.

[00261] 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. Usually, one region may include a plurality of AZs.

[00262] Similarly, the plurality of hosts/virtual machines/containers configured to run the code may be distributed in a same VPC, or may be distributed in a plurality of VPCs. Usually, 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.

[00263] The module is used as an example of a hardware functional unit, and the receiving module 910 may include at least one compute device, for example, a server. Alternatively, the receiving module 910 may be a device implemented by using an application-specific integrated circuit (application-specific integrated circuit, ASIC), 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.

[00264] A plurality of compute devices included in the receiving module 910 may be distributed in a same region or different regions. The plurality of compute devices included in the receiving module 910 may be distributed in a same AZ or different AZs. Similarly, the plurality of compute devices included in the receiving module 910 may be distributed in a same VPC or 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.

[00265] It should be noted that in another embodiment, the receiving module 910 may be configured to perform any step in the model training method, the splitting module 920 may be configured to perform any step in the model training method, the storage module 930 may be configured to perform any step in the model training method, and the obtaining module 940 may be configured to perform any step in the model training method. Steps implemented by the receiving module 910, the splitting module 920, the storage module 930, and the obtaining module 940 may be specified as required. The receiving module 910, the splitting module 920, the storage module 930, and the obtaining module 940 separately implement different steps in the model training method to implement all functions of the first worker node 210.

[00266] It should be noted that the first worker node 210 in the foregoing embodiments may correspond to the model training apparatus 900, and may correspond to a corresponding entity for performing the methods in FIG. 2 to FIG. 8b according to embodiments of this application, and operations and/or functions of the modules in the model training apparatus 900 are separately used to implement procedures of the methods in the corresponding embodiments in FIG. 2 to FIG. 5. For brevity, details are not described herein again.

[00267] In addition, the model training apparatus 900 shown in FIG. 9 or FIG. 10 may alternatively be implemented by using a communication device. The communication device herein may be the first worker node 210 in the foregoing embodiments. When the communication device

is a chip or a chip system applied to a processing device, the model training apparatus 900 may alternatively be implemented by using the chip or the chip system.

[00268] An embodiment of this application further provides a chip system. The chip system includes a control circuit and an interface circuit. The interface circuit is configured to obtain an access control policy and an operation request. The control circuit is configured to implement functions of the first worker node 210 in the foregoing method based on the obtained access control policy and the obtained operation request.

[00269] In a possible design, the chip system further includes a memory configured to store program instructions and/or data. The chip system may include a chip, or may include a chip and another discrete component.

[00270] An embodiment of this application further provides a compute device. FIG. 11 is a diagram of a structure of a compute device according to an embodiment of this application. The compute device 1100 includes a bus 1102, a processor 1104, a memory 1106, and a communication interface 1108. The processor 1104, the memory 1106, and the communication interface 1108 are communicatively connected to each other through the bus 1102. The compute device 1100 may be a server or a terminal device. It should be understood that quantities of processors and memories in the compute device 1100 are not limited in this application. For example, the compute device 1100 may be the foregoing first worker node 210 or the foregoing second worker node.

[00271] The bus 1102 may be a peripheral component interconnect (peripheral component interconnect, PCI) bus, an extended industry standard architecture (extended industry standard architecture, EISA) bus, or the like. Buses may be classified into an address bus, a data bus, a control bus, and the like. For ease of indication, the bus is indicated by using only one line in FIG. 11. However, it does not indicate that there is only one bus or only one type of bus. The bus 1102 may include a path for information transmission between components (for example, the processor 1104, the memory 1106, and the communication interface 1108) of the compute device 1100.

[00272] The processor 1104 may include any one or more of processors such as a CPU, a GPU, a microprocessor (microprocessor, MP), or a DSP.

[00273] The memory 1106 may include a volatile memory (volatile memory), for example, a random access memory (random access memory, RAM). The processor 1104 may further include a non-volatile memory (non-volatile memory), for example, a read-only memory (read-only memory, ROM), a flash memory, a hard disk drive (hard disk drive, HDD), or a solid-state drive (solid-state drive, SSD).

[00274] The memory 1106 stores executable program code, and the processor 1104 executes the executable program code to separately implement functions of the receiving module 910, the splitting module 920, the storage module 930, and the obtaining module 940, so as to implement

the model training method. In other words, the memory 1106 stores instructions used for performing the model training method.

[00275] The communication interface 1108 uses a transceiver module, for example, but not limited to, a network interface card or a transceiver, to implement communication between the compute device 1100 and another device or a communication network. The compute device 1100 may be a computer (for example, a server) in a cloud data center, or a computer or a terminal in an edge data center.

[00276] 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, for example, a central server, an edge server, or a local server in a local data center. In some embodiments, the compute device may alternatively be a terminal device, for example, a desktop computer, a notebook computer, or a smartphone.

[00277] FIG. 12 is a diagram of a structure of a compute device cluster according to this application. The compute device cluster includes at least one compute device 1100. A memory 1106 in one or more compute devices 1100 in the compute device cluster may store same instructions for performing the model training method.

[00278] In some possible implementations, the memory 1106 in the one or more compute devices 1100 in the compute device cluster may alternatively separately store some instructions for performing the model training method. In other words, a combination of the one or more compute devices 1100 may jointly execute the instructions for performing the model training method.

[00279] It should be noted that, memories 1106 in different compute devices 1100 in the compute device cluster may store different instructions that are separately for performing some functions of the model training method. In other words, the instructions stored in the memories 1106 in different compute devices 1100 may implement functions of one or more of the receiving module 910, the splitting module 920, the storage module 930, and the obtaining module 940.

[00280] 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. 13 shows a possible implementation. FIG. 13 is a diagram of connection between compute devices according to this application. A compute device 1100A and a compute device 1100B are connected through a network. Specifically, the compute devices are connected to the network through communication interfaces of the compute devices. In this type of possible implementation, a memory 1106 in the compute device 1100A stores instructions for performing functions of the receiving module 910 and the splitting module 920. In addition, a memory 1106 in the compute device 1100B stores instructions for performing functions of the

storage module 930 and the obtaining module 940.

[00281] It should be understood that functions of the compute device 1100A shown in FIG. 13 may alternatively be implemented by a plurality of compute devices 1100. Similarly, functions of the compute device 1100B may alternatively be implemented by a plurality of compute devices 1100.

[00282] An embodiment of this application further provides a computer program product including instructions. The computer program product may be software or a program product that includes the instructions and that can be run on a compute device or be stored in any usable medium. When the computer program product is run on at least one compute device, the at least one compute device is enabled to perform the foregoing model training method.

[00283] An embodiment of this application further provides a computer-readable storage medium. The computer-readable storage medium may be any usable medium accessible by a compute device, or a data storage device, for example, 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 drive, 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. The instructions indicate the compute device to perform the model training method.

[00284] All or some of the foregoing embodiments may be implemented by using software, hardware, firmware, or any combination thereof. When software is used to implement the embodiments, all or a part of the embodiments may be implemented in a form of computer program product. The computer program product includes one or more computer programs or instructions. When the computer programs or the instructions are loaded and executed on a computer, the procedures or functions in embodiments of this application are all or partially performed. The computer may be a general-purpose computer, a dedicated computer, a computer network, a network device, user equipment, or another programmable apparatus. The computer program or instructions may be stored in a computer-readable storage medium, or may be transmitted from a computer-readable storage medium to another computer-readable storage medium. For example, the computer program or instructions may be transmitted from a website, computer, server, or data center to another website, computer, server, or data center in a wired or wireless manner. The computer-readable storage medium may be any usable medium that can be accessed by the computer, or a data storage device, for example, a server or a data center, integrating 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, may be an optical medium, for example, a digital video disc (digital video disc, DVD), or may be a semiconductor medium, for example, a solid-state drive (solid-state drive, SSD).

[00285] The foregoing descriptions are merely specific embodiments of this application, but are not intended to limit the protection scope of this application. Any modification 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 training method, wherein the method is applied to a first worker node, the first worker node is deployed on a cloud computing platform, at least one second worker node is further deployed on the cloud computing platform, the first worker node and the at least one second worker node use different training data to collaboratively train a model or a part of network layers in the model, and the method comprises:

receiving, by the first worker node, a storage request, wherein the storage request carries a first field used for splitting data used by the model or the part of network layers in the model, and the data used by the model or the part of network layers in the model comprises a parameter and/or training data of the model or the part of network layers in the model;

splitting, by the first worker node based on the first field in the storage request, the data that is in the first worker node and that needs to be used by the model or the part of network layers in the model into N pieces of sub-data, wherein $N \geq 2$;

storing, by the first worker node, A pieces of sub-data of the N pieces of sub-data as a checkpoint file, wherein $1 \leq A \leq N$, a checkpoint file stored by the at least one second worker node comprises B pieces of sub-data of the N pieces of sub-data, and B is a difference between N and A; and

obtaining, by the first worker node, a part or all of the B pieces of sub-data if data loss occurs in a process in which the first worker node trains the model.

2. The method according to claim 1, wherein a value of the first field is N, and the first worker node and (N–1) second worker nodes use different training data to collaboratively train the model or the part of network layers in the model.

3. The method according to claim 1 or 2, wherein the cloud computing platform further comprises a cloud management platform, the cloud management platform is configured to manage the first worker node and the second worker node, the cloud management platform provides a storage request configuration interface, the storage request configuration interface is configured to obtain configuration information entered by a user for data to be stored during model training, the configuration information is used to determine to split the data that is in the first worker node and that needs to be used by the model into the N pieces of sub-data, the first field in the storage request is determined based on the configuration information, and the receiving, by the first worker node, the storage request comprises:

receiving, by the first worker node, the storage request from the cloud management platform.

4. The method according to any one of claims 1 to 3, wherein if the data used by the model comprises the training data, the checkpoint file in the first worker node comprises first sub-data

and compressed second sub-data, wherein the first sub-data is the A pieces of sub-data, the second sub-data is training data used by the at least one second worker node in a process of training the model, and the B pieces of sub-data comprise the second sub-data; and

the obtaining, by the first worker node, the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model comprises:

obtaining, by the first worker node, the part of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model.

5. The method according to any one of claims 1 to 4, wherein if the data used by the model comprises the parameter, the checkpoint file in the first worker node comprises the A pieces of sub-data, wherein $1 \leq A < N$; and

the obtaining, by the first worker node, the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model comprises:

obtaining, by the first worker node, the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model.

6. The method according to any one of claims 1 to 5, wherein the storing, by the first worker node, the A pieces of sub-data of the N pieces of sub-data as the checkpoint file comprises:

storing, by the first worker node locally in the first worker node, the checkpoint file comprising the A pieces of sub-data; and

the obtaining, by the first worker node, the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model comprises:

obtaining, by the first worker node, the B pieces of sub-data from the at least one second worker node if the data loss occurs in the process in which the first worker node trains the model.

7. The method according to any one of claims 1 to 6, wherein the cloud computing platform further comprises a storage node, and the method further comprises:

sending, by the first worker node to the storage node, the checkpoint file comprising the A pieces of sub-data; and

the obtaining, by the first worker node, the part or all of the B pieces of sub-data if the data loss occurs in the process in which the first worker node trains the model comprises:

if the data loss occurs in the process in which the first worker node trains the model, obtaining, by the first worker node, the B pieces of sub-data from the at least one second worker node, and obtaining the A pieces of sub-data from the storage node.

8. The method according to any one of claims 1 to 7, wherein the storage request further carries a status field indicating storage of the checkpoint file by the first worker node, the status field indicates a target network layer in the model, and the first worker node is configured to train the target network layer.

9. The method according to any one of claims 1 to 8, wherein the storage request further carries a second field indicating storage frequency, the storage frequency comprises time frequency or training epoch frequency, and the storing, by the first worker node, the A pieces of sub-data of the N pieces of sub-data as the checkpoint file comprises:

if the time frequency or the training epoch frequency is satisfied in a process in which the first worker node trains the model or the part of network layers in the model, storing the A pieces of sub-data of the N pieces of sub-data as the checkpoint file.

10. The method according to any one of claims 1 to 9, wherein the data loss indicates one or more of the following: a fault of the first worker node, and an interruption of the first worker node.

11. A model training apparatus, wherein the apparatus is used in a first worker node, the first worker node is deployed on a cloud computing platform, at least one second worker node is further deployed on the cloud computing platform, the first worker node and the at least one second worker node use different training data to collaboratively train a model or a part of network layers in the model, and the apparatus comprises:

a receiving module, configured to receive a storage request, wherein the storage request carries a first field used for splitting data used by the model or the part of network layers in the model, and the data used by the model or the part of network layers in the model comprises a parameter and/or training data of the model or the part of network layers in the model;

a splitting module, configured to split, based on the first field in the storage request, the data that needs to be used by the model or the part of network layers in the model into N pieces of sub-data, wherein $N \geq 2$;

a storage module, configured to store A pieces of sub-data of the N pieces of sub-data as a checkpoint file, wherein $1 \leq A \leq N$, a checkpoint file stored by the at least one second worker node comprises B pieces of sub-data of the N pieces of sub-data, and B is a difference between N and A; and

an obtaining module, configured to: obtain a part or all of the B pieces of sub-data if data loss occurs in a process of training the model.

12. The apparatus according to claim 11, wherein a value of the first field is N, and the first worker node and (N–1) second worker nodes use different training data to collaboratively train the model or the part of network layers in the model.

13. The apparatus according to claim 11 or 12, wherein the cloud computing platform further comprises a cloud management platform, the cloud management platform is configured to manage the first worker node and the second worker node, the cloud management platform provides a storage request configuration interface, the storage request configuration interface is configured to obtain configuration information entered by a user for data to be stored during model training, the

configuration information is used to determine to split the data that is in the first worker node and that needs to be used by the model into the N pieces of sub-data, the first field in the storage request is determined based on the configuration information, and the receiving module is specifically configured to receive the storage request from the cloud management platform.

14. The apparatus according to any one of claims 11 to 13, wherein if the data used by the model comprises the training data, the checkpoint file in the first worker node comprises first sub-data and compressed second sub-data, wherein the first sub-data is the A pieces of sub-data, the second sub-data is training data used by the at least one second worker node in a process of training the model, the B pieces of sub-data comprise the second sub-data, and the obtaining module is specifically configured to: obtain the part of the B pieces of sub-data if the data loss occurs in the process of training the model.

15. The apparatus according to any one of claims 11 to 14, wherein if the data used by the model comprises the parameter, the checkpoint file in the first worker node comprises the A pieces of sub-data, wherein $1 \leq A < N$, and the obtaining module is specifically configured to: obtain the B pieces of sub-data if the data loss occurs in the process of training the model.

16. The apparatus according to any one of claims 11 to 15, wherein the storage module is specifically configured to store, locally in the first worker node, the checkpoint file comprising the A pieces of sub-data, and the obtaining module is specifically configured to: obtain the part or all of the B pieces of sub-data from the at least one second worker node if the data loss occurs in the process of training the model.

17. The apparatus according to any one of claims 11 to 16, wherein the cloud computing platform further comprises a storage node, and the apparatus further comprises a sending module;

the sending module is configured to send, to the storage node, the checkpoint file comprising the A pieces of sub-data; and

the obtaining module is specifically configured to: if the data loss occurs in the process of training the model, obtain the B pieces of sub-data from the at least one second worker node, and obtain the A pieces of sub-data from the storage node.

18. A computer cluster, comprising at least one compute device, wherein each compute device comprises a processor and a memory; and

a processor of the at least one compute device is configured to execute instructions stored in a memory of the at least one compute device, to enable the computer cluster to perform the method according to any one of claims 1 to 10.

19. A computer-readable storage medium, wherein the storage medium stores a computer program or instructions, and when the computer program or the instructions are executed by a processing device, the method according to any one of claims 1 to 10 is implemented.

20. A computer program product, comprising a computer program or instructions, wherein when the computer program or the instructions are executed by a processing device, the method according to any one of claims 1 to 10 is implemented.