VERSION MANAGEMENT METHOD AND APPARATUS FOR LARGE LANGUAGE MODEL, DEVICE, AND MEDIUM

[0001] This application claims priority to Chinese Patent Application No. 202510193916.1 filed with the China National Intellectual Property Administration (CNIPA) on Feb. 21, 2025, the disclosure of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The present application relates to the technical field of artificial intelligence, for example, a version management method and apparatus for a large language model, a device, and a medium.

BACKGROUND

[0003] With the widespread penetration of large language models across various industries, the demand for the application of large language models in the field of car finance is growing increasingly and becoming more refined.

[0004] As an efficient and cost-effective adaptation approach, large model fine-tuning has become a popular application mode in this field by using basic models and combining the basic models with data from specific fields for targeted training. However, in the special scenario of car finance, large model fine-tuning is still confronted with many challenges, especially differences in version requirements caused by changes in characteristics such as a time node and a function. As a result, the complexity of version management is increased, and a version rollback operation becomes particularly difficult.

SUMMARY

[0005] The present application provides a version management method and apparatus for a large language model, a device, and a medium, which can greatly simplify the complexity of version management and improve the ability to perform a quick and accurate rollback for a specific error.

[0006] According to an aspect of the present application, a version management method for a large language model is provided. The method includes the steps below.

[0007] Version information and metadata that correspond to a large language model training task in a target tenant are acquired, where the version information includes timestamp information and region information, and the metadata includes performance indicator data.

[0008] A version rollback strategy of the large language model is determined according to the timestamp information, the region information, and the performance indicator data.

[0009] A version rollback operation is performed on the large language model according to the version rollback strategy to complete a version rollback of the large language model.

[0010] According to another aspect of the present application, a version management apparatus for a large language model is provided. The apparatus includes the modules below.

[0011] A data acquisition module is configured to acquire version information and metadata that correspond to a large language model training task in a target tenant, where the version information includes timestamp information and region information, and the metadata includes performance indicator data.

[0012] A strategy determination module is configured to determine a version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data.

[0013] A rollback operation execution module is configured to perform a version rollback operation on the large language model according to the version rollback strategy to complete a version rollback of the large language model.

[0014] According to another aspect of the present application, an electronic device is provided.

[0015] The electronic device includes at least one processor and a memory communicatively connected to the at least one processor.

[0016] The memory is configured to store a computer program executable by the at least one processor to cause the at least one processor to perform the version management method for a large language model of any embodiment of the present application.

[0017] According to another aspect of the present application, a computer-readable storage medium is provided. The computer-readable storage medium stores computer instructions that, when executed by a processor, are configured to cause the processor to perform the version management method for a large language model of any embodiment of the present application.

[0018] In the technical solution of embodiments of the present application, the version information and the metadata that correspond to the large language model training task in the target tenant are acquired, where the version information includes the timestamp information and the region information, and the metadata includes the performance indicator data; the version rollback strategy of the large language model is determined according to the timestamp information, the region information, and the performance indicator data; and the version rollback operation is performed on the large language model according to the version rollback strategy to complete the version rollback of the large language model. In the technical solution, the corresponding version rollback strategy can be determined according to the performance data of the large language model and the timestamp information of the large language model, thereby greatly simplifying the complexity of version management and improving the ability to perform a quick and accurate rollback for a specific error.

BRIEF DESCRIPTION OF DRAWINGS

[0019] FIG. 1 is a flowchart of a version management method for a large language model according to embodiment one of the present application.

[0020] FIG. 2 is a flowchart of a version management method for a large language model according to embodiment two of the present application.FIG. 3 is a diagram illustrating the structure of a version management apparatus for a large language model according to embodiment three of the present application.

[0022] FIG. 4 is a diagram illustrating the structure of an electronic device according to embodiment four of the present application.

DETAILED DESCRIPTION

[0023] For a better understanding of the solution of the present application by those skilled in the art, the technical solutions in embodiments of the present application are described clearly and completely hereinafter in conjunction with the drawings in the embodiments of the present application. Apparently, the embodiments described hereinafter are part, not all, of embodiments of the present application. Based on the embodiments of the present application, all other embodiments obtained by those of ordinary skill in the art on the premise that no creative work is done are within the scope of the present application.

[0024] It is to be noted that the terms "first", "second", and "target" in the specification, claims, and drawings of the present application are used for distinguishing between similar objects and are not necessarily used for describing a particular order or sequence. It is to be understood that data used in this manner are interchangeable where appropriate so that the embodiments of the present application described herein can be implemented in an order not illustrated or described herein. Additionally, terms "include", "have", and any variations thereof are intended to encompass a non-exclusive inclusion. For example, a process, method, system, product, or device that includes a series of steps or units not only includes the expressly listed steps or units but may also include other steps or units that are not expressly listed or are inherent to such process, method, product, or device.

[0025] Embodiment one

[0026] FIG. 1 is a flowchart of a version management method for a large language model

according to embodiment one of the present application. This embodiment is applicable to the case of managing a version of a large language model. The method may be executed by a version management apparatus for a large language model. The version management apparatus for a large language model may be implemented in the form of hardware and/or software. The version management apparatus for a large language model may be configured in an electronic device having a data processing capability. As shown in FIG. 1, the method includes the steps below.In S110, version information and metadata that correspond to a large language model training task in a target tenant are acquired.

[0028] The version information includes timestamp information and region information, and the metadata includes performance indicator data. The timestamp information may include the training start time, end time, and deployment time of a version corresponding to each large language model training task. The version information typically refers to model version information used by the large language model training task. The metadata may refer to data of the large language model training task, which may specifically include training data, algorithm parameters, performance indicators, and other data. The region information may refer to a geographical region or market information that is used by the large language model. It is to be understood that in this embodiment, there are differences in service requirements for different regions, so large language models corresponding to the different regions are also different. The large language model refers to a deep learning model trained using a large amount of text data so that the model can generate natural language text or understand the meaning of language text. The large language model in this embodiment refers to an advanced model that provides customized natural language processing services for tenants in a multi-tenant service environment.

[0029] The target tenant may be any tenant on a training platform. In this embodiment, tenant-specific resource allocation management and permission management are further refined for each tenant to achieve efficient resource utilization and resource isolation between tenants. In this embodiment, an enhanced resource isolation manner is designed for

each tenant on the training platform.

[0030] Regarding the resource isolation part, in this embodiment, targeted optimization and innovation have been carried out according to the powerful container orchestration and resource management capabilities of Kubernetes (K8s). Specifically, this embodiment is targeted at the establishment of a tenant database on the training platform and the isolation at the database level. First, after new tenants complete the information entry through a registration system on the training platform, the platform automatically triggers database creation processes and generates an independent database instance for each new tenant. These database instances have achieved strict database-level isolation at both the physical and logical levels to ensure the security of tenant data. Subsequently, a role-based access control (RBAC) mechanism is introduced to further refine the permission management between tenants. According to the service needs of the tenants and the division of responsibilities of the tenants, multiple roles such as an administrator, a developer, and a tester are defined, and a clear permission boundary is assigned to each role. These permission boundaries are implemented through the RBAC mechanism of K8s, that is, fine permission management is implemented by defining Role and RoleBinding. In terms of implementation details, precise control of permissions is achieved by ensuring that a role definition is detailed to a specific application programming interface (API) resource, a specific operation type, and a specific scope. Moreover, the use of permissions is also regularly audited to promptly detect and correct any abnormality or abuse of permissions so as to ensure that operations between the tenants do not interfere with each other. In terms of resource allocation and management, a custom resource type "TenantResource" is defined by using the Custom Resource Definition (CRD) function of K8s. This innovative measure can allow a dedicated resource quota to be configured for each "TenantResource" according to the specific needs of a tenant, including key resources such as a central processing unit (CPU), a memory, and storage. In terms of implementation details, the definition of "TenantResource" includes key information such as a tenant identifier, a resource type, and a resource quota to ensure the accuracy and rationality of resource allocation. Moreover, the

configuration of the resource quota is dynamically adjusted according to the service scale, growth trend, and peak demand of the tenant to satisfy the changing service needs of the tenant. Additionally, a suitable scheduling strategy is also formulated to comprehensively consider the availability of resources, the priority of the tenants, and affinity/anti-affinity rules to achieve efficient resource utilization and isolation between the tenants. Finally, network security is also given great attention. An independent virtual network is created for each tenant by selecting a K8s network plug-in (such as Calico or Cilium) that supports virtual network isolation. These virtual networks are logically isolated from each other, effectively preventing network attacks and data leakage risks between the tenants. In terms of implementation details, the creation process of the virtual network is carefully planned according to the service needs of the tenant and a network topological structure, so that the configuration of a network policy is detailed to a specific network port, a specific protocol, and a specific IP address range is ensured. Moreover, network traffic and abnormal behaviors are also regularly monitored to promptly detect and handle any potential network security threats to ensure that the network environment of the tenant is safe and stable.

[0031] In this embodiment, physical resources corresponding to each tenant are different, and the large language model is deployed on the physical resources corresponding to each tenant. In this embodiment, detailed version information and metadata may be recorded for each large language model training task by using the Annotations function of K8s, including training data, algorithm parameters, performance indicators, and others. Specifically, when the version information and the metadata are recorded using the Annotations of K8s, the timestamp is added to each version, including the training start time, end time, and deployment time, which helps to track the performance of the model in different time periods and provide data support for a time-based rollback strategy. A region label is added to each version to indicate the geographical region or market to which the model is applicable, which helps to quickly locate when a model rollback is required for a specific region. Therefore, in this embodiment, when version management processing is performed for different tenants, the following data may be acquired: the detailed version information

corresponding to each large language model training task of the large language model of the target tenant, the training data, the algorithm parameters, and the performance indicators.

[0032] In S120, a version rollback strategy of the large language model is determined according to the timestamp information, the region information, and the performance indicator data.

[0033] The version rollback strategy may be considered the most appropriate model version determined when the large language model requires a rollback operation.

[0034] In this embodiment, model performance data of different versions in the large language model may be analyzed according to the timestamp information, the region information, and the performance indicator data, and a model version with the optimal model performance or the least problems may be selected as a rollback target; if there are multiple candidate versions, the best rollback version may also be determined through further testing or performance assessment to obtain the version rollback strategy of the large language model.

[0035] In this embodiment, when the large language model requires the rollback operation, the system not only locates the most appropriate rollback version according to the timestamp information, the region information, and the performance indicator data, but also automatically analyzes the cause of an error and generates a rollback recommendation report for user reference. Additionally, the concept of version chain is introduced in this embodiment, allowing a user to select a specific version chain for rollback when the user performs a rollback, thereby avoiding the impact of erroneous data or algorithms on subsequent versions. In a special service scenario, such as when a credit assessment model is updated, a time window-based rollback strategy is also designed to ensure that the smooth rollback of the model is completed without affecting the service continuity.

[0036] In S130, a version rollback operation is performed on the large language model according to the version rollback strategy to complete a version rollback of the large

language model.

[0037] In this embodiment, after the best model version in the version rollback strategy is determined, the large language model service of the current version in the large language model may be stopped first to avoid data conflicts or inconsistencies; then the version rollback operation is performed on the large language model according to the version rollback strategy, so that the large language model is restored to the specified best model version. Moreover, whether model files and data after the rollback are complete and correct is verified, thereby completing the smooth rollback of the large language model and ensuring the model performance of the large language model without affecting the service continuity.

[0038] In the technical solution of the embodiment of the present application, the version information and the metadata that correspond to the large language model training task in the target tenant are acquired, where the version information includes the timestamp information and the region information, and the metadata includes the performance indicator data; the version rollback strategy of the large language model is determined according to the timestamp information, the region information, and the performance indicator data; and the version rollback operation is performed on the large language model according to the version rollback strategy to complete the version rollback of the large language model. In the technical solution, the corresponding version rollback strategy can be determined according to the performance data of the large language model and the timestamp information of the large language model, thereby greatly simplifying the complexity of version management and improving the ability to perform a quick and accurate rollback for a specific error.

[0039] Embodiment two

[0040] FIG. 2 is a flowchart of a version management method for a large language model according to embodiment two of the present application. This embodiment is optimized

based on the preceding embodiment. Specifically, determining the version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data includes determining target performance data of versions in the large language model according to the timestamp information and the performance indicator data; and determining the version rollback strategy of the large language model according to the target performance data of the versions and/or the region information. As shown in FIG. 2, the method includes the steps below.

[0041] In S210, the version information and the metadata that correspond to the large language model training task in the target tenant are acquired.

[0042] The version information includes the timestamp information and the region information, and the metadata includes the performance indicator data.

[0043] In S220, target performance data of versions in the large language model is determined according to the timestamp information and the performance indicator data.

[0044] The target performance data may be considered as specified performance data corresponding to the versions in the large language model. Exemplarily, the target performance data may be performance data such as service continuity and credit assessment accuracy. In this embodiment, the corresponding target performance data may be set according to the actual needs of the user.

[0045] In this embodiment, credit assessment impact results of different versions in the large language model may be assessed through the performance indicator data, and the changes in different credit assessment results within a time window may be determined in combination with the timestamp information, thereby determining the impact results of rolling back to the different versions on the model performance data.

[0046] In this embodiment, optionally, determining the target performance data of the versions in the large language model according to the timestamp information and the performance indicator data includes: assessing credit assessment impact results of different

versions in the large language model according to the performance indicator data; and determining the target performance data of the large language model according to the credit assessment impact results and the timestamp information.

[0047] The credit assessment impact results may refer to the expected change results of the credit assessment results corresponding to the different versions. In this embodiment, the credit assessment impact results of the different versions may be determined through the performance indicator data and a machine learning algorithm prediction model. In this embodiment, the performance indicator data corresponding to the versions and a machine learning algorithm may be used for obtaining the credit assessment impact results of the versions, the changes in the credit assessment impact results of the large language model in time periods are analyzed in combination with the timestamp information, and target performance data of each version in the large language model is determined according to comparison results of a time sequence analysis and the versions.

[0048] In this embodiment, through this configuration, the target performance data of the versions in the large language model can be analyzed in combination with the timestamp information and the performance indicator information, so as to determine the best rollback version of the large language model through analysis and comparison.

[0049] In this embodiment, optionally, the performance indicator data includes accuracy indicator data, recall indicator data, and reconciliation indicator data; and assessing the credit assessment impact results of the versions in the large language model according to the performance indicator data includes: inputting the accuracy indicator data, the recall indicator data, and the reconciliation indicator data that correspond to the versions into a credit assessment prediction model to obtain the credit assessment impact results of the versions.

[0050] The accuracy indicator data may be considered accuracy data of the large language model and may be an indicator to assess the accuracy of a model processing result. The

recall indicator data may be considered recall data of the large language model, may also be referred to as the recall ratio, and is the proportion of instances correctly identified as the positive class (the true class) by the model to all the actual positive class instances. The reconciliation indicator data may refer to an F1 score, which is expressed as the reconciled mean of the accuracy and the recall ratio. The reconciliation indicator data takes into account two indicators to reduce the misleading nature brought by a single indicator.

[0051] In this embodiment, the metadata includes key performance indicator data of credit assessment, such as the accuracy, the recall ratio, the F1 score, and other indicator data, as well as the changing trends of these indicators over time, so as to assess the performance of the large language model in a credit assessment task and provide a basis for a rollback decision.

[0052] In this embodiment, accuracy, recall ratios, F1 scores, and other indicator data that correspond to the versions may be input into the trained credit assessment prediction model to obtain the credit assessment impact results corresponding to the versions.

[0053] Further, in this embodiment, the time window-based rollback strategy is designed for the credit assessment prediction model. For example, before the service peak or within a specific time period, the system automatically assesses the performance of the current large language model and decides, according to a preset threshold, whether to perform a rollback. When the rollback operation is performed, the most recent version that performs stably within the time window and has good credit assessment performance may be selected for deployment.

[0054] In this embodiment, through this configuration, historical data and a machine learning algorithm may be used for predicting the expected changes in credit assessment results after rolling back to the different versions, helping make a more informed rollback decision and improving the reliability of the rollback operation.

[0055] In S230, the version rollback strategy of the large language model is determined

according to the target performance data of the versions and/or the region information.

[0056] In this embodiment, in the version chain, the linear relationship of the versions is recorded, and the regional applicability of each version and the target performance data of each version are marked. The version rollback strategy of the large language model can be determined according to the target performance data of the versions, or the region information, or the target performance data of the versions and the region information. In this embodiment, the version rollback strategy may include a model version to be rolled back and a specific rollback manner.

[0057] Specifically, in this embodiment, when the rollback operation is performed, a version chain that has the least impact on a specific region or credit assessment performance may be selected for the rollback operation. In the process of determining the version rollback strategy according to the region information, if an error or performance degradation only affects a certain specific region, the system recommends a rollback only in a model in that region to maintain the stability and accuracy of models in other regions.

[0058] In this embodiment, optionally, determining the version rollback strategy of the large language model according to the target performance data of the versions and/or the region information includes generating a rollback recommendation report of the large language model according to the target performance data of the versions and/or the region information; and determining the version rollback strategy of the large language model according to the rollback recommendation report.

[0059] The rollback recommendation report may include multiple rollback versions, rollback manners, and risk assessments. In this embodiment, one or more historical versions may be selected as rollback targets according to the target performance data and the region information, and a specific rollback manner is determined according to the actual needs, such as directly rolling back to a specified version or fine-tuning based on a specific version. The direct rollback may involve the operation of a version control system, while the

fine-tuning requires adjustment and optimization based on a selected version. Subsequently, risks that may be brought about when the rollback versions are executed are assessed, such as performance degradation and data loss. These risks should be identified and mitigated through sufficient testing. In this embodiment, the rollback recommendation report of the large language model is generated by combining the rollback targets, the rollback manners, and the risk assessments, and the most suitable rollback version and the most suitable rollback manner are determined as the version rollback strategy according to the rollback recommendation report and the actual needs of the user.

[0060] Additionally, in this embodiment, when the rollback recommendation report is generated, the system not only analyzes the cause of the error, but also assesses the impacts of rolling back to the different versions on the service continuity and credit assessment accuracy in combination with the changes in credit assessment indicators within the time window.

[0061] In this embodiment, through this configuration, the most suitable version rollback strategy for the large language model is determined according to the target performance data of the versions and region data, which can ensure the stability and performance of the large language model in different tasks and regions.

[0062] In S240, the version rollback operation is performed on the large language model according to the version rollback strategy to complete the version rollback of the large language model.

[0063] In this embodiment, optionally, performing the version rollback operation on the large language model according to the version rollback strategy to complete the version rollback of the large language model includes: performing the version rollback operation within a first rollback range of the large language model according to the version rollback strategy to obtain model performance data after the version rollback; and continuing performing the version rollback operation on the large language model according to the

model performance data after the version rollback to complete the version rollback of the large language model.

[0064] The first rollback range may be a small range in the environment deployed by the large language model. In this embodiment, the first rollback range may be set according to the actual needs.

[0065] In this embodiment, for a large language model deployed on a large scale, the rollback operation is performed on the large language model in a gradual rollback manner. Rollback testing may be performed within a small range first to ensure that the model performance after the rollback is stable and satisfies expectations, and the rollback range may be gradually expanded to complete the version rollback of the large language model. Exemplarily, the large language model deployed on the large scale may be considered a large-scale application of the currently deployed large model. For example, dozens of services deployed on an online server all use large language models.

[0066] In this embodiment, the version rollback operation may be performed within the preset first rollback range of the large language model according to the determined version rollback strategy to obtain the performance data after the version rollback within the first rollback range; whether the expected effect is satisfied is determined according to the model performance data of the large language model within the first rollback range after the rollback; the rollback range is gradually expanded to continue performing the version rollback operation on the large language model, thereby gradually enabling the version rollback of the large language model.

[0067] In this embodiment, through this configuration, the rollback operation can be performed on the large language model deployed on the large scale in the gradual rollback manner, further ensuring the performance and rollback effect of the large language model.

[0068] In this embodiment, optionally, continuing performing the version rollback operation on the large language model according to the model performance data after the

version rollback to complete the version rollback of the large language model includes, in response to the model performance data after the version rollback satisfying a set condition, gradually continuing performing the version rollback operation within a second rollback range until the version rollback operation is completed within the entire range of the large language model.

[0069] The second rollback range is larger than the first rollback range. The set condition may be set for the model performance data and may be preset. In this embodiment, the set condition may be that the model performance data satisfies the expected data of the model, and the specific expected data may be configured according to the actual large language model.

[0070] In this embodiment, a gradual rollback strategy is adopted for the model deployed on the large scale. The rollback testing is first performed within the first rollback range to ensure that the model performance data after the version rollback satisfies the set submission, that is, the model performance after the version rollback is stable and satisfies the expectations, and the rollback range is gradually expanded to gradually continue performing the version rollback operation within the second rollback range of the large language model until the version rollback operation is completed within the entire range of the large language model; in response to the model performance data after the version rollback not satisfying the set condition, the version rollback operation on the large language model may be stopped, and the version rollback strategy may be adjusted according to the model performance.

[0071] In this embodiment, in the rollback process of the large language model, the performance of the model and the credit assessment indicators of the model are monitored in real time to promptly detect and handle any abnormality. Moreover, user feedback is collected to continuously assess and optimize the rollback effect.

[0072] In this embodiment, through this configuration, the gradual rollback manner can

be adopted for the large language model deployed on the large scale, which ensures the flexibility and reliability of the rollback of the large language model and improves the stability of the performance of the large language model.

[0073] In this embodiment, the version rollback process is further improved and optimized by combining the timeliness, regionalism, and credit assessment attributes of car finance, which not only improves the flexibility and accuracy of the rollback, but also ensures the service continuity and the accuracy of the credit assessment.

[0074] In the technical solution of the embodiment of the present application, the version information and the metadata that correspond to the large language model training task in the target tenant are acquired, where the version information includes the timestamp information and the region information, and the metadata includes the performance indicator data; the target performance data of the versions in the large language model is determined according to the timestamp information and the performance indicator data; the version rollback strategy of the large language model is determined according to the target performance data of the versions and/or the region information. The version rollback operation is performed on the large language model according to the version rollback strategy to complete the version rollback of the large language model. In the technical solution, the corresponding version rollback strategy can be determined according to the performance data of the large language model and the timestamp information of the large language model, thereby greatly simplifying the complexity of version management and improving the ability to perform a quick and accurate rollback for a specific error.

[0075] Embodiment three

[0076] FIG. 3 is a diagram illustrating the structure of a version management apparatus for a large language model according to embodiment three of the present application. As shown in FIG. 3, the apparatus includes the modules below.

[0077] A data acquisition module 310 is configured to acquire version information and

metadata that correspond to a large language model training task in a target tenant, where the version information includes timestamp information and region information, and the metadata includes performance indicator data.

[0078] A strategy determination module 320 is configured to determine a version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data.

[0079] A rollback operation execution module 330 is configured to perform a version rollback operation on the large language model according to the version rollback strategy to complete a version rollback of the large language model.

[0080] Optionally, the strategy determination module 320 includes the units below.

[0081] A target performance data determination unit is configured to determine target performance data of versions in the large language model according to the timestamp information and the performance indicator data.

[0082] A version rollback strategy determination unit is configured to determine the version rollback strategy of the large language model according to the target performance data of the versions and/or the region information.

[0083] Optionally, the target performance data determination unit includes the subunits below.

[0084] An assessment impact result subunit is configured to assess credit assessment impact results of the large language model in the versions according to the performance indicator data.

[0085] A performance data determination subunit is configured to determine the target performance data of the large language model according to the credit assessment impact results and the timestamp information.

[0086] Optionally, the performance indicator data includes accuracy indicator data, recall indicator data, and reconciliation indicator data.

[0087] The assessment impact result subunit is specifically configured to input the accuracy indicator data, the recall indicator data, and the reconciliation indicator data that correspond to the versions into a credit assessment prediction model to obtain the credit assessment impact results of the versions.

[0088] Optionally, the version rollback strategy determination unit is specifically configured to generate a rollback recommendation report of the large language model according to the target performance data of the versions and/or the region information and determine the version rollback strategy of the large language model according to the rollback recommendation report.

[0089] Optionally, the rollback operation execution module 330 includes the units below.

[0090] A first-range rollback processing unit is configured to perform the version rollback operation within a first rollback range of the large language model according to the version rollback strategy to obtain model performance data after the version rollback.

[0091] A rollback operation continuous execution unit is configured to continue performing the version rollback operation on the large language model according to the model performance data after the version rollback to complete the version rollback of the large language model.

[0092] Optionally, the rollback operation continuous execution unit is specifically configured to, in response to the model performance data after the version rollback satisfying a set condition, gradually continue performing the version rollback operation within a second rollback range until the version rollback operation is completed within the entire range of the large language model, where the second rollback range is larger than the first rollback range.

[0093] The version management apparatus for a large language model provided in this embodiment of the present application may perform the version management method for a large language model provided in any embodiment of the present application and has function modules and beneficial effects that correspond to the performed method.

[0094] Embodiment four

[0095] FIG. 4 is a diagram illustrating the structure of an electronic device according to embodiment four of the present application. The electronic device 10 is intended to represent various forms of digital computers, for example, a laptop computer, a desktop computer, a worktable, a personal digital assistant, a server, a blade server, a mainframe computer, and another applicable computer. The electronic device may also represent various forms of mobile apparatuses, for example, a personal digital assistant, a cellphone, a smartphone, a wearable device (such as a helmet, glasses, or a watch), and another similar computing apparatus. Herein the shown components, the connections and relationships between these components, and the functions of these components are illustrative and are not intended to limit the implementation of the present application as described and/or claimed herein.

[0096] As shown in FIG. 4, the electronic device 10 includes at least one processor 11 and a memory, such as a read-only memory (ROM) 12 or a random-access memory (RAM) 13, communicatively connected to the at least one processor 11. The memory stores a computer program executable by the at least one processor. The at least one processor 11 may perform various types of appropriate operations and processing according to a computer program stored in the read-only memory (ROM) 12 or a computer program loaded from a storage unit 18 to the random-access memory (RAM) 13. Various programs and data required for the operation of the electronic device 10 may also be stored in the RAM 13. The at least one processor 11, the ROM 12, and the RAM 13 are connected to each other through a bus 14. An input/output (I/O) interface 15 is also connected to the bus 14.

[0097] Multiple components in the electronic device 10 are connected to the I/O interface 15. The multiple components include an input unit 16 such as a keyboard or a mouse, an output unit 17 such as various types of display or speaker, the storage unit 18 such as a magnetic disk or an optical disk, and a communication unit 19 such as a network card, a modem, or a wireless communication transceiver. The communication unit 19 allows the electronic device 10 to exchange information/data with other devices over a computer network such as the Internet and/or various telecommunications networks.

[0098] The at least one processor 11 may be various general-purpose and/or special-purpose processing components having processing and computing capabilities. Examples of a processor 11 include, but are not limited to, a central processing unit (CPU), a graphics processing unit (GPU), a special-purpose artificial intelligence (AI) computing chip, a processor executing machine learning models and algorithms, a digital signal processor (DSP), and any appropriate processor, controller, and microcontroller. The at least one processor 11 performs various preceding methods and processing, such as the version management method for a large language model.

[0099] In some embodiments, the version management method for a large language model may be implemented as computer programs tangibly contained in a computer-readable storage medium such as the storage unit 18. In some embodiments, part or all of the computer programs may be loaded and/or installed onto the electronic device 10 via the ROM 12 and/or the communication unit 19. When the computer programs are loaded to the RAM 13 and executed by the at least one processor 11, one or more steps of the preceding version management method for a large language model may be performed. Alternatively, in other embodiments, the at least one processor 11 may be configured, in any other appropriate manner (for example, by means of firmware), to perform the version management method for a large language model.

[0100] Herein various embodiments of the preceding systems and techniques may be implemented in digital electronic circuitry, integrated circuitry, field-programmable gate

arrays (FPGAs), application-specific integrated circuits (ASICs), application-specific standard products (ASSPs), systems on chips (SoCs), complex programmable logic devices (CPLDs), computer hardware, firmware, software, and/or combinations thereof. The various embodiments may include implementations in one or more computer programs. The one or more computer programs are executable and/or interpretable on a programmable system including at least one programmable processor. A programmable processor may be a special-purpose or general-purpose programmable processor for receiving data and instructions from a memory system, at least one input apparatus, and at least one output apparatus and transmitting data and instructions to the memory system, the at least one input apparatus, and the at least one output apparatus.

[0101] Computer programs for implementation of the methods of the present application may be written in one programming language or any combination of multiple programming languages. The computer programs may be provided for a processor of a general-purpose computer, a special-purpose computer, or another programmable data processing apparatus to enable functions/operations specified in a flowchart and/or a block diagram to be implemented when the computer programs are executed by a processor. The computer programs may be executed entirely on a machine, partly on a machine, as a stand-alone software package, partly on a machine and partly on a remote machine, or entirely on a remote machine or a server.

[0102] In the context of the present application, the computer-readable storage medium may be a tangible medium that may include or store a computer program used by or in conjunction with an instruction execution system, apparatus, or device. The computer-readable storage medium may include, but is not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any appropriate combination thereof. Alternatively, the computer-readable storage medium may be a machine-readable signal medium. Examples of a machine-readable storage medium include an electrical connection based on one or more wires, a portable

computer disk, a hard disk, a RAM, a ROM, an erasable programmable read-only memory (EPROM), a flash memory, an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any appropriate combination thereof.

[0103] In order that interaction with a user is provided, the systems and techniques described herein may be implemented on the electronic device. The electronic device has a display apparatus (for example, a cathode-ray tube (CRT) or a liquid-crystal display (LCD) monitor) for displaying information to the user; and a keyboard and a pointing apparatus (for example, a mouse or a trackball) through which the user can provide input for the electronic device. Other types of apparatuses may also be used for providing interaction with the user. For example, feedback provided for the user may be sensory feedback in any form (for example, visual feedback, auditory feedback, or tactile feedback); and input from the user may be received in any form (including acoustic input, voice input, or tactile input).

[0104] The systems and techniques described herein may be implemented in a computing system including a back-end component (for example, a data server), a computing system including a middleware component (for example, an application server), a computing system including a front-end component (for example, a client computer having a graphical user interface or a web browser through which the user can interact with embodiments of the systems and techniques described herein), or a computing system including any combination of such back-end, middleware, or front-end components. Components of a system may be interconnected by any form or medium of digital data communication (for example, a communication network). Examples of the communication network include a local area network (LAN), a wide area network (WAN), a blockchain network, and the Internet.

[0105] The computing system may include clients and servers. A client and a server are generally remote from each other and typically interact through a communication network. The relationship between the client and the server arises by virtue of computer programs

running on respective computers and having a client-server relationship with each other. The server may be a cloud server, also referred to as a cloud computing server or a cloud host. As a host product in a cloud computing service system, the server solves the defects of difficult management and weak service scalability in a related physical host and a related virtual private server (VPS).

[0106] It is to be understood that various forms of the preceding flows may be used with steps reordered, added, or deleted. For example, the steps described in the present application may be executed in parallel, in sequence, or in a different order, as long as the desired results of the technical solutions of the present application can be achieved. The execution sequence of the steps is not limited herein.

CLAIMS

1. A version management method for a large language model, comprising:

acquiring version information and metadata that correspond to a large language model training task in a target tenant, wherein the version information comprises timestamp information and region information, and the metadata comprises performance indicator data;

determining a version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data; and

performing a version rollback operation on the large language model according to the version rollback strategy to complete a version rollback of the large language model.

2. The version management method for the large language model of claim 1, wherein determining the version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data comprises:

determining target performance data of versions in the large language model according to the timestamp information and the performance indicator data; and

determining the version rollback strategy of the large language model according to the target performance data of the versions and/or the region information.

3. The version management method for the large language model of claim 2, wherein determining the target performance data of the versions in the large language model according to the timestamp information and the performance indicator data comprises:

for each version in the large language model, assessing a credit assessment impact result of the version according to the performance indicator data; and

determining the target performance data of the large language model according to the credit assessment impact result and the timestamp information.

4. The version management method for the large language model of claim 3, wherein the performance indicator data comprises accuracy indicator data, recall indicator data, and reconciliation indicator data; and

for each version in the large language model, assessing the credit assessment impact result of the version according to the performance indicator data comprises:

inputting accuracy indicator data, recall indicator data, and reconciliation indicator data that correspond to the version into a credit assessment prediction model to obtain the credit assessment impact result of the version.

5. The version management method for the large language model of claim 2, wherein determining the version rollback strategy of the large language model according to the target performance data of the versions and/or the region information comprises:

generating a rollback recommendation report of the large language model according to the target performance data of the versions and/or the region information; and

determining the version rollback strategy of the large language model according to the rollback recommendation report.

6. The version management method for the large language model of claim 1, wherein performing the version rollback operation on the large language model according to the version rollback strategy to complete the version rollback of the large language model comprises:

performing the version rollback operation within a first rollback range of the large language model according to the version rollback strategy to obtain model performance data after the version rollback; and

continuing performing the version rollback operation on the large language model according to the model performance data after the version rollback to complete the version rollback of the large language model.

7. The version management method for the large language model of claim 6, wherein continuing performing the version rollback operation on the large language model according to the model performance data after the version rollback to complete the version rollback of the large language model comprises:

in response to the model performance data after the version rollback satisfying a set condition, gradually continuing performing the version rollback operation within a second rollback range until the version rollback operation is completed within an entire range of the

large language model, wherein the second rollback range is larger than the first rollback range.

8. A version management apparatus for a large language model, comprising:

a data acquisition module configured to acquire version information and metadata that correspond to a large language model training task in a target tenant, wherein the version information comprises timestamp information and region information, and the metadata comprises performance indicator data;

a strategy determination module configured to determine a version rollback strategy of the large language model according to the timestamp information, the region information, and the performance indicator data; and

a rollback operation execution module configured to perform a version rollback operation on the large language model according to the version rollback strategy to complete a version rollback of the large language model.

9. An electronic device, comprising:

at least one processor; and

a memory communicatively connected to the at least one processor,

wherein the memory is configured to store a computer program executable by the at least one processor to cause the at least one processor to perform the version management method for the large language model of any one of claims 1 to 7.

10. A computer-readable storage medium storing computer instructions that, when executed by a processor, are configured to cause the processor to perform the version management method for the large language model of any one of claims 1 to 7.