

FEDERATED TRAINING METHOD AND APPARATUS, AND COMPUTER SYSTEM

TECHNICAL FIELD

[0001] One or more embodiments of this specification relate to the field of computer technologies, and in particular, to a federated training method and apparatus, and a computer system.

BACKGROUND

[0002] As artificial intelligence and a big data era arrive, machine learning technologies are widely applied to various fields. However, due to data security and privacy preservation, a conventional centralized machine learning method has many limitations. Consequently, federated learning emerges.

[0003] Federal learning (or referred to as federated training) is a distributed machine learning technology, and allows a plurality of participants to perform model training when data privacy is protected. Each participant only locally maintains a dataset of the participant, and exchanges some model updating parameters based on a federated learning algorithm, to implement joint training of a model. Federated learning is widely applied to fields such as the medical field, the financial field, and the internet of things filed because of characteristics of data privacy preservation and confidentiality of federated learning.

[0004] However, because federated learning involves cooperation of the plurality of participants, a model optimization problem of federated learning becomes more complex. Therefore, embodiments of this specification provide a new algorithm for optimizing a model in a federated learning framework, to better satisfy an actual application requirement, for example, improve training efficiency of federated learning, implement fast convergence of a model parameter, and improve availability of a trained model.

SUMMARY

[0005] One or more embodiments of this specification describe a federated training method and a corresponding apparatus, to satisfy a higher requirement for federated learning performance and effect in actual applications.

[0006] According to a first aspect, a federated training method is provided, involving an intermediate party and a plurality of participants. The method is applied to any participant, and any round of iterative training involved in the method includes: calculating a first gradient corresponding to a current first parameter of a local model based on a batch of local training data
5 corresponding to the current round; calculating, based on the first gradient and a current local approximation matrix, a target vector indicating a parameter updating direction, where the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; updating the first parameter to a second parameter based on the target vector and an obtained step, and calculating a second gradient corresponding to the second parameter;
10 sending the second parameter and the second gradient to the intermediate party; receiving a target parameter and a target matrix from the intermediate party, where the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of participants; and correspondingly updating a local model parameter
15 and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.

[0007] According to a second aspect, a federated training method is provided, involving an intermediate party and a plurality of participants. The method is applied to the intermediate party, and any round of iterative training involved in the method includes: receiving a current second
20 parameter and a corresponding second gradient of a local model of any participant from the participant, where the second parameter is obtained by updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the first gradient is a gradient of the first parameter on a batch of local training data corresponding to the current round, and the local approximation matrix is used to
25 approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; determining a global model parameter and a global approximation matrix that are obtained through updating in the current round based on second parameters and second gradients that are respectively provided by the plurality of participants; and providing a target parameter and a target matrix that correspond to any participant to the participant based on the global model parameter
30 and the global approximation matrix that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to perform a next round of iterative training.

[0008] According to a third aspect, a federated training apparatus is provided, involving an
35 intermediate party and a plurality of participants. The apparatus is integrated into any participant,

and the apparatus performs any round of iterative training by using the following modules: a first gradient calculation module, configured to calculate a first gradient corresponding to a current first parameter of a local model based on a batch of local training data corresponding to the current round; a target vector calculation module, configured to calculate, based on the first gradient and
5 a current local approximation matrix, a target vector indicating a parameter updating direction, where the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; a first parameter updating module, configured to: update the first parameter to a second parameter based on the target vector and an obtained step, and calculate a second gradient corresponding to the second parameter; an intermediate data transmission
10 module, configured to send the second parameter and the second gradient to the intermediate party; a target data receiving module, configured to receive a target parameter and a target matrix from the intermediate party, where the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of
15 participants; and a local updating module, configured to correspondingly update a local model parameter and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.

[0009] According to a fourth aspect, a federated training apparatus is provided, involving an intermediate party and a plurality of participants. The apparatus is integrated into the intermediate
20 party, and the apparatus performs any round of iterative training by using the following modules: an intermediate data receiving module, configured to receive a current second parameter and a corresponding second gradient of a local model of any participant from the participant, where the second parameter is obtained by updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the
25 first gradient is a gradient of the first parameter on a batch of local training data corresponding to the current round, and the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; a global data updating module, configured to determine, based on second parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are
30 obtained through updating in the current round; and a target data sending module, configured to provide a target parameter and a target matrix that correspond to any participant to the participant based on the global model parameter and the global approximation matrix that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to
35 perform a next round of iterative training.

- 5 [0010] According to a fifth aspect, a computer system is provided, including a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor. The processor is integrated into any one of a plurality of participants, the plurality of participants cooperatively perform federated training with an intermediate party, and when the processor executes the program, steps in the method provided in the first aspect are implemented:
- 10 [0011] According to a sixth aspect, a computer system is provided, including a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor. The processor is integrated into an intermediate party, the intermediate party cooperatively perform federated training with a plurality of participants, and when the processor executes the program, steps in the method provided in the first aspect are implemented:
- 15 [0012] According to a seventh aspect, a computing device is provided. The computing device includes a storage and a processor. The storage stores executable code, and when the processor executes the executable code, the method provided in the first aspect or the second aspect is implemented.
- [0013] According to the methods and apparatuses provided in the embodiments of this specification, in federated learning, data privacy of each of a plurality of participants can be better protected, a data transmission amount is reduced, and model optimization efficiency is improved.

BRIEF DESCRIPTION OF DRAWINGS

- 20 [0014] To describe the technical solutions in the embodiments of the present invention more clearly, the following briefly describes the accompanying drawings needed for describing the embodiments. Clearly, the accompanying drawings in the following descriptions show merely some embodiments of the present invention, and a person of ordinary skill in the art can still derive other drawings from these accompanying drawings without creative efforts.
- 25 [0015] FIG. 1 is a schematic diagram of an implementation architecture of a federated training scheme according to an embodiment of this specification;
- [0016] FIG. 2 is a schematic diagram of communication and interaction between an intermediate party and a plurality of participants involved in a federated training method according to an embodiment of this specification;
- 30 [0017] FIG. 3 is a schematic diagram of a functional structure of a federated training apparatus integrated into any participant according to an embodiment of this specification;
- [0018] FIG. 4 is a schematic diagram of a functional structure of a federated training apparatus integrated into an intermediate party according to an embodiment of this specification; and
- [0019] FIG. 5 is a schematic structural diagram of a computer system according to an

embodiment of this specification.

DESCRIPTION OF EMBODIMENTS

[0020] The following describes the solutions provided in this specification with reference to the accompanying drawings.

5 [0021] As mentioned above, because federated learning involves cooperation of a plurality of participants, a model optimization problem of federated learning becomes more complex. A higher requirement in actual applications cannot be satisfied based on a conventional optimization algorithm in federated learning. For example, in the conventional optimization algorithm, there is a relatively high requirement for computing and storage resources, and there is a slower
10 convergence speed.

[0022] The applicant further observes that in a Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm and an improved algorithm of the BFGS optimization algorithm, for example, a limited-memory BFGS (L-BFGS) optimization algorithm or an L-BFGS with bound constraints (L-BFGS-B) optimization algorithm, there is a relatively fast convergence speed and
15 relatively good numerical stability, and a large-scale problem can be processed. For brief description, the BFGS algorithm and the improved algorithm of the BFGS algorithm are collectively referred to as a BFGS system algorithm below.

[0023] However, a conventional BFGS system algorithm is only applicable to centralized machine learning, and is inapplicable to a federated learning scenario. Based on the above-
20 mentioned observations and analysis, the applicant proposes to modify the BFGS system algorithm, so that the BFGS system algorithm is applicable to the federated learning scenario, in other words, to organically combine the BFGS system algorithm and a federated learning framework. In this way, a new federated learning scheme is proposed.

[0024] FIG. 1 is a schematic diagram of an implementation architecture of a federated training
25 scheme according to an embodiment of this specification. As shown in FIG. 1, implementation of the scheme involves an intermediate party and a plurality of participants. In FIG. 1, a Pi party represents a participant. In any $(k + 1)^{\text{th}}$ round of iterative training involved in the scheme, the Pi party updates a local model parameter x_k and a gradient ∇_k on data d of the Pi party based on a batch of local training data d and an approximation matrix B_k of Hessian inverse matrix for
30 the local model parameter x_k , and provides a parameter x_k' and a gradient ∇_k' that are obtained through updating to the intermediate party. Therefore, the intermediate party can determine, based on parameters x_k' and gradients ∇_k' that are respectively provided by the plurality of participants, a global model parameter $x_{k+1}^{(G)}$ and a global approximation matrix $B_{k+1}^{(G)}$ that are

obtained through updating in the current round, and further determine, based on $x_{k+1}^{(G)}$ and $B_{k+1}^{(G)}$, a target parameter $x^{(t)}$ and a target matrix $B^{(t)}$ that are to be sent to the Pi party. Based on this, the Pi party updates a parameter x_k' of a local model to x_{k+1} based on the target parameter $x^{(t)}$, and updates the approximation matrix B_k to an approximation matrix B_{k+1} based on the target matrix $B^{(t)}$, so that the Pi party can participate in a next round (the $(k + 2)^{\text{th}}$ round) of iterative training based on the parameter x_{k+1} and the approximation matrix B_{k+1} that are obtained through updating in the current round. In this way, a quantity of iteration round and a data transmission amount can be reduced when a training effect is ensured, to effectively implement model optimization in a federated learning framework.

10 **[0025]** Before specific implementation steps of the federated training scheme are further described, an association relationship between and entity forms of the intermediate party and the plurality of participants and a distribution of training data and a global model parameter in the plurality of participants are first described.

15 **[0026]** (1) For the association relationship between the intermediate party and the plurality of participants, usually, the plurality of participants are independent of each other. The intermediate party can be one of the plurality of participants, or can be another party different from the plurality of participants.

20 **[0027]** (2) For the entity forms of the intermediate party and the plurality of participants, it can be understood that all parties involved in the scheme can be implemented as any apparatus, platform, server, device cluster, etc. that has a computing and processing capability.

[0028] (3) For the distribution of the training data in the plurality of participants:

25 **[0029]** In a typical case, the training data are vertically distributed in the plurality of participants. To be specific, the plurality of participants have the same sample space and different feature space. In other words, the plurality of participants have different feature parts of the same sample.

[0030] In an embodiment, a shared model (or referred to as a global model) that the plurality of participants want to jointly train is a risk assessment model, and is used to predict a future credit risk of a user. For example, a data distribution in the plurality of participants is as follows:

30 **[0031]** A participant A (for example, a bank) has credit information such as a loan record and a repayment history of the user.

[0032] A participant B (for example, an e-commerce platform) has E-commerce data such as an online shopping behavior and a consumption habit of the same user.

[0033] A participant C (for example, a credit rating agency) has a credit score of the same user and other credit-related information.

35 **[0034]** In another embodiment, the plurality of participants want to jointly train a

recommendation model, to recommend, to the user, an object that the user may be interested in, for example, a product, digital content such as an article or audio and videos, or an official account. For example, a data distribution in the plurality of participants is as follows:

[0035] A participant A (for example, an online video platform) has data such as a watching history, a preferred video type, and a watching time of the user.

[0036] A participant B (for example, an e-commerce platform) has data such as a shopping history, a preferred product type, and a search history of the same user.

[0037] Different feature sets for the same sample in the plurality of participants are described above. It should be noted that all user data mentioned in this specification are collected and used when the user confirms to perform authorization.

[0038] In addition, the plurality of participants each have a sample label. For content of the sample label, for example, in the above-mentioned risk assessment scenario, the sample label can be a credit risk label. For another example, in the above-mentioned recommendation scenario, the sample label can be a behavior category label, and indicates whether a specific user makes a predetermined behavior (for example, purchasing) for a specific object (for example, a specific product). In addition, for a source of the sample label in each participant, for example, the sample label can be obtained by each participant by independently perform marking. For another example, the plurality of participants can share and unify labels for the same sample.

[0039] In addition, for the same sample mentioned above, usually, data between the plurality of participants are in a secure isolation state, and no participant learns of a sample owned by another participant. Therefore, the plurality of participants can jointly execute a private set intersection (PSI) protocol based on local sample identifier sets, and an execution result is that the plurality of participants can learn of an intersection set of a plurality of sample identifier sets, but cannot learn of any other information.

[0040] A typical case in which the training data are vertically distributed in the plurality of participants is described above. In addition, federated learning used to cope with the case in which the training data are vertically distributed is also referred to as vertical federated learning.

[0041] In another typical case, the training data are horizontally distributed in the plurality of participants. To be specific, the plurality of participants have different sample space and the same feature space. In other words, the plurality of participants have all features of different samples.

[0042] In an embodiment, the plurality of participants want to jointly train a credit scoring model for a user, to predict a future credit risk of the user. For example, a data distribution in the plurality of participants is as follows:

[0043] A participant A (for example, a bank 1) is located in an area 1, and has credit information such as a loan record, a repayment history, and a credit card use book of the user.

- [0044] A participant B (for example, a bank 2) is located in an area 2, and also has credit information such as a credit record, a repayment history, and a credit card use book of a customer of the participant B.
- [0045] A participant C (for example, an online lending platform) collects a borrowing record and a repayment status of the user and transaction behavior data of the user on the platform.
- [0046] It can be understood that in a horizontal distribution scenario, the plurality of participants independently hold a sample label of a local sample. Because the plurality of participants respectively maintain different samples, label values of the samples do not need to be exchanged or shared.
- [0047] A typical case in which the training data are vertically distributed in the plurality of participants is described above. In addition, federated learning used to cope with the case in which the training data are vertically distributed is also referred to as horizontal federated learning.
- [0048] It should be understood that the distribution of the training data in the plurality of participants is not limited to a horizontal distribution and a vertical distribution, and there is still a more complex case. Examples are not exhaustively listed here. The solutions of the embodiments of this specification are applicable to all data distribution scenarios. To help understand the solutions in the following, example descriptions are mainly provided with reference to vertical federated learning and horizontal federated learning.
- [0049] (4) For a distribution of a global model parameter in the plurality of participants:
- [0050] It should be understood that the distribution of the global model parameter in the plurality of participants usually needs to be adaptively designed based on the distribution of the training data in the plurality of participants. For example, in a horizontal federated learning scenario, a relatively simple case is that the plurality of participants can each hold a complete copy of the global model.
- [0051] For another example, in a vertical federated learning scenario, the plurality of participants hold different feature parts of the same sample, and correspondingly, each participant can hold only some model parameters in the global model parameter that are related to processing of a local feature part. In this case, a local model of each participant is a local model of the global model.
- [0052] To facilitate understanding, the association relationship between the entity forms of the intermediate party and the plurality of participants and the distribution of the training data and the global model parameter in the plurality of participants are described above. Next, specific implementation steps of the federated training scheme shown in FIG. 1 are further described with reference to FIG. 2 and more embodiments.
- [0053] FIG. 2 is a schematic diagram of communication and interaction between an

intermediate party and a plurality of participants involved in a federated training method according to an embodiment of this specification. The federated training method includes a plurality of rounds of iterative training of a local model of each participant and a global model maintained by the intermediate party. Any $(k + 1)^{\text{th}}$ round of iterative training includes the following interaction steps.

[0054] First, in step S201, each participant (which is shown as a Pi party in FIG. 2) calculates a first gradient ∇_k corresponding to a current first parameter x_k of the local model based on a batch of local training data d corresponding to the current round.

[0055] It should be understood that, in the current round of iterative training, local training data respectively used by the plurality of participants correspond to the same batch of sample identifiers (for example, a batch of user identifiers). A mapping relationship between a quantity of iteration rounds and a sample identifier can be negotiated in advance by the plurality of participants, or can be broadcast to another participant after a specified participant randomly samples a batch of sample identifiers in each round of iterative training. This is not limited in this embodiment of this specification.

[0056] In an embodiment, it is assumed that federated learning is vertical federated learning, and the plurality of participants respectively have different feature parts of the same sample. In this case, each participant (Pi party) can determine a batch of sample identifiers corresponding to the current round based on the mapping relationship that is between a quantity of iteration rounds and a sample identifier and that is negotiated in advance by the plurality of participants, and then use a local feature part corresponding to the batch of sample identifiers as a batch of local training data. Alternatively, the Pi party can obtain a corresponding local feature and label based on a batch of sample identifiers provided by the intermediate party, and use the local feature and label as a batch of local training data. In conclusion, in a vertical federated learning scenario, local training data respectively used by the plurality of participants in each round correspond to the same batch of sample identifiers. In addition, a local model of the Pi party can be the i^{th} local model of a global model.

[0057] In another embodiment, it is assumed that federated learning is horizontal federated learning, and the plurality of participants respectively have complete features of different samples. In this case, a Pi party can randomly sample a batch of sample data as a batch of training data to be used in the current round. In conclusion, the plurality of participants are completely independent of each other in each round of determining the used local training data. In addition, a local model of the Pi party can be a copy of the global model.

[0058] It should be understood that, in this specification, a start parameter of the local model deployed by the Pi party in the current round (the $(k + 1)^{\text{th}}$ round) of iteration is referred to as a

first parameter (denoted as x_k), and the parameter of the local model is updated for a plurality of times in the current round of iteration. When the current round of iteration is the first round, a first parameter x_0 can be obtained by the Pi party through local initialization, or can be obtained from the intermediate party. When the current round of iteration is not the first round, the first parameter x_k is a parameter obtained through updating in a previous round (the k^{th} round). In addition, similar terms such as "first" in "first parameter" and "first" and "second" elsewhere in this specification are used to distinguish between similar things, and do not have another limitation function such as sorting.

[0059] Usually, to calculate a gradient of the model parameter, a target function (for example, a loss function) in a model optimization objective, a current parameter value of a model parameter, and training data (including a sample feature and a label) need to be determined. For the current parameter value of the model parameter and the training data, the first parameter x_k and the batch of local training data d are correspondingly described above. A target function $f(x)$ used by the Pi party can be independently selected or designed by the Pi party. Alternatively, the plurality of participants can negotiate a function type in advance, for example, a cross-entropy loss function. In addition, the gradient can be specifically calculated in an existing manner. Details are omitted for simplicity.

[0060] Based on the above-mentioned descriptions, the Pi party can calculate the first gradient ∇_k (or denoted as ∇f_k) corresponding to the current first parameter x_k of the local model based on the local training data used in the current round (denoted as the $(k + 1)^{\text{th}}$ round) and the target function $f(x)$.

[0061] Next, in step S202, the Pi party calculates, based on the first gradient ∇f_k and a current local approximation matrix B_k , a target vector p_k indicating a parameter updating direction. The local approximation matrix B_k is used to approximate an inverse matrix of a Hessian matrix corresponding to the current first parameter x_k .

[0062] It should be noted that, compared with a case of directly using the first gradient ∇f_k (the gradient is a first derivative of the model parameter) as the target vector p_k indicating the parameter updating direction, in a case of additionally introducing the inverse matrix of the Hessian matrix (which is a second derivative of the model parameter) based on the first gradient ∇f_k , there are the following beneficial effects: (1) Local curvature of the target function $f(x)$ can be estimated more accurately, to implement a faster convergence speed with a smaller quantity of iterations. (2) The second derivative is considered, to resist numerical instability and overfitting risks in a training process to some extent, and enhance robustness of the model. (3) It helps better understand and adapt to a nonlinear change of the target function $f(x)$, to improve a training effect and prediction accuracy of the model.

[0063] However, when there are a relatively large quantity of model parameters, a large amount of computing resources need to be consumed for solving the Hessian matrix or the inverse matrix of the Hessian matrix. Therefore, with reference to an idea of a BFGS algorithm, it is proposed to give up explicitly calculating the Hessian matrix and the inverse matrix of the Hessian matrix; instead, the model parameter is updated and optimized by maintaining an estimate of the inverse matrix of the Hessian matrix. In this specification, the estimate of the inverse matrix of the Hessian matrix is referred to as an approximation matrix.

[0064] The local approximation matrix B_k of the Pi party is used to approximate the inverse matrix of the Hessian matrix corresponding to the current first parameter x_k of the local model.

It should be understood that, similar to the local model parameters, the local approximation matrix B_k is also updated in each round of iteration. When the current round of iteration is the first round, the local approximation matrix B_k can be obtained by the Pi party through local initialization, for example, initialized into a unit matrix, or can be obtained from the intermediate party. When the current round of iteration is not the first round, the local approximation matrix B_k is obtained through updating in the previous round. In addition, in the vertical federated learning scenario, the local model parameter of the Pi party is a part of a global model parameter. Correspondingly, a magnitude of the local approximation matrix B_k is relatively small, and a relatively small quantity of storage resources need to be occupied.

[0065] In this step, an opposite number of the product of the first gradient ∇f_k and the current local approximation matrix B_k can be calculated and used as the target vector p_k indicating the parameter updating direction. This can be expressed as the following formula:

$$p_k = -B_k * \nabla f_k \quad (1)$$

[0066] It should be understood that, in this step, the target vector p_k can be further calculated based on a variant of Formula (1). For example, the right side of the formula is multiplied by a scaling coefficient.

[0067] Based on the above-mentioned descriptions, each participant (Pi party) can calculate the target vector p_k indicating the parameter updating direction.

[0068] Then, in step S203, the Pi party updates the current first parameter x_k of the local model to a second parameter x_k' based on the target vector p_k and an obtained step α_k , and calculates a second gradient $\nabla f_k'$ corresponding to the second parameter x_k' . It should be noted that the second parameter x_k' is further updated subsequently in the current round of training.

[0069] For obtaining of the step α_k , in an embodiment, the Pi party can determine the step through a line search based on the target vector p_k , so that a descending amplitude of the target function on the batch of local training data used in the current round is relatively large. It should be understood that the line search can be an exact line search: An optimal step is exactly determined

in a parsing method or a high-precision numerical method. In this method, the optimal step can be obtained theoretically, but in practice, calculation costs are usually relatively high. Alternatively, an inexact line search can be used. A good step is approximately determined based on some criteria and an iteration process. Inexact line search methods such as a Backtracking-line search, a Goldstein condition, and a Wolfe condition are more commonly used in actual applications, because calculation costs are reduced when a satisfactory convergence speed is provided.

[0070] Further, with reference to an idea of an L-BFGS-B algorithm, it is proposed to support the Pi party to set a local service constraint condition, so that the local constraint condition needs to be satisfied when the step α_k is determined through the line search. For a source of the local constraint condition, in a possible case, the local constraint condition can be independently set by the Pi party based on a service requirement, etc. For example, the local service constraint condition can be a budget constraint, an exposure diversity constraint, or a non-negative constraint of the parameter. In another possible case, a plurality of participants can jointly negotiate a global constraint condition, and then each adaptively set the local constraint condition based on the local model parameter, the training data, etc. For example, the global constraint condition can also be a budget constraint, an exposure diversity constraint, a non-negative constraint of a parameter, etc.

[0071] In addition, mathematical forms of the local constraint condition can basically include an equation constraint and an inequation constraint, which are respectively expressed as $h(x) = 0$ and $g(x) \leq 0$ in this specification. It should be understood that, $h(x) = 0$ represents a set of several equation constraints, and $g(x) \leq 0$ represents a set of several inequation constraints. In addition, $h(x) = 0$ and $g(x) \leq 0$ do not need to exist simultaneously, and there can be only one of the constraints. To facilitate understanding, for example, it is assumed that marketing costs need to be controlled within a budget in an advertisement recommendation scenario. In this case, for a target user group, a probability that each target user taps an advertisement can be calculated based on a parameter x of the local model, and then an expected value of total costs is calculated with reference to costs of each advertisement. The constraint condition can be expressed as that the expected value minus the budget is less than or equal to zero, and does not satisfy a form of the inequation $g(x) \leq 0$.

[0072] Based on this, this step can be implemented as follows: The Pi party determines the step α_k through the line search based on the target vector p_k , so that the descending amplitude of the target function on the batch of local training data reaches a predetermined standard, for example, reaches a maximum amplitude or is greater than a predetermined threshold when the local constraint condition is satisfied. The local constraint conditions $g(x) \leq 0$ and $h(x) = 0$ are converted into the following forms in the current round of iteration:

$$g(x_k + \alpha_k * p_k) \leq 0 \quad (2)$$

$$h(x_k + \alpha_k * p_k) = 0 \quad (3)$$

[0073] In Formula (2) and Formula (3), x_k represents the current first parameter of the local model, α_k represents the step to be solved, and p_k represents the target vector indicating the parameter updating direction. It can be understood that α_k and p_k are calculated to update the first parameter x_k . Correspondingly, when p_k and x_k are known, α_k can be solved with an objective that an updated model parameter (that is, $x_k + \alpha_k * p_k$) still satisfies the local constraint condition.

[0074] For example, the step α_k that satisfies the local constraint condition may be determined through the line search in a projection method, a penalty method, a binary search, etc.

[0075] The foregoing describes a case in which the step α_k is determined through the line search, so that the descending amplitude of the target function on the batch of local training data used in the current round satisfies the predetermined standard (optionally, when the local constraint condition is satisfied).

[0076] In another embodiment, the step α_k can be preset by a worker based on actual experience. For example, the step α_k is reduced step by step as the quantity of rounds of iteration increases.

[0077] Obtaining of the step α_k used in the current round is described above. The current first parameter x_k of the local model is updated to the second parameter x_k' based on the target vector p_k and the obtained step α_k . For example, the following formula can be used:

$$x_k' = x_k + \alpha_k * p_k \quad (4)$$

[0078] In an embodiment, if the local constraint condition is not introduced when the step α_k is determined, after the second parameter x_k' is calculated based on Formula (4), the second parameter x_k' can be further updated to satisfy the local constraint condition. It should be understood that, for descriptions of the local constraint condition, references can be made to related descriptions in the above-mentioned embodiments. Alternatively, even if the local constraint condition is introduced when the step α_k is determined, an update operation can still be performed on the second parameter x_k' , to ensure that the second parameter x_k' satisfies the local constraint condition.

[0079] Further, in a specific embodiment, the local constraint condition includes an inequation constraint condition $q(x) \leq 0$. Correspondingly, that the second parameter x_k' is updated to satisfy the local constraint condition can include: mapping, to a constraint boundary in a projection method, a component that is in the second parameter x_k' and that is outside a constraint range corresponding to the inequation constraint condition. For example, for each component $q_i(x_k') > 0$ that does not satisfy the inequation constraint, a corresponding component of an updating point x_k' can be obtained through the following projection operation:

$$x_i(k)' = \max(x_i(k)', 0) \quad (5)$$

[0080] In another specific embodiment, the local constraint condition includes the equation constraint condition $h(x) = 0$. Correspondingly, that the second parameter x_k' is updated to satisfy the local constraint condition can include: updating the second parameter x_k' by solving a Lagrange function constructed based on the target function $f(x)$ and the equation constraint condition $h(x) = 0$.

[0081] For example, a form of the constructed Lagrange function (or referred to as an augmented target function) is in the following formula:

$$L(x_k', \lambda) = f(x_k') + \lambda^T * h(x_k') \quad (6)$$

[0082] Here, T represents a transpose operator. It should be understood that, when $h(x_k')$ is a group of equations, a Lagrange multiplier λ is a vector, and $\lambda^T * h(x_k')$ can implement a linear combination of a group of equations corresponding to $h(x_k')$ based on the vector λ . In addition, when x_k' is updated, the Lagrange multiplier λ needs to be updated simultaneously. In addition, an updating point x_k' and the Lagrange multiplier λ can be specifically obtained by solving a condition that a gradient of the Lagrange function is zero.

[0083] Based on the above-mentioned descriptions, the second parameter x_k' calculated based on Formula (4) can be obtained, or the second parameter x_k' that is further updated to satisfy the local constraint condition can be obtained.

[0084] Further, the second gradient $\nabla f_k'$ corresponding to the second parameter x_k' can be calculated. As described above, to calculate the gradient of the model parameter, the target function in the model optimization target, the current parameter value of the model parameter, and the training data need to be determined. Correspondingly, the second gradient $\nabla f_k'$ is calculated based on the local target function $f(x)$, the second parameter x_k' , and the batch of local training data d corresponding to the current round.

[0085] In the foregoing, the Pi party can obtain the second gradient $\nabla f_k'$ corresponding to the second parameter x_k' .

[0086] Then, in step S204, the Pi party sends the second parameter x_k' and the second gradient $\nabla f_k'$ to the intermediate party.

[0087] In an embodiment, in consideration that there is still a risk of disclosing local original training data of the Pi party when a parameter plaintext or a gradient plaintext is directly transferred, it is proposed that the Pi party sends the second parameter x_k' and the second gradient $\nabla f_k'$ to the intermediate party after separately performing noise addition processing on the second parameter x_k' and the second gradient $\nabla f_k'$. For example, noise added in noise addition processing can be a differential privacy (DP) noise that can be used to quantize privacy

preservation.

[0088] In another embodiment, the Pi party can directly send plaintexts of the second parameter x_k' and the second gradient $\nabla f_k'$ to the intermediate party.

[0089] Based on the above-mentioned descriptions, the intermediate party can receive the second parameter x_k' and the second gradient $\nabla f_k'$ that are provided by each party.

[0090] Therefore, in step S205, the intermediate party determines, based on a plurality of second parameters $\{x_k'\}$ and a plurality of second gradients $\{\nabla f_k'\}$ that are correspondingly provided by the plurality of participants, a global model parameter $x_{k+1}^{(G)}$ and a global approximation matrix $B_{k+1}^{(G)}$ that are obtained through updating in the current round.

[0091] Specifically, the plurality of second parameters $\{x_k'\}$ corresponding to a plurality of local models are aggregated into a parameter corresponding to a global model, and the parameter is used as a global model parameter $x_{k+1}^{(G)}$ obtained through updating in the current round. The plurality of second gradients $\{\nabla f_k'\}$ corresponding to the plurality of local models are aggregated into a parameter gradient corresponding to the global model, and the parameter gradient is used as a global gradient $\nabla f_{k+1}^{(G)}$ obtained through updating in the current round. It should be understood that an aggregation manner of the parameter and the gradient adapts to a relationship between the local model and the global model. In addition, a weight can be further introduced during aggregation. For example, the intermediate party tends to allocate a higher weight to a parameter and a gradient that are provided by a participant with a stronger authority or a larger magnitude of training data.

[0092] In an embodiment, it is assumed that federated learning is vertical federated learning, and the plurality of local models correspondingly deployed by the plurality of participants are all local models of the global model. In this case, processing such as concatenation can be performed on the plurality of second parameters $\{x_k'\}$ based on a structural relationship between the local model and the global model, to obtain the global model parameter $x_{k+1}^{(G)}$. Similarly, processing such as concatenation is performed on the plurality of second gradients $\{\nabla f_k'\}$, to obtain the global gradient $\nabla f_{k+1}^{(G)}$.

[0093] In another embodiment, it is assumed that federated learning is horizontal federated learning, and the plurality of local models correspondingly deployed by the plurality of participants are all copies of the global model. In this case, a result of performing weighted summation on the plurality of second parameters $\{x_k'\}$ can be used as the global model parameter $x_{k+1}^{(G)}$. For example, weighted summation of the parameters can be performed based on the following formula:

$$x_{k+1}^{(G)} = \sum_{i=1}^n \beta_i * i_x_k' \quad (7)$$

[0094] Here, i in $i_{x_k'}$ explicitly denotes the i^{th} participant, β_i denotes a weight allocated by the intermediate party to $i_{x_k'}$, and n denotes a quantity of the plurality of participants.

[0095] Similarly, a result of performing weighted summation on the plurality of second gradients $\{\nabla f_k'\}$ can be used as the global gradient $\nabla f_{k+1}^{(G)}$ obtained through updating in the current round. For example, weighted summation of the gradients can be performed based on the following formula:

$$\nabla f_{k+1}^{(G)} = \sum_{i=1}^n \gamma_i * i_{\nabla f_k'} \quad (8)$$

[0096] Here, i in $i_{\nabla f_k'}$ explicitly denotes the i^{th} participant, γ_i denotes a weight allocated by the intermediate party to $i_{\nabla f_k'}$, and n denotes the quantity of the plurality of participants. It should be understood that, γ_i and β_i can be the same or can be different.

[0097] In addition, in an embodiment, the global model parameter $x_{k+1}^{(G)}$ obtained through aggregation can be further updated to satisfy the global constraint condition. It should be noted that for descriptions of the global constraint condition, references can be made to related descriptions in the above-mentioned embodiments; and for descriptions of updating the global model parameter $x_{k+1}^{(G)}$ based on the global constraint condition, references can be made to related descriptions of updating the local model parameter by the Pi party based on the local constraint condition in the above-mentioned embodiments. Details are omitted for simplicity. In this way, a trained model can satisfy a global constraint of a service.

[0098] The global approximation matrix $B_{k+1}^{(G)}$ obtained through updating in the current round can be further determined based on the global model parameter $x_{k+1}^{(G)}$ and the global gradient $g_{k+1}^{(G)}$ that are obtained above. The global approximation matrix $B_{k+1}^{(G)}$ may be updated with reference to an approximation matrix updating manner in the BFGS algorithm. A difference mainly lies in that a conventional BFGS algorithm is used for centralized machine learning rather than distributed machine learning, and no distinction is performed between local and global concepts. For example, a global approximation matrix approximation matrix $B_k^{(G)}$ can be updated to the global approximation matrix $B_{k+1}^{(G)}$ based on Formula (9).

$$B_{k+1}^{(G)} = B_k^{(G)} + \frac{S * S^T}{S^T * Y} - \left(\frac{B_k^{(G)} * Y * Y^T * B_k^{(G)}}{Y^T * B_k^{(G)} * Y} \right) \quad (9)$$

[0099] Here, $S = x_{k+1}^{(G)} - x_k^{(G)}$ represents a step difference, $Y = g_{k+1}^{(G)} - g_k^{(G)}$ represents a global gradient difference, and a superscript T represents a transpose operator. It should be understood that, for $x_k^{(G)}$, $g_k^{(G)}$, and $B_k^{(G)}$, when the current round of iteration is not the first round, that is, $k > 0$, the three pieces of data are obtained through a previous round of iteration. When the current round of iteration is the first round, that is, $k = 0$, the three pieces of data can be

obtained through initialization. For example, the intermediate party can separately receive, from the plurality of participants, x_0 and B_0 that are generated through local initialization, to respectively and correspondingly aggregate $\{x_0\}$ and $\{B_0\}$ into $x_0^{(G)}$ and $B_0^{(G)}$, and directly initialize $g_0^{(G)}$ into a zero vector.

5 **[0100]** It can be learned, by observing Formula (9), that the intermediate party needs to store $B_k^{(G)}$ to calculate $B_{k+1}^{(G)}$. If a quantity of dimensions of a parameter of the global model is 10 million, a quantity of dimensions of the matrix $B_k^{(G)}$ is 10 million * 10 million. Consequently, huge memory is consumed. Therefore, it is proposed that the idea of the L-BFGS algorithm can be further referred to. It is observed that $B_{k+1}^{(G)}$ is actually obtained through iteration based on
 10 $B_0^{(G)}$ and k groups of S and Y . $B_0^{(G)}$ can usually be directly initialized into a unit matrix. Therefore, memory space occupied by the matrix $B_k^{(G)}$ is very small. Therefore, only $B_1^{(G)}$ and k groups of S and Y can be stored. For example, when $B_{10}^{(G)}$ needs to be calculated, $B_{10}^{(G)}$ only needs to be calculated based on the unit matrix and S_1 and Y_1 to S_{10} and Y_{10} that are stored. In such a manner of exchanging time for space, when the parameter of the global model has 10
 15 million dimensions, storage of the 10 million * 10 million matrix changes to storage of 20 vectors with 10 million dimensions, to effectively save memory space.

[0101] In addition, when a quantity of memory iterations is very large, some data that cannot be stored can be discarded. It is assumed that a specified quantity of stored vectors is 100 groups S and Y . When the quantity of iterations exceeds 100, the 1st group of S and Y is discarded, and
 20 the forefront group of S and Y is discarded every time one more iteration is performed. In this way, although some precision is lost, availability is high enough.

[0102] Based on the above-mentioned descriptions, the intermediate party can determine, based on parameters and gradients provided by the plurality of participants in the current round, the global model parameter $x_{k+1}^{(G)}$ and the global approximation matrix $B_{k+1}^{(G)}$ that are obtained
 25 through updating in the current round.

[0103] Then, in step S206, the intermediate party separately determines, the global model parameter $x_{k+1}^{(G)}$ and the global approximation matrix $B_{k+1}^{(G)}$ that are obtained through updating in the current round, a target parameter $x^{(t)}$ and a target matrix $B^{(t)}$ that correspond to each participant (Pi party).

30 **[0104]** It should be understood that for each participant, the intermediate party can determine the target parameter $x^{(t)}$ and the target matrix $B^{(t)}$ based on a correspondence between parameter variables of the global model and a local model of the participant.

[0105] In an embodiment, it is assumed that federated learning is vertical federated learning.

In this case, for each participant, the intermediate party can extract, based on a correspondence between parameter variables of the global model and a local model locally deployed by the participant, a parameter part corresponding to the local model from the global model parameter $x_{k+1}^{(G)}$ obtained through updating in the current round, and use the parameter part as the target parameter $x^{(t)}$.

[0106] Similarly, for each participant, the intermediate party can extract a submatrix corresponding to the local model of the participant from the global approximation matrix $B_{k+1}^{(G)}$, and use the submatrix as the target matrix $B^{(t)}$.

[0107] In another embodiment, it is assumed that federated learning is horizontal federated learning, and the copy of the global model is locally deployed by each participant. In this case, the intermediate party can directly determine the global model parameter $x_{k+1}^{(G)}$ as the target parameter $x^{(t)}$, and determine the global approximation matrix $B_{k+1}^{(G)}$ as the target matrix $B^{(t)}$.

[0108] Based on the above-mentioned descriptions, the intermediate party can determine the corresponding target parameter $x^{(t)}$ and the corresponding target matrix $B^{(t)}$ for each of the plurality of participants. Therefore, in step S207, the intermediate party can provide each group of a target parameter and a target matrix to a corresponding participant.

[0109] Then, in step S208, each participant (Pi party) correspondingly updates the local model parameter and the local approximation matrix to the target parameter $x^{(t)}$ and the target matrix $B^{(t)}$ that are provided by the intermediate party, to perform a next round (the $(k + 2)^{\text{th}}$ round) of iterative training. For example, execution of this step can be denoted as follows: the local model parameter $x_{k+1} = x^{(t)}$ and the local approximation matrix $B_{k+1} = B^{(t)}$.

[0110] The above-mentioned interaction steps shown in FIG. 2 are performed, to implement any round of iteration and updating in the federated learning scheme disclosed in this embodiment of this specification can be implemented. It is known by a person skilled in the art that training of the model usually continues until the model converges. Correspondingly, in each round of iteration and updating, after obtaining, through aggregation, the global model parameter obtained through updating in the current round, the intermediate party can determine whether the global model parameter satisfies a convergence condition (for example, a maximum quantity of iterations is reached or the target function converges). An iteration stops once the convergence condition is satisfied. Otherwise, the updated model parameter is sent to the participant, to continue a new round of iteration and updating.

[0111] In conclusion, in the federated learning scheme provided in this embodiment of this specification, the BFGS system algorithm is updated, so that the BFGS system algorithm is applicable to a federated learning framework. In this way, in federated learning, data privacy of

each of the plurality of participants can be better protected, a data transmission amount is reduced, and model optimization efficiency is improved.

[0112] Corresponding to the above-mentioned federated learning method, an embodiment of this specification further discloses a federated learning apparatus. FIG. 3 is a schematic diagram of a functional structure of a federated training apparatus integrated into any participant according to an embodiment of this specification. The apparatus 300 performs any round of iterative training by using the following modules:

a first gradient calculation module 310, configured to calculate a first gradient corresponding to a current first parameter of a local model based on a batch of local training data corresponding to the current round; a target vector calculation module 320, configured to calculate, based on the first gradient and a current local approximation matrix, a target vector indicating a parameter updating direction, where the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; a first parameter updating module 330, configured to: update the first parameter to a second parameter based on the target vector and an obtained step, and calculate a second gradient corresponding to the second parameter; an intermediate data transmission module 340, configured to send the second parameter and the second gradient to the intermediate party; a target data receiving module 350, configured to receive a target parameter and a target matrix from the intermediate party, where the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of participants; and a local updating module 360, configured to correspondingly update a local model parameter and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.

[0113] In an embodiment, the current round is a first round, and the first parameter and the local approximation matrix are obtained through local initialization.

[0114] In an embodiment, the first gradient is further calculated based on a target function for optimizing the local model. The apparatus 300 further includes a step obtaining module 370, configured to: determine, based on the target vector through a line search, a step that satisfies a local constraint condition, so that a descending amplitude of the target function on the batch of local training data is greater than a predetermined threshold, and the local constraint condition is preset based on a service requirement.

[0115] In an embodiment, the first parameter updating module 330 is specifically configured to: calculate the sum of the first parameter and the product of the target vector and the step as a third parameter; and update the third parameter to the second parameter that satisfies a local constraint condition, where the local constraint condition is preset based on a service requirement.

- [0116] Further, in a specific embodiment, the local constraint condition includes an inequation constraint condition; and the first parameter updating module 330 is further configured to map, to a constraint boundary in a projection method, a component that is in the third parameter and that is outside a constraint range corresponding to the inequation constraint condition.
- 5 [0117] In another specific embodiment, the first gradient is further calculated based on a target function for optimizing the local model, and the local constraint condition includes an equation constraint condition; and the first parameter updating module 330 is further configured to update the third parameter to the second parameter by solving a Lagrange function constructed based on the target function and the equation constraint condition.
- 10 [0118] In an embodiment, the intermediate data transmission module 340 is specifically configured to send the second parameter and/or the second gradient to the intermediate party after differential privacy noise is added.
- [0119] In an embodiment, batches of local training data respectively used by the plurality of participants in the current round correspond to different feature parts of the same sample, and a
15 local model deployed by each of the plurality of participants is a local model of a global model.
- [0120] FIG. 4 is a schematic diagram of a functional structure of a federated training apparatus integrated into an intermediate party according to an embodiment of this specification. The intermediate party cooperates with a plurality of participants to perform federated training. The apparatus 400 performs any round of iterative training by using the following modules:
- 20 an intermediate data receiving module 410, configured to receive a current second parameter and a corresponding second gradient of a local model of any participant from the participant, where the second parameter is obtained by updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the first gradient is a gradient of the first parameter on a batch of local
25 training data corresponding to the current round, and the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter; a global data updating module 420, configured to determine, based on second parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are obtained through updating in the current round; and a
30 target data sending module 430, configured to provide a target parameter and a target matrix that correspond to any participant to the participant based on the global model parameter and the global approximation matrix that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to perform a next round of iterative training.
- 35 [0121] In an embodiment, the global data updating module 420 includes: a global parameter

aggregation unit 421, configured to: aggregate the second parameters respectively provided by the plurality of participants into a parameter corresponding to a global model, and use the parameter as the global model parameter obtained through updating in the current round; a global gradient aggregation unit 422, configured to: aggregate second gradients respectively provided by the plurality of participants into a parameter gradient corresponding to the global model, and use the parameter gradient as a global gradient obtained through updating in the current round; and a global matrix updating unit 423, configured to determine, based on the global model parameter obtained through updating in the current round, a global model parameter obtained through updating in a previous round, the global gradient obtained through updating in the current round, and a global gradient obtained through updating in the previous round, the global approximation matrix obtained through updating in the current round.

[0122] In a specific embodiment, the global parameter aggregation unit 421 is specifically configured to: after the parameter obtained through aggregation is updated to satisfy a global constraint condition, use the parameter as the global model parameter obtained through updating in the current round. The global constraint condition is preset based on a service requirement.

[0123] In a specific embodiment, the global model is configured to perform a user-oriented recommendation task, and the global constraint condition includes at least one of the following: a budget constraint, an exposure diversity constraint, or a non-negative constraint of the parameter.

[0124] In a specific embodiment, the second parameter satisfies a local constraint condition of a participant corresponding to the second parameter, and the local constraint condition is adaptively set based on the global constraint condition, the local model, and the local training data.

[0125] It should be understood that for ease of description, the above-mentioned apparatus is described by dividing the apparatus into various modules based on functions. Certainly, during implementation of this specification, functions of all modules can be implemented in one or more pieces of software and/or hardware.

[0126] The specification also provides a computer-readable non-transitory storage medium, wherein the storage medium stores a computer program, which, when executed by a processor, can be used to perform one or more steps of one or more methods described or illustrated herein or provides functionality described or illustrated herein. Herein, the computer-readable non-transitory storage medium or media may include one or more semiconductor-based or other integrated circuits (ICs) (such, as for example, field-programmable gate arrays (FPGAs) or application-specific ICs (ASICs)), hard disk drives (HDDs), hybrid hard drives (HHDs), optical discs, optical disc drives (ODDs), magneto-optical discs, magneto-optical drives, floppy diskettes, floppy disk drives (FDDs), magnetic tapes, solid-state drives (SSDs), RAM-drives, SECURE DIGITAL cards or drives, any other suitable computer-readable non-transitory storage media, or

any suitable combination of two or more of these, where appropriate. A computer-readable non-transitory storage medium may be volatile, non-volatile, or a combination of volatile and non-volatile, where appropriate.

[0127] The specification also provides a computer system. FIG. 5 illustrates an example
5 computer system 500. In particular embodiments, one or more computer systems 500 perform one or more steps of one or more methods described or illustrated herein. In particular embodiments, one or more computer systems 500 provide functionality described or illustrated herein. In particular embodiments, software running on one or more computer systems 500 performs one or more steps of one or more methods described or illustrated herein or provides functionality
10 described or illustrated herein. Particular embodiments include one or more portions of one or more computer systems 500. Herein, reference to a computer system may encompass a computing device, and vice versa, where appropriate. Moreover, reference to a computer system may encompass one or more computer systems, where appropriate.

[0128] This disclosure contemplates any suitable number of computer systems 500. This
15 disclosure contemplates computer system 500 taking any suitable physical form. As example and not by way of limitation, computer system 500 may be an embedded computer system, a system-on-chip (SOC), a single-board computer system (SBC) (such as, for example, a computer-on-module (COM) or system-on-module (SOM)), a desktop computer system, a laptop or notebook computer system, an interactive kiosk, a mainframe, a mesh of computer systems, a mobile telephone, a personal digital assistant (PDA), a server, a tablet computer system, or a combination
20 of two or more of these. Where appropriate, computer system 500 may include one or more computer systems 500; be unitary or distributed; span multiple locations; span multiple machines; span multiple data centers; or reside in a cloud, which may include one or more cloud components in one or more networks. Where appropriate, one or more computer systems 500 may perform
25 without substantial spatial or temporal limitation one or more steps of one or more methods described or illustrated herein. As an example and not by way of limitation, one or more computer systems 500 may perform in real time or in batch mode one or more steps of one or more methods described or illustrated herein. One or more computer systems 500 may perform at different times or at different locations one or more steps of one or more methods described or illustrated herein,
30 where appropriate.

[0129] In particular embodiments, computer system 500 includes a processor 502, memory 504, storage 506, an input/output (I/O) interface 508, a communication interface 510, and a bus 512. Although this disclosure describes and illustrates a particular computer system having a particular number of particular components in a particular arrangement, this disclosure
35 contemplates any suitable computer system having any suitable number of any suitable

components in any suitable arrangement.

[0130] In particular embodiments, processor 502 includes hardware for executing instructions, such as those making up a computer program. As an example and not by way of limitation, to execute instructions, processor 502 may retrieve (or fetch) the instructions from an internal register, an internal cache, memory 504, or storage 506; decode and execute them; and then write one or more results to an internal register, an internal cache, memory 504, or storage 506. In particular embodiments, processor 502 may include one or more internal caches for data, instructions, or addresses. This disclosure contemplates processor 502 including any suitable number of any suitable internal caches, where appropriate. As an example and not by way of limitation, processor 502 may include one or more instruction caches, one or more data caches, and one or more translation lookaside buffers (TLBs). Instructions in the instruction caches may be copies of instructions in memory 504 or storage 506, and the instruction caches may speed up retrieval of those instructions by processor 502. Data in the data caches may be copies of data of 50 in memory 504 or storage 506 for instructions executing at processor 502 to operate on; the results of previous instructions executed at processor 502 for access by subsequent instructions executing at processor 502 or for writing to memory 504 or storage 506; or other suitable data. The data caches may speed up read or write operations by processor 502. The TLBs may speed up virtual-address translation for processor 502. In particular embodiments, processor 502 may include one or more internal registers for data, instructions, or addresses. This disclosure contemplates processor 502 including any suitable number of any suitable internal registers, where appropriate. Where appropriate, processor 502 may include one or more arithmetic logic units (ALUs); be a multi-core processor; or include one or more processors 502. Although this disclosure describes and illustrates a particular processor, this disclosure contemplates any suitable processor.

[0131] In particular embodiments, memory 504 includes main memory for storing instructions for processor 502 to execute or data for processor 502 to operate on. As an example and not by way of limitation, computer system 500 may load instructions from storage 506 or another source (such as, for example, another computer system 500) to memory 504. Processor 502 may then load the instructions from memory 504 to an internal register or internal cache. To execute the instructions, processor 502 may retrieve the instructions from the internal register or internal cache and decode them. During or after execution of the instructions, processor 502 may write one or more results (which may be intermediate or final results) to the internal register or internal cache. Processor 502 may then write one or more of those results to memory 504. In particular embodiments, processor 502 executes only instructions in one or more internal registers or internal caches or in memory 504 (as opposed to storage 506 or elsewhere) and operates only on data in one or more internal registers or internal caches or in memory 504 (as opposed to storage 506 or

elsewhere). One or more memory buses (which may each include an address bus and a data bus) may couple processor 502 to memory 504. Bus 512 may include one or more memory buses, as described below. In particular embodiments, one or more memory management units (MMUs) reside between processor 502 and memory 504 and facilitate accesses to memory 504 requested by processor 502. In particular embodiments, memory 504 includes random access memory (RAM). This RAM may be volatile memory, where appropriate. Where appropriate, this RAM may be dynamic RAM (DRAM) or static RAM (SRAM). Moreover, where appropriate, this RAM may be single-ported or multi-ported RAM. This disclosure contemplates any suitable RAM. Memory 504 may include one or more memories 504, where appropriate. Although this disclosure describes and illustrates particular memory, this disclosure contemplates any suitable memory.

[0132] In particular embodiments, storage 506 includes mass storage for data or instructions. As an example and not by way of limitation, storage 506 may include a hard disk drive (HDD), a floppy disk drive, flash memory, an optical disc, a magneto-optical disc, magnetic tape, or a Universal Serial Bus (USB) drive or a combination of two or more of these. Storage 506 may include removable or non-removable (or fixed) media, where appropriate. Storage 506 may be internal or external to computer system 500, where appropriate. In particular embodiments, storage 506 is non-volatile, solid-state memory. In particular embodiments, storage 506 includes read-only memory (ROM). Where appropriate, this ROM may be mask-programmed ROM, programmable ROM (PROM), erasable PROM (EPROM), electrically erasable PROM (EEPROM), electrically alterable ROM (EAROM), or flash memory or a combination of two or more of these. This disclosure contemplates mass storage 506 taking any suitable physical form. Storage 506 may include one or more storage control units facilitating communication between processor 502 and storage 506, where appropriate. Where appropriate, storage 506 may include one or more storages 506. Although this disclosure describes and illustrates particular storage, this disclosure contemplates any suitable storage.

[0133] In particular embodiments, I/O interface 508 includes hardware, software, or both, providing one or more interfaces for communication between computer system 500 and one or more I/O devices. Computer system 500 may include one or more of these I/O devices, where appropriate. One or more of these I/O devices may enable communication between a person and computer system 500. As an example and not by way of limitation, an I/O device may include a keyboard, keypad, microphone, monitor, mouse, printer, scanner, speaker, still camera, stylus, tablet, touch screen, trackball, video camera, another suitable I/O device or a combination of two or more of these. An I/O device may include one or more sensors. This disclosure contemplates any suitable I/O devices and any suitable I/O interfaces 508 for them. Where appropriate, I/O interface 508 may include one or more device or software drivers enabling processor 502 to drive

one or more of these I/O devices. I/O interface 508 may include one or more I/O interfaces 508, where appropriate. Although this disclosure describes and illustrates a particular I/O interface, this disclosure contemplates any suitable I/O interface.

[0134] In particular embodiments, communication interface 510 includes hardware, software, or both providing one or more interfaces for communication (such as, for example, packet-based communication) between computer system 500 and one or more other computer systems 500 or one or more networks. As an example and not by way of limitation, communication interface 510 may include a network interface controller (NIC) or network adapter for communicating with an Ethernet or other wire-based network or a wireless NIC (WNIC) or wireless adapter for communicating with a wireless network, such as a WI-FI network. This disclosure contemplates any suitable network and any suitable communication interface 510 for it. As an example and not by way of limitation, computer system 500 may communicate with an ad hoc network, a personal area network (PAN), a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), or one or more portions of the Internet or a combination of two or more of these. One or more portions of one or more of these networks may be wired or wireless. As an example, computer system 500 may communicate with a wireless PAN (WPAN) (such as, for example, a BLUETOOTH WPAN), a WI-FI network, a WI-MAX network, a cellular telephone network (such as, for example, a Global System for Mobile Communications (GSM) network), or other suitable wireless network or a combination of two or more of these. Computer system 500 may include any suitable communication interface 510 for any of these networks, where appropriate. Communication interface 510 may include one or more communication interfaces 510, where appropriate. Although this disclosure describes and illustrates a particular communication interface, this disclosure contemplates any suitable communication interface.

[0135] In particular embodiments, bus 512 includes hardware, software, or both coupling components of computer system 500 to each other. As an example and not by way of limitation, bus 512 may include an Accelerated Graphics Port (AGP) or other graphics bus, an Enhanced Industry Standard Architecture (EISA) bus, a front-side bus (FSB), a HYPERTRANSPORT (HT) interconnect, an Industry Standard Architecture (ISA) bus, an INFINIBAND interconnect, a low-pin-count (LPC) bus, a memory bus, a Micro Channel Architecture (MCA) bus, a Peripheral Component Interconnect (PCI) bus, a PCI-Express (PCIe) bus, a serial advanced technology attachment (SATA) bus, a Video Electronics Standards Association local (VLB) bus, or another suitable bus or a combination of two or more of these. Bus 512 may include one or more buses 512, where appropriate. Although this disclosure describes and illustrates a particular bus, this disclosure contemplates any suitable bus or interconnect.

[0136] A person skilled in the art should be aware that in the above-mentioned one or more

examples, functions described in this application can be implemented by hardware, software, firmware, or any combination thereof. When the functions are implemented by software, the above-mentioned functions can be stored in a computer-readable medium or transmitted as one or more instructions or code in the computer-readable medium.

5 **[0137]** In the above-mentioned specific implementations, the objectives, technical solutions, and beneficial effects of the present invention are further described in detail. It should be understood that the above-mentioned descriptions are merely specific implementations of the present invention, but are not intended to limit the protection scope of the present invention. Any modification, equivalent replacement, improvement, etc. made based on the technical solutions of
10 the present invention shall fall within the protection scope of the present invention.

CLAIMS

What is claimed is:

1. A federated training method, involving an intermediate party and a plurality of participants, wherein the method is applied to any participant, and any round of iterative training involved in the method comprises:
 - calculating a first gradient corresponding to a current first parameter of a local model based on a batch of local training data corresponding to the current round;
 - calculating, based on the first gradient and a current local approximation matrix, a target vector indicating a parameter updating direction, wherein the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter;
 - updating the first parameter to a second parameter based on the target vector and an obtained step, and calculating a second gradient corresponding to the second parameter;
 - sending the second parameter and the second gradient to the intermediate party;
 - receiving a target parameter and a target matrix from the intermediate party, wherein the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of participants; and
 - correspondingly updating a local model parameter and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.
2. The method according to claim 1, wherein the current round is a first round, and the first parameter and the local approximation matrix are obtained through local initialization.
3. The method according to claim 1, wherein the first gradient is further calculated based on a target function for optimizing the local model, and obtaining the step comprises:
 - determining, based on the target vector through a line search, a step that satisfies a local constraint condition, so that a descending amplitude of the target function on the batch of local training data is greater than a predetermined threshold, and the local constraint condition is preset based on a service requirement.
4. The method according to claim 1, wherein the updating the first parameter to a second parameter based on the target vector and an obtained step comprises:
 - calculating the sum of the first parameter and the product of the target vector and the step as a third parameter; and
 - updating the third parameter to the second parameter that satisfies a local constraint

condition, wherein the local constraint condition is preset based on a service requirement.

5. The method according to claim 4, wherein the local constraint condition comprises an inequation constraint condition, and the updating the third parameter to the second parameter that satisfies a local constraint condition comprises:

5 mapping, to a constraint boundary in a projection method, a component that is in the third parameter and that is outside a constraint range corresponding to the inequation constraint condition.

6. The method according to claim 4, wherein the first gradient is further calculated based on a target function for optimizing the local model, the local constraint condition comprises
10 an equation constraint condition, and the updating the third parameter to the second parameter that satisfies a local constraint condition comprises:

updating the third parameter to the second parameter by solving a Lagrange function constructed based on the target function and the equation constraint condition.

7. The method according to claim 1, wherein before the sending the second parameter
15 and the second gradient to the intermediate party, the method further comprises:

adding differential privacy noise to the second parameter and/or the second gradient.

8. The method according to claim 1, wherein batches of local training data respectively used by the plurality of participants in the current round correspond to different feature parts of the same sample, and a local model deployed by each of the plurality of participants is a
20 local model of a global model.

9. A federated training method, involving an intermediate party and a plurality of participants, wherein the method is applied to the intermediate party, and any round of iterative training involved in the method comprises:

receiving a current second parameter and a corresponding second gradient of a local
25 model of any participant from the participant, wherein the second parameter is obtained by updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the first gradient is a gradient of the first parameter on a batch of local training data corresponding to the current round, and the local approximation matrix is used to approximate an inverse matrix of a
30 Hessian matrix corresponding to the first parameter;

determining, based on second parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are obtained through updating in the current round; and

providing a target parameter and a target matrix that correspond to any participant to
35 the participant based on the global model parameter and the global approximation matrix

that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to perform a next round of iterative training.

10. The method according to claim 9, wherein the determining, based on second
5 parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are obtained through updating in the current round comprises:

aggregating the second parameters respectively provided by the plurality of participants into a parameter corresponding to a global model, and using the parameter as the global
10 model parameter obtained through updating in the current round;

aggregating second gradients respectively provided by the plurality of participants into a parameter gradient corresponding to the global model, and using the parameter gradient as a global gradient obtained through updating in the current round; and

determining, based on the global model parameter obtained through updating in the
15 current round, a global model parameter obtained through updating in a previous round, the global gradient obtained through updating in the current round, and a global gradient obtained through updating in the previous round, the global approximation matrix obtained through updating in the current round.

11. The method according to claim 10, wherein the aggregating the second parameters
20 respectively provided by the plurality of participants into a parameter corresponding to a global model, and using the parameter as the global model parameter obtained through updating in the current round comprises:

after the parameter obtained through aggregation is updated to satisfy a global constraint condition, using the parameter as the global model parameter obtained through updating in
25 the current round, wherein the global constraint condition is preset based on a service requirement.

12. The method according to claim 11, wherein the global model is configured to perform a user-oriented recommendation task, and the global constraint condition comprises at least one of the following: a budget constraint, an exposure diversity constraint, or a non-
30 negative constraint of the parameter.

13. The method according to claim 11, wherein the second parameter satisfies a local constraint condition of a participant corresponding to the second parameter, and the local constraint condition is adaptively set based on the global constraint condition, the local model, and the local training data.

35 14. A federated training apparatus, involving an intermediate party and a plurality of

participants, wherein the apparatus is integrated into any participant, and the apparatus performs any round of iterative training by using the following modules:

a first gradient calculation module, configured to calculate a first gradient corresponding to a current first parameter of a local model based on a batch of local training data corresponding to the current round;

a target vector calculation module, configured to calculate, based on the first gradient and a current local approximation matrix, a target vector indicating a parameter updating direction, wherein the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter;

a first parameter updating module, configured to: update the first parameter to a second parameter based on the target vector and an obtained step, and calculate a second gradient corresponding to the second parameter;

an intermediate data transmission module, configured to send the second parameter and the second gradient to the intermediate party;

a target data receiving module, configured to receive a target parameter and a target matrix from the intermediate party, wherein the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of participants; and

a local updating module, configured to correspondingly update a local model parameter and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.

15. A federated training apparatus, involving an intermediate party and a plurality of participants, wherein the apparatus is integrated into the intermediate party, and the apparatus performs any round of iterative training by using the following modules:

an intermediate data receiving module, configured to receive a current second parameter and a corresponding second gradient of a local model of any participant from the participant, wherein the second parameter is obtained by updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the first gradient is a gradient of the first parameter on a batch of local training data corresponding to the current round, and the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter;

a global data updating module, configured to determine, based on second parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are obtained through updating in

the current round; and

a target data sending module, configured to provide a target parameter and a target matrix that correspond to any participant to the participant based on the global model parameter and the global approximation matrix that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to perform a next round of iterative training.

16. A computer system, comprising a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor, wherein the processor is integrated into any one of a plurality of participants, the plurality of participants cooperatively perform federated training with an intermediate party, and when the processor executes the program, the following steps are implemented:

calculating a first gradient corresponding to a current first parameter of a local model based on a batch of local training data corresponding to the current round;

calculating, based on the first gradient and a current local approximation matrix, a target vector indicating a parameter updating direction, wherein the local approximation matrix is used to approximate an inverse matrix of a Hessian matrix corresponding to the first parameter;

updating the first parameter to a second parameter based on the target vector and an obtained step, and calculating a second gradient corresponding to the second parameter;

sending the second parameter and the second gradient to the intermediate party;

receiving a target parameter and a target matrix from the intermediate party, wherein the target parameter and the target matrix are determined by the intermediate party by updating a global model parameter and a global approximation matrix based on second parameters and second gradients that are respectively provided by the plurality of participants; and

correspondingly updating a local model parameter and the local approximation matrix to the target parameter and the target matrix, to perform a next round of iterative training.

17. A computer system, comprising a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor, wherein the processor is integrated into an intermediate party, the intermediate party cooperatively perform federated training with a plurality of participants, and when the processor executes the program, the following steps are implemented:

receiving a current second parameter and a corresponding second gradient of a local model of any participant from the participant, wherein the second parameter is obtained by

updating a first parameter based on a target vector and a step, the target vector is calculated based on a first gradient and a current local approximation matrix, the first gradient is a gradient of the first parameter on a batch of local training data corresponding to the current round, and the local approximation matrix is used to approximate an inverse matrix of a

5 Hessian matrix corresponding to the first parameter;

determining, based on second parameters and second gradients that are respectively provided by the plurality of participants, a global model parameter and a global approximation matrix that are obtained through updating in the current round; and

10 providing a target parameter and a target matrix that correspond to any participant to the participant based on the global model parameter and the global approximation matrix that are obtained through updating in the current round, so that the participant correspondingly updates a local model parameter and a local approximation matrix based on the target parameter and the target matrix, to perform a next round of iterative training.

15 18. A computer-readable nonvolatile storage medium, wherein the storage medium stores a computer program, and when the computer program is executed by a processor, the method according to any one of claims 1 to 13 is implemented.