(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 870 706**

(51) Int. CI.:

H04L 9/08 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **11.01.2019 PCT/CN2019/071402**

(87) Fecha y número de publicación internacional: **18.04.2019 WO19072316**

(96) Fecha de presentación y número de la solicitud europea: **11.01.2019 E 19717099 (6)**

(97) Fecha y número de publicación de la concesión europea: **10.03.2021 EP 3602379**

(54) Título: **Un marco de trabajo de entrenamiento de modelo de seguridad distribuido de múltiples partes para la protección de privacidad**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
27.10.2021

(73) Titular/es:

ADVANCED NEW TECHNOLOGIES CO., LTD. (100.0%)
Cayman Corporate Centre, 27 Hospital Road George Town, Grand Cayman KY1-9008, KY

(72) Inventor/es:

WANG, HUAZHONG;
YIN, SHAN y
YING, PENGFEI

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

DESCRIPCIÓN

Un marco de trabajo de entrenamiento de modelo de seguridad distribuido de múltiples partes para la protección de privacidad

ANTECEDENTES

El aprendizaje automático es un subconjunto de la ciencia de datos que utiliza modelos estadísticos para extraer conclusiones y hacer predicciones. Para facilitar el intercambio de datos y la cooperación, diferentes partes pueden trabajar juntas para establecer un modelo de aprendizaje automático. Los proyectos tradicionales de aprendizaje automático agregan datos de entrenamiento obtenidos de múltiples partes en un solo lugar. Luego, en la fase de entrenamiento del proceso de aprendizaje automático, se crea un modelo de entrenamiento utilizando herramientas de aprendizaje automático en base a los datos agregados, de modo que el modelo se puede entrenar de manera uniforme. Los datos de entrenamiento pueden agregarse por cualquier parte participante, o por un tercero que sea confiable y seleccionado por todas las partes participantes.

Yichen Jiang et al: "SecureLR", IEEE/ACM, transacciones sobre biología computacional y bioinformática, vol. 16, no. 1, pág. 113-123, se refiere a un modelo de regresión logística seguro.

RESUMEN

La presente divulgación describe un marco de trabajo de entrenamiento de modelo de seguridad distribuido de múltiples partes para la protección de la privacidad.

En general, un aspecto innovador de la materia objeto descrita en esta memoria descriptiva puede realizarse en métodos que incluyen las acciones de recibir, en una pluralidad de nodos de cálculo seguro (SCN), una pluralidad de números aleatorios de un proveedor de números aleatorios; cifrar, en cada uno de los SCN, los datos almacenados en el SCN utilizando los números aleatorios recibidos; actualizar iterativamente un modelo de regresión logística seguro (SLRM) utilizando los datos cifrados de cada uno de los SCN; y después de actualizar iterativamente el SLRM, emitir un resultado del SLRM, en donde el resultado se configura para permitir que cada uno de los SCN realice un servicio. Otras realizaciones de este aspecto incluyen los correspondientes sistemas informáticos, aparatos y programas informáticos grabados en uno o más dispositivos de almacenamiento informáticos, cada uno configurado para realizar las acciones de los métodos.

Las anteriores y otras realizaciones pueden incluir opcionalmente una o más de las siguientes características, solas o en combinación. En particular, una realización incluye todas las siguientes características en combinación.

Esta memoria descriptiva también proporciona uno o más medios de almacenamiento legibles por computadora no transitorios acoplados a uno o más procesadores y que tienen instrucciones almacenadas en los mismos que, cuando se ejecutan por uno o más procesadores, hacen que el uno o más procesadores realicen operaciones de acuerdo con las implementaciones de los métodos proporcionados en el presente documento.

Esta memoria descriptiva proporciona además un sistema para implementar los métodos proporcionados en el presente documento. El sistema incluye uno o más procesadores y un medio de almacenamiento legible por computadora acoplado al uno o más procesadores que tiene instrucciones almacenadas en el mismo que, cuando se ejecutan por el uno o más procesadores, hacen que el uno o más procesadores realicen operaciones de acuerdo con las implementaciones de los métodos proporcionados en el presente documento.

Se aprecia que los métodos de acuerdo con esta memoria descriptiva pueden incluir cualquier combinación de los aspectos y características descritos en el presente documento. Es decir, los métodos de acuerdo con esta memoria descriptiva no se limitan a las combinaciones de aspectos y características descritas específicamente en el presente documento, sino que también incluyen cualquier combinación de los aspectos y características proporcionados.

Los detalles de una o más implementaciones de esta memoria descriptiva se establecen en los dibujos adjuntos y la descripción a continuación. Otras características y ventajas de esta memoria descriptiva serán evidentes a partir de la descripción y los dibujos, y de las reivindicaciones.

DESCRIPCION DE LOS DIBUJOS

La FIG. 1 representa un ejemplo de un entorno para entrenar un modelo de regresión logística seguro (SLRM) de aprendizaje automático de múltiples partes utilizando el intercambio de secretos de acuerdo con las implementaciones de la presente memoria descriptiva.

La FIG. 2A representa un ejemplo de un subproceso para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.

La FIG. 2B representa un ejemplo de un subproceso para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.

La FIG. 3 representa un ejemplo de un subproceso para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.
La FIG. 4A representa un ejemplo de un subproceso para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.
La FIG. 4B representa un ejemplo de un subproceso para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.
La FIG. 5 representa un ejemplo de un proceso para determinar si se deben terminar las iteraciones de actualización de parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.
La FIG. 6 representa un ejemplo de un proceso para entrenar un SLRM de múltiples partes utilizando un procedimiento de modelado seguro interactivo impulsado por eventos de acuerdo con implementaciones de esta memoria descriptiva.
La FIG. 7 representa un ejemplo de un proceso que puede ejecutarse de acuerdo con implementaciones de la presente memoria descriptiva.
La FIG. 8 representa un ejemplo de un proceso que puede ejecutarse de acuerdo con implementaciones de la presente memoria descriptiva.
La FIG. 9 representa un ejemplo de un diagrama que ilustra módulos de un aparato de acuerdo con implementaciones de la especificación

Los números de referencia y las designaciones similares en los diversos dibujos indican elementos similares.

DESCRIPCIÓN DETALLADA

La siguiente descripción detallada describe un marco de trabajo de entrenamiento de modelo de seguridad distribuido de múltiples partes para la protección de privacidad, y se presenta para permitir que cualquier experto en la técnica realice y utilice la materia objeto divulgada en el contexto de una o más implementaciones particulares. Se pueden hacer diversas modificaciones, alteraciones y permutaciones de las implementaciones divulgadas y serán fácilmente evidentes para los expertos en la técnica, y los principios generales definidos se pueden aplicar a otras implementaciones y aplicaciones, sin apartarse del alcance de la divulgación. En algunos casos, se pueden omitir detalles innecesarios para obtener una comprensión de la materia objeto descrita para no oscurecer una o más implementaciones descritas con detalles innecesarios y en la medida en que tales detalles están dentro del conocimiento de un experto en la técnica. La presente divulgación no se pretende que esté limitada a las implementaciones divulgadas o ilustradas, sino que se le conceda el alcance más amplio de acuerdo con los principios y características descritos.

Se aprovechan grandes conjuntos de información sobre los consumidores, a veces denominados "macrodatos", para proporcionar una mejor comprensión de los hábitos de los consumidores, una mejor orientación para las campañas de marketing, una mayor eficiencia operativa, costos más bajos y menores riesgos, junto con otros beneficios. Con los avances tecnológicos del ecosistema de macrodatos, el intercambio de datos y la cooperación entre diferentes organizaciones se ha vuelto más frecuente, refinado y complicado. Como tal, algunas organizaciones han establecido laboratorios de datos cooperativos para construir un modelo de entrenamiento de datos seguro y para facilitar el intercambio de datos entre organizaciones. Durante el proceso de cooperación de datos, se construye un modelo de aprendizaje automático en base a datos obtenidos de diferentes organizaciones participantes. Las organizaciones pueden agregar sus datos y colocar físicamente los datos agregados en un solo lugar, y luego utilizar una herramienta madura de aprendizaje automático para construir un modelo en base a los datos agregados. Los datos agregados se pueden almacenar en una organización participante o en un tercero en el que todos los participantes confíen.

Se pueden implementar esquemas de aprendizaje automático basados en cifrado homomórfico o privacidad diferencial para entrenar dichos modelos a fin de garantizar la protección de privacidad de datos. Por ejemplo, el esquema de aprendizaje automático que se basa en el cifrado homomórfico o micromórfico permite que los datos de ambas partes se cifren completamente de manera homomórfica y luego agrega los datos cifrados para completar el entrenamiento del modelo en base al texto cifrado. Debido a que el cifrado es homomórfico, el entrenamiento del modelo producirá resultados cifrados que, cuando se descifren, coincidirán con los resultados de entrenamiento que se obtendrían del entrenamiento con los datos de entrenamiento no cifrados. Los resultados de salida del entrenamiento del modelo se devuelven a las partes participantes para que puedan descifrar y obtener los resultados finales.

Existen diversos problemas en las tecnologías actuales de cooperación de datos. En primer lugar, según el enfoque actual, cada una de las partes debe exponer sus propios datos a otras partes, lo que genera un riesgo de fuga de privacidad de datos, especialmente para los datos altamente sensibles. En segundo lugar, aunque muchos laboratorios de datos han implementado técnicas tales como computación en la nube, separación de múltiples inquilinos, desensibilización de datos, auditoría de datos, etc. para garantizar la privacidad de datos, las partes aún no tienen una garantía de seguridad después de que los datos abandonan su dominio. Como tal, la selección de una parte participante o un tercero neutral para agregar datos y procesar los datos agregados puede ser difícil debido a la falta de confianza entre las partes. En tercer lugar, incluso si ambas partes confían una en la otra y están de acuerdo con la salida de la muestra, debido a que la industria carece de estándares y pautas para la cooperación de datos, el nivel de seguridad de la tecnología implementada por cada una de las partes puede ser diferente, lo que resulta en una gran cantidad de tiempo y costos de comunicación para el cifrado de datos de muestra, la desensibilización de datos,

la colisión de datos, la aprobación de seguridad, etc. En cuarto lugar, el esquema de aprendizaje automático que, en base a cifrado homomórfico o privacidad diferencial, sigue siendo una solución de modelado centralizada que es ineficiente y genera salidas inexactas en implementaciones de ingeniería. Por ejemplo, la velocidad computacional del esquema de aprendizaje automático en base al cifrado homomórfico es lenta. La tecnología homomórfica actual no está madura y aún presenta problemas tales como largos tiempos de entrenamiento, baja eficiencia de trabajo, gestión complicada de claves de cifrado, etc. Asimismo, aunque el esquema de aprendizaje automático basado en la privacidad diferencial puede reducir el riesgo de fuga de privacidad para los datos utilizados en el proceso de cooperación de datos, el modelo resultante sufre una pérdida de exactitud y precisión debido a la transformación de los datos. En quinto lugar, las herramientas y procesos de modelado tradicionales, aunque potentes, son exigentes para los modeladores y requieren mucho tiempo y esfuerzo para construir los modelos de entrenamiento.

Las implementaciones de esta divulgación introducen un nuevo enfoque de entrenamiento de SLRM utilizando SS y un procedimiento de modelado seguro interactivo dirigido por eventos. Las implementaciones descritas aplican un modelo de SLRM que se basa en la regresión logística y puede actualizarse iterativamente alimentando los datos de entrenamiento recibidos de ambas partes. La regresión logística es una regresión lineal generalizada y es un tipo de algoritmos de clasificación y predicción. El algoritmo de regresión logística estima valores discretos de una serie de variables dependientes conocidas y estima la probabilidad de que ocurra un evento ajustando los datos a una función lógica. La regresión logística se utiliza principalmente para la clasificación, tal como la clasificación de correo electrónico no deseado, la clasificación de predicción de riesgo crediticio, etc.

Las fuentes de los datos de entrenamiento de muestra son nodos de cálculo seguro independientes (SCN) y, cada uno de los SCN, mantiene sus datos de entrenamiento en secreto de otros nodos utilizando un esquema de *intercambio de secretos* (SS). Como se describió anteriormente, dado que los datos de entrenamiento se proporcionan por diferentes SCN, asegurar los datos de entrenamiento para compartirlos entre los SCN se vuelve muy importante. Específicamente, para consolidar dicha cooperación, las técnicas descritas introducen números aleatorios para permitir que cada uno de los SCN proteja sus datos de entrenamiento privados de los otros SCN. Las técnicas descritas pueden abordar problemas como la falta de confianza mutua entre las partes en el proceso de cooperación de datos, evitar la filtración de datos de entrenamiento recibidos de ambas partes y promover de manera efectiva la cooperación entre las partes en la construcción de un modelo de aprendizaje automático.

Además, el intercambio de datos de entrenamiento entre los SCN se puede controlar mediante un procedimiento de modelado seguro integrador dirigido por eventos. En algunas implementaciones, el procedimiento de modelado seguro interactivo dirigido por eventos se basa en un modelo dirigido por eventos (o el llamado modelo de "publicación-suscripción"). Este modelo puede separar a las partes que dependen unas de otras para completar su propia tarea. Al utilizar este modelo, ambas partes pueden centrarse en su propio proceso de servicio. Una parte (el "editor") envía un aviso cuando se completa su proceso de servicio. La otra parte (el "suscriptor") monitoriza la notificación y, una vez que se recibe la notificación, su proceso de servicio puede activarse en consecuencia. Durante el proceso de entrenamiento, cada uno de los SCN mantiene una cola de mensajes para recibir datos de la otra parte y, en respuesta, activa los pasos de modelado correspondientes para continuar con el SLRM.

Las técnicas descritas pueden tener una variedad de aplicaciones. Por ejemplo, las técnicas se pueden aplicar en la cooperación de datos entre instituciones financieras, instituciones financieras y una entidad gubernamental u otras organizaciones.

En algunas implementaciones, el marco de trabajo divulgado utiliza un modelo de regresión logística seguro (SLRM) y un esquema de intercambio de secretos (SS). El esquema SS es un tipo de cifrado que se utiliza en escenarios que involucran la colaboración entre múltiples partes. En un esquema de SS, un secreto (p. ej., una clave de cifrado o un conjunto de datos) se divide en varias participaciones diferentes de una manera predeterminada y, cada una de las participaciones, se proporciona a una parte participante diferente. El secreto no puede recuperarse o restaurarse por una sola parte participante, de esta forma se asegura el secreto y la seguridad del secreto. A los efectos de esta divulgación, el algoritmo utilizado para proteger el entrenamiento utilizado por el SLRM no se limita a SS. En comparación con el cifrado homomórfico, la eficiencia computacional general se mejora enormemente utilizando intercambio de secretos. Además, debido a que los datos de entrenamiento en bruto no se transforman, el proceso de entrenamiento es un cálculo preciso basado en los datos de entrenamiento en bruto, y el resultado del modelo de salida del entrenamiento es un modelo entrenado con precisión basado en los datos de entrenamiento sin procesar.

En algunas implementaciones, se implementa un marco de trabajo de entrenamiento de modelo distribuido y el marco de trabajo introduce un servicio de números aleatorios del tercero independiente. En estas implementaciones, antes de que se ingrese cualquier dato en bruto al modelo de entrenamiento, éste pasa por un proceso de cálculo junto con los números aleatorios generados y distribuidos por el servicio de números aleatorios del tercero. En tales implementaciones, se adopta además un procedimiento seguro interactivo dirigido por eventos para mejorar la eficiencia del modelo distribuido. De esta manera, se puede evitar la fuga de información de los datos en bruto y se asegura la integridad del resultado del modelado y la precisión del modelo.

La FIG. 1 representa un ejemplo de un entorno 100 para entrenar un modelo de regresión logística seguro (SLRM) de aprendizaje automático de múltiples partes utilizando el intercambio de secretos de acuerdo con implementaciones de

la presente memoria descriptiva. El servicio de cooperación de datos se puede realizar por uno o más servidores que proporcionan a las organizaciones una plataforma/entorno para la cooperación de datos. El entorno 100 de ejemplo incluye un usuario 102, un agente 104 de gestión de seguridad, un agente 106 de gestión de nodos, un proveedor 108 de números aleatorios, una red 110 y al menos dos nodos SCN A 112 y SCN B 114 de cálculo seguro (SCN). El entorno 100 de ejemplo puede incluir usuarios adicionales, computadoras, redes, sistemas u otros componentes del nodo de cálculo seguro. El entorno 100 se puede configurar de otra manera en algunas implementaciones.

En algunas implementaciones, la red 110 incluye una red de área local (LAN), una red de área amplia (WAN), el Internet o una combinación de estas u otras redes. La red 110 puede incluir una o más de una red inalámbrica o redes cableadas. La red 110 conecta dispositivos informáticos (p. ej., los servidores 104-108) y nodos de cálculo seguros (p. ej., los SCN 112, 114). En algunas implementaciones, se puede acceder a la red 110 a través de un enlace de comunicaciones cableado y/o inalámbrico.

En algunos casos, antes del inicio del proceso de entrenamiento, un usuario 102 puede predeterminar diversos parámetros y ajustes asociados con el SLRM. Estos parámetros pueden incluir, por ejemplo, el tamaño de los datos de entrenamiento, las características de los datos de entrenamiento y los correspondientes ajustes del procesamiento de datos, o un hiperparámetro (es decir, un parámetro de una distribución previa) del modelo. Después de inicializar los ajustes del modelo, la información asociada con el modelo se envía al agente 104 de gestión de seguridad, que posteriormente puede coordinarse con los SCN para completar la lógica de modelado específica.

En algunas implementaciones, el agente 104 de gestión de seguridad y el agente 106 de gestión de nodos pueden integrarse como un solo componente para gestionar y controlar el proceso de modelado, así como los SCN (p. ej., 112, 114) que participan en el proceso. El agente 104 de gestión de seguridad y el agente 106 de gestión de nodos también pueden ser componentes separados, como se muestra en la FIG. 1.

En algunas implementaciones, el agente 104 de gestión de seguridad puede estar configurado para gestionar el entrenamiento del SLRM, incluyendo la configuración, el modelado, el cálculo iterativo y la gestión de procesos del modelo de entrenamiento. Específicamente, el agente 104 de gestión de seguridad puede facilitar el proceso de modelado llevado a cabo conjuntamente por los SCN 112, 114, incluyendo, por ejemplo, inicializar cada uno de los ciclos de iteración y determinar si el modelo de entrenamiento converge después de una serie de círculos de iteración. Además, el agente 104 de gestión de seguridad también puede obtener una estructura de datos específica y otra información asociada con los datos de entrenamiento de cada uno de los SCN. Por ejemplo, obtener información tal como la fuente de los datos de entrenamiento, las características de los datos de entrenamiento (p. ej., el número de filas y columnas de cada uno de los conjuntos de datos de entrenamiento).

En algunas implementaciones, el agente 106 de gestión de nodos puede estar configurado para realizar la selección de nodos, la gestión de nodos, la gestión de datos de proyecto, etc. Por ejemplo, una vez que el agente 104 de gestión de seguridad recibió solicitudes de modelado del SCN A 112 y el SCN B 114, y recopiló las correspondientes información de datos de los SCN A y B, el agente 106 de gestión de nodos puede informar, a través del agente 104 de gestión de seguridad, a cada uno de los SCN que la identidad de otros SCN con los que necesita trabajar, la ubicación para obtener datos de entrenamiento adicionales y distribuir los parámetros correspondientes a cada uno de los SCN que participa en el proceso de entrenamiento. Además, el agente 106 de gestión de nodos también notifica a los SCN A y B de la ubicación del proveedor 108 de números aleatorios.

En el ejemplo representado, el SCN A 112 y el SCN B 114 pueden operarse por organizaciones separadas que poseen conjuntos de datos sobre sus usuarios separados que buscan facilitar el intercambio de datos o la cooperación de datos entre sí. Por ejemplo, las organizaciones que operan el SCN A 112 y el SCN B 114 pueden poseer conjuntos P0 y P1 de datos de entrenamiento, respectivamente, siendo P0 y P1 entradas para un modelo de aprendizaje automático. El resultado emitido del modelo de entrenamiento puede ser, por ejemplo, un resultado de predicción que puede utilizarse tanto por el SCN A 112 como por el SCN B 114 para llevar a cabo un determinado servicio, tal como una predicción del riesgo de emitir un préstamo a un cliente del SCN A 112 o el SCN B 114. Después de que los SCN A 112 y B 114 recibieron los parámetros y ajustes de modelo del agente 104 de gestión de seguridad, necesitan completar la tarea de modelado a través de comunicaciones de red entre sí. Por motivos de seguridad, especialmente en los casos en los que los datos de entrenamiento propiedad de cada uno de los SCN son datos de privacidad altamente sensibles, los SCN A 112 y B 114 pueden ocultar u ofuscar porciones de los datos de entrenamiento entre sí. Para ofuscar los datos de entrenamiento, los SCN A 112 y B 114 pueden solicitar números aleatorios del proveedor 108 de números aleatorios y realizar cálculos utilizando sus propios datos de entrenamiento y los números aleatorios recibidos (por ejemplo, suma o multiplicación de números aleatorios y porciones de los datos). Los números aleatorios se pueden utilizar para proporcionar la ofuscación de datos o el cifrado de los datos de entrenamiento. En algunos casos, solo se pueden ofuscar o cifrar porciones de los datos de entrenamiento que contienen datos de privacidad altamente sensibles (p. ej., información personal de los usuarios), lo que permite que la otra parte utilice las porciones no sensibles de los datos para el entrenamiento sin permitir el acceso de otra parte a los datos de privacidad altamente sensibles. Se puede utilizar cualquier esquema de cifrado adecuado, tal como RSA, DES/TripleDES y otros esquemas de cifrado bien conocidos. En algunas implementaciones, el número aleatorio puede ser un número, un vector o una matriz, etc. En algunas implementaciones, el número aleatorio puede generarse por cualquiera de los SCN 112, 114. En algunas implementaciones, los números aleatorios se pueden proporcionar por un un tercero independiente para

garantizar que el otro SCN no pueda revelar datos privados durante el intercambio y el entrenamiento. En algunas implementaciones, los servicios relacionados con el proveedor 108 de números aleatorios, el agente 104 de gestión de seguridad y el agente 106 de gestión de nodos pueden proporcionarse y realizarse por un agente externo de confianza y seleccionado mutuamente.

Para cada uno de los SCN, una vez que los datos se cifran utilizando números aleatorios, los datos cifrados se pueden enviar al otro SCN. Debido a que los datos transmitidos entre los SCN 112, 114 están cifrados, la información sensible de los datos no está expuesta. En algunas implementaciones, los resultados de los cálculos de los SCN A y B 112, 114 se utilizan para el entrenamiento del modelo de aprendizaje automático. Específicamente, cada uno de los SCN utiliza los datos cifrados recibidos como entradas al SLRM, para actualizar iterativamente el parámetro de la función de regresión logística en la que se basa el SLRM. Después de varias iteraciones, el entrenamiento puede terminarse en base una condición predeterminada. En algunas implementaciones, la gestión de datos local la realiza cada uno de los SCN 112, 114 y puede incluir el almacenamiento de datos en una base de datos, un almacén de objetos, un almacén de datos en memoria (p. ej., Redis) u otro tipo de almacenamiento.

La FIG. 2A representa un ejemplo de un subproceso 200a para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.

En el ejemplo representado, para optimizar una función objetivo del SLRM, se utiliza un método de descenso de gradiente estocástico (SGD) de mini lotes. El SGD es un método iterativo para optimizar una función objetivo diferenciable en base a una aproximación estocástica de la optimización del descenso de gradiente. En esta divulgación, para optimizar la función objetivo, el método de SGD realizaría las siguientes iteraciones:

$$\theta[j] := \theta[j] - \frac{\alpha}{m}\sum_{i=1}^{m}(sigmoide(X_i \cdot \theta) - Y_i)X_i[j] - \frac{\lambda}{m}\theta[j]$$

ec. 1

donde m representa el tamaño de la muestra del SGD de mini lotes. En este ejemplo, X es una matriz de muestra de $m * k$. Cada una de las filas de la matriz X representa una muestra, y X_i representa la $i^{ésima}$ fila de la matriz X, [j] representa el $j^{ésimo}$ elemento del vector X_i. θ representa un vector de columna de parámetro. Y_i representa la etiqueta de la $i^{ésima}$ muestra de las muestras de mini lotes, donde Y_i puede ser 0 o 1. α y λ son hiperparámetros que determinan una estructura de red de aprendizaje automático y cómo se entrena la red. Se pueden configurar antes del entrenamiento y por un usuario.

En este ejemplo representado, al utiliza SS, todos los conjuntos de datos de entrenamiento de muestra utilizados en la ecuación (1) se dividen en dos participaciones. Es decir, los datos de entrenamiento de muestra para el SLRM provienen de dos fuentes, SCN P_0 y SCN P_1. Específicamente, la matriz X de datos de muestra se divide en X^L y X^R, por lo que los datos X^L de muestra pertenecen a SCN P_0, mientras que X^R pertenecen a SCN P_1. Suponiendo que los datos de las muestras en SCN P_0 y SCN P_1 tienen características k_0 y k_1, respectivamente, y $k = k_0 + k_1$. Para un conjunto X_i de datos de muestra, SCN P_0 contiene elementos de X_i, expresados como $X_i[1]$, $X_i[2]$,... $X_i[k_0]$, mientras que SCN P_1 contiene elementos de X_i, expresado como $X_i[k_0 + 1]$, $X_i[k_0 + 2]$,... $X_i[k]$. Del mismo modo, la vector θ columna de parámetros también se puede dividir en θ^L y θ^R. Como tal, utilizando estos parámetros redefinidos, bajo la división de datos vertical, la ecuación (1) se puede expresar como:

$$\theta[j] := \theta[j] - \frac{\alpha}{m}\sum_{i=1}^{m}(sigmoide(X_i^L \cdot \theta^L + X_i^R \cdot \theta^R) - Y_i)X_i[j] - \frac{\lambda}{m}\theta[j]$$

ec. 2

Donde X e Y son los datos de entrada a ser protegidos. X representa datos de características e Y representa una predicación basada en los datos de características. Por ejemplo, en un escenario de préstamo-empresito, X puede ser el ingreso, historial educativo, historial crediticio, etc. En base a la información, el banco puede hacer una predicción Y, que es si el riesgo del prestatario del préstamo es lo suficientemente bajo como para emitir el préstamo. θ es un resultado intermedio que debe protegerse y, una vez finalizado el proceso de entrenamiento, θ también es un resultado final del resultado de entrenamiento. Otros parámetros enumerados en la ecuación (2) son parámetros bien conocidos que se utilizan habitualmente.

El algoritmo de entrenamiento de regresión logística seguro utilizado en el SLRM se basa en un SGD de mini lotes que tiene una regularización de segundo orden y aplica el circuito (A) matemático y el Circuito Garbled (Yao) del marco de trabajo de cálculo seguro ABY de dos partes. El circuito (A) matemático se basa en un triple de multiplicación y SS, y se puede utilizar para calcular sumas, restas y multiplicaciones, por lo tanto, puede calcular polinomios, productos escalares y multiplicaciones de matrices que se utilizan en el algoritmo. Mientras tanto, el circuito Garbled (Yao) se puede utilizar para realizar otros tipos de cálculos. En este marco de trabajo computacional, cada uno de los SCN utiliza SS para un cálculo seguro. Después de cada uno de los pasos de cálculo, se genera un resultado de cálculo intermedio y se divide en participaciones (que pueden ser iguales o desiguales), mientras que cada uno de los SCN obtiene una participación del resultado de cálculo intermedio. Después de cada uno de los pasos, ambos SCN ingresan

al siguiente paso utilizando las participaciones obtenidas y, finalmente, combinan las participaciones finales generadas cuando finaliza la iteración.

Antes de que comience el bucle de iteración, primero se inicializan todos los parámetros utilizados para actualizar el modelo. Durante el proceso de inicialización, ambos SCN generan un vector columna aleatorio como una participación del vector θ columna de parámetros inicializado, donde dim(θ) = (k_0 + k_1 +1, 1), dim(θ^L) = (k_0 +1, 1), dim(θ^R) = (k_1, 1). El vector θ columna de parámetros inicializado se puede representar por sus participaciones como:

$$\theta^L = <\theta^L>_0 + <\theta^L>_1$$

ec. 3

$$\theta^R = <\theta^R>_0 + <\theta^R>_1$$

ec. 4

donde <> representa una participación, e [i] representa el *i-ésimo* elemento de un vector (es decir, θ[i] = <θ [i]>$_0$ + <θ [i]>$_1$). Por lo tanto, el vector columna aleatorio generado en SCN P_0 se puede expresar como la siguiente ecuación:

$$<\theta>_0 = (<\theta[1]>_0, <\theta[2]>_0, <\theta[3]>_0 \ \ldots <\theta[k]>_0) \qquad \text{ec. 5}$$

Debido a que cada uno de los conjuntos de datos de muestra se divide verticalmente, <θ>$_0$ se puede dividir además por los siguientes dos vectores columna:

$$<\theta^L>_0 = (<\theta^L[1]>_0, <\theta^L[2]>_0, <\theta^L[3]>_0 \ \ldots <\theta^L[k_0]>_0) \qquad \text{ec. 6}$$

$$<\theta^R>_0 = (<\theta^R[1]>_0, <\theta^R[2]>_0, <\theta^R[3]>_0 \ \ldots <\theta^R[k_1]>_0)$$

ec. 7

donde <θ>$_0$ = <θ^L>$_0$ ‖ <6^R >$_0$, y donde ‖ representa una relación de conexión.

Asimismo, el vector columna aleatorio generado en SCN P_1 se puede expresar como la siguiente ecuación:

$$= (<\theta[1]>_1, <\theta[2]>_1, <\theta[3]>_1 \ \ldots <\theta[k]>_1)$$

ec. 8

Debido a que cada uno de los conjuntos de datos de muestra se divide verticalmente, <θ>$_1$ se puede dividir además por los siguientes dos vectores columna:

$$<\theta^L>_1 = (<\theta^L[1]>_1, <\theta^L[2]>_1, <\theta^L[3]>_1 \ \ldots <\theta^L[k_0]>_1)$$

ec. 9

$$<\theta^R>_1 = (<\theta^R[1]>_1, <\theta^R[2]>_1, <\theta^R[3]>_1 \ \ldots <\theta^R[k_1]>_1)$$

ec. 10

donde <θ>$_1$ = <θ^L>$_1$ ‖ <θ^R >$_1$, y donde ‖ representa una relación de conexión.

El proceso de iteraciones se ilustra en las FIG. 2-6. Como se explicó anteriormente, el propósito de las iteraciones es actualizar el parámetro θ que se muestra en la ecuación (2).

La FIG. 2A ilustra el primer subpaso (en el presente documento como "paso 1.1") del primer paso del círculo de iteración (en el presente documento como "paso 1") durante una iteración. En el paso 1, cada uno de los SCN calcula primero una participación de A en base a la ecuación:

$$A = X\Theta = X^L\theta^L + X^R\theta^R$$

ec. 11

donde A se calcula utilizando el vector X de muestras de mini lotes multiplicado por el vector columna del parámetro θ.

En algunas implementaciones, X^L y X^R también se utilizarán para actualizar los parámetros en el paso 3 del círculo de iteración. En algunas implementaciones, los mismos datos de muestra de mini lotes se pueden solicitar para su cálculo en una iteración posterior.

La FIG. 2A (paso 1.1) ilustra el cálculo del valor de $X^L\theta^L$, utilizando la multiplicación triple y SS. Como se muestra en la FIG. 2A, el SCN P_0 es la parte que proporciona los datos originales. Para ocultar los datos originales, el SCN P_0 y el SCN P_1 primero obtienen números aleatorios. En algunas implementaciones, el número aleatorio puede ser un número, un vector o una matriz. En alguna implementación, los SCN pueden generar los números aleatorios. En algunas implementaciones, el número aleatorio se puede solicitar y recibir de una agencia de terceros, por ejemplo, el proveedor 108 de números aleatorios mostrado en la FIG. 1.

Como se muestra en la FIG. 2A, los números aleatorios obtenidos incluyen una matriz u aleatoria, un vector v aleatorio y un número z aleatorio. Los números aleatorios obtenidos se distribuyen entre SCN P_0 y SCN P_1. En algunas implementaciones, SCN P_0 obtiene una matriz u aleatoria, una participación de v que se expresa como v_0, una participación de z que se expresa como z_0. El SCN P_1 obtiene una participación de v que se expresa como v_1, y una participación de z que se expresa como z_1. En algunas implementaciones, las participaciones de z pueden generarse mediante un cálculo homomórfico. En algunas implementaciones, el vector u aleatorio se genera por el SCN P_0, mientras que la participación v_0 y v_1 del vector v aleatorio se generan por el SCN P_0 y el SCN P_1, respectivamente. En algunas implementaciones, u, v y z y sus participaciones correspondientes se generados todos por un servidor de productos de terceros de confianza. Cada uno de los números aleatorios y la participación del número aleatorio correspondiente están interrelacionados y la relación entre ellos se puede expresar como:

$$z_0 + z_1 = u*v = u*(v_0 + v_1)$$

ec. 12

Suponiendo que a = X^L, y $b_0 = \langle\theta^L\rangle_0$. Para ocultar los datos de SCN P_0, cifra a utilizando u, por ejemplo, realizando una suma, una resta o una multiplicación de a y u. En alguna implementación, la matriz u aleatoria puede ser la misma cuando se oculta la misma X^L en una iteración posterior. Asimismo, $\langle\theta^L\rangle_0$ se cifra utilizando el vector v_0 aleatorio. Como se muestra en la FIG. 2A, el SCN P_0 envía la X^L cifrada (que se expresa como e = a - u) y el $\langle\theta^L\rangle_0$ cifrado (que se expresa como (b_0 - v_0)) al SCN P_1. En algunas implementaciones, si se utiliza la misma matriz u aleatoria para ocultar la misma X^L en una iteración posterior, no es necesario volver a enviar la X^L cifrada (e = a-u). En el lado del SCN P_1, suponiendo que $b_1 = \langle\theta^L\rangle_1$. En este caso, el vector v_1 aleatorio compartido se utiliza para ocultar el valor de $\langle\theta^L\rangle_1$, y (b_1 - v_1) se envía desde el SCN P_1 al SCN P_0.

Después del intercambio de datos entre SCN P_0 y SCN P_1, la cola de datos del paso 1.1 en SCN P_0 se actualiza como:

$$c_0 = u*f + e*b_0 + z_0$$ ec. 13

donde f = b - v

Y la cola de datos del paso 1.1 en SCN P_1 se actualiza como:

$$c_1 = e*b_1 + z_1$$ ec. 14

donde cada uno de c_0 y c_1 es una participación de $X^L\theta^L$,

Según las ecuaciones anteriores, $X^L\theta^L$ se puede calcular como:

$$X^L\theta^L = c_0 + c_1$$ ec. 15

$$c_0 + c_1 = u*f + e*b + u*v$$ ec. 16

$$c_0 + c_1 = u*b - u*v + a*b - u*b + u*v$$ ec. 17

FIG. 2B representa un ejemplo de un subproceso 200b para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.

Específicamente, la FIG. 2B muestra un segundo subpaso (en el presente documento como paso 1.2) del paso 1 del círculo de iteración. En el paso 1.2, el valor de $X^R\theta^R$ se calcula utilizando la multiplicación triple y SS. Como se muestra en la FIG. 2B, el SCN P_1 es la parte que proporciona los datos originales. Para ocultar los datos originales, el SCN P_1 y el SCN P_0 primero obtienen números aleatorios.

Como se muestra en la FIG. 2B, los números aleatorios obtenidos incluyen una matriz u aleatoria, un vector v aleatorio y un número z aleatorio. Los números aleatorios obtenidos se distribuyen entre SCN P_1 y SCN P_0. El SCN P_1 obtiene una matriz u aleatoria, una participación de v que se expresa como v_0, una participación de z que se expresa como z_0. El SCN P_0 obtiene una participación de v que se expresa como v_1, y una participación de z que se expresa como z_1. En algunas implementaciones, las participaciones de z pueden generarse mediante cálculo homomórfico. En algunas implementaciones, el vector aleatorio u se genera por SCN P_1, mientras que la participación v_0 y v_1 del vector v aleatorio se generan por el SCN P_1 y el SCN P_0, respectivamente. En algunas implementaciones, u, v y z y su participación se generan todos por un servidor de productos de terceros de confianza. Cada uno de los números aleatorios y la participación del número aleatorio correspondiente están interrelacionados y satisfacen una relación como se muestra en la ecuación (12).

Suponiendo que a = X^R, y b_0 = $<\theta^R>_0$, para ocultar la información de datos de a, el SCN P_1 primero cifrar a mediante u, por ejemplo, realizar una adición, una sustracción o una multiplicación entre a y u. En alguna implementación, la matriz u aleatoria puede ser la misma cuando se oculta la misma X^R en una iteración posterior. Asimismo, $<\theta^R>_0$ se cifra primero utilizando el vector v_0 aleatorio. Como se muestra en la FIG. 2B, el SCN P_1 envía la X^R cifrada (que se expresa como e = a - u) y la $<\theta^R>_0$ cifrada (que se expresa como (b_0 - v_0)) al SCN P_0. En algunas implementaciones, si se utiliza la misma matriz u aleatoria para ocultar la misma X^L en una iteración posterior, no es necesario volver a enviar la X^L cifrada (e = a - u). En el lado del SCN P_0, suponiendo que b_1 = $<\theta^R>_1$. En este caso, el vector v_1 aleatorio compartido se utiliza para ocultar el valor de $<\theta^R>_1$, y (b_1 - v_1) se envía desde el SCN P_0 al SCN P_1.

Después del intercambio de datos entre el SCN P_1 y el SCN P_0, la cola de datos del paso 1.2 en SCN P_1 se actualiza mediante la ecuación (13), y la cola de datos del paso 1.2 en SCN P_0 se actualiza mediante la ecuación (14). En base a las ecuaciones anteriores, $X^L\theta^L$ se puede calcular mediante las ecuaciones (16) y (17), así como también:

$$X^R\theta^R = c_0 + c_1 \qquad \text{ec. 18}$$

Como tal, se completa el paso 1 de un círculo de iteración y la ecuación (11) se puede calcular combinando los resultados de las ecuaciones (15) y (18).

La FIG. 3 representa un ejemplo de un subproceso 300 para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva. Específicamente, la FIG. 3 ilustra el segundo paso en el ciclo de iteración (en el presente documento denominado "paso 2").

En el paso 2, se calcula un vector columna del error de predicción utilizando una ecuación:

$$E = g(A) - Y_i$$

ec. 19

Donde A es el mismo valor que en la ecuación (11) descrita en el paso 1, g es una función de ajuste de una función sigmoide, que puede ser un ajuste de función polinómica. Es decir, g() puede ser una función polinómica y su orden más alto es d. En el ejemplo representado en esta divulgación, ajustar d = 7, y g (x) se puede expresar mediante la ecuación:

$$g(x) = 0.5 + 1.73496*(x/8) - 4.19407*(x/8)^3 + 5.43402*(x/8)^5 - 2.50739*(x/8)^7 \qquad \text{ec. 20}$$

donde g(A) representa el vector columna compuesto por g (A[1]), g A [2]),... g(A[m]).

Utilizando la multiplicación triples y SS, cada uno del SCN P_0 y del SCN P_1 necesita calcular g($<A[i]_0>$ + $<A[i]_1>$). Suponiendo que p = $<A[i]_0>$ y q = $<A[i]_1>$). La función g(p + q) polinómica se puede expandir como la función:

$$g(p+q) = h_0(q) + h_1(q)p + h_2(q)p^2 + \ldots + h_d(q)p^d$$

ec. 21

Donde cada uno de h_0, h_1,... h_d es una función polinómica de q, y se puede calcular mediante el SCN P_1, mientras que p, p^2, p^3... p^d se pueden calcular mediante el SCN P_0.

Suponiendo que el vector a = (p, p^2, p^3... p^d) y el vector b = ($h_1(q)$, $h_2(q)$,... $h_d(q)$), entonces g(p + q) = a • b + $h_d(q)$, donde el producto escalar (a • b) puede calcularse mediante el proceso ilustrado en la FIG. 3, que incluye la multiplicación triple y SS optimizados. Se puede calcular un resultado final de E realizando una suma en base al SS.

Como se muestra en la FIG. 3, los números aleatorios obtenidos incluyen una matriz u aleatoria, un vector v aleatorio y un número z aleatorio. Los números aleatorios obtenidos se distribuyen entre SCN P_0 y SCN P_1. El SCN P_0 obtiene una matriz u aleatoria, una participación de z que se expresa como z_0. El SCN P_1 obtiene el vector v aleatorio y una participación de z que se expresa como z_1. Cada uno de los números aleatorios y la participación del número aleatorio correspondiente están interrelacionados y la relación entre ellos se puede expresar como:

$$z_0 + z_1 = u*v \qquad \text{ec. 22}$$

Entonces, como se ilustra en la FIG. 3, el SCN P_0 envía primero datos cifrados e = a - u al SCN P_1, y el SCN P_1 envía datos cifrados f = b - v al SCN P_0. Los pasos y cálculos son similares a los del paso 1, para obtener más detalles, por favor, consultar las descripciones anteriores de las FIG. 2A y 2B.

Después del intercambio de datos entre SCN P_0 y SCN P_1, la cola de datos del paso 2 en SCN P_0 se actualiza como:

$$c_0 = u*f + z_0 \qquad \text{ec. 23}$$

Y la cola de datos del paso 2 en SCN P_1 se actualiza como:

$$c_1 = e*b + z_1 \qquad \text{ec. 24}$$

Según las ecuaciones anteriores, (a • b) se puede calcular como:

$$(a \bullet b) = c_0 + c_1 \quad \text{ec. 25}$$

$$c_0 + c_1 = u*f + e*b + u*v \quad \text{ec. 26}$$

$$c_0 + c_1 = u*b - u*v + a*b - u*b + u*v \quad \text{ec. 27}$$

La FIG. 4A representa un ejemplo de un subproceso 400a para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva. FIG. 4B representa un ejemplo de un subproceso 400b para actualizar iterativamente parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva. Específicamente, las FIG. 4A y 4B ilustran el proceso para realizar el tercer paso de un círculo de iteración (en el presente documento como "paso 3").

En el paso 3, el vector θ columna y cada uno de los SCN P_0 y SCN P_1 pueden obtener una participación del θ actualizado. En este paso, el θ actualizado se puede expresar como:

$$\theta := (1 - \frac{\lambda}{m})\theta - \frac{\alpha}{m} X^T E \quad \text{ec. 28}$$

Donde E es la misma E en la ecuación (19) del paso 2, y donde

$$X^T E = (X^L)^T E \| (X^R)^T E \quad \text{ec. 29}$$

El método para calcular $(X^L)^T E$ es el mismo que el método para calcular $X^L\theta^L$, mientras que el método para calcular $(X^R)^T E$ es el mismo que el método para calcular $X^R\theta^R$, y no se repetirá aquí. En la ecuación (29), X representa una matriz que contiene datos de características de múltiples muestras y E representa un vector de error.

Después del paso 3, se completa un círculo de iteración y el proceso de entrenamiento ingresa al siguiente círculo de iteración y repite los pasos 1-3, o el proceso de entrenamiento se termina si se cumple una condición, como se describe con más detalles a continuación.

La FIG. 5 representa un ejemplo de un proceso 500 para determinar si se deben terminar las iteraciones de actualización de parámetros de un SLRM de acuerdo con implementaciones de esta memoria descriptiva.

En algunas implementaciones, las iteraciones se pueden terminar cuando el número de círculos de iteración completados ha alcanzado un número predeterminado. En algunas implementaciones, se predetermina un umbral, y cuando la diferencia entre dos resultados de iteraciones consecutivas es menor que ese umbral, la iteración finaliza.

Específicamente, por ejemplo, suponiendo que el parámetro antes y después de un círculo de iteración es θ y θ', calcular la diferencia D = (θ' - θ) utilizando SS. Suponiendo que $a_0 = <D^T>_0$, $a_1 = <D^T>_1$, $b_0 = <D>_0$ y $b_1 = <D>_1$. Cada uno del SCN P_0 y del SCN P_1 calcula una participación de D^TD utilizando la multiplicación triple y SS, y la combinación de cada uno de los resultados para obtener D.

Como se muestra en la FIG. 5. En algunas implementaciones, el SCN P_0 genera la matriz u_0 y v_0 aleatoria, y el SCN P_0 genera la matriz u_1 y v_1 aleatoria. El método de cálculo posterior es similar a los métodos descritos previamente en las FIG. 2 y 4, y no se repetirá aquí. Después de los datos intercambiados entre el SCN P_0 y el SCN P_1, el resultado del cálculo en el SCN P_0 es:

$$c_0 = -e*f + a_0 * f + eb_0 + z_0 \quad \text{ec. 30}$$

donde f = b - v.

El resultado del cálculo en el SCN P_1 es:

$$c_1 = a_1 * f + eb_1 + z_1 \quad \text{ec. 31}$$

donde cada uno de c_0 y c_1 es una participación de D^TD.

Según las ecuaciones anteriores, D^TD se puede calcular como:

$$D^TD = c_0 + c_1 \quad \text{ec. 32}$$

$$c_0 + c_1 = e*f + a*f + e*b + u*v \quad \text{ec. 33}$$

$$c_0 + c_1 = u*f + a*b - u*b + u*v \quad \text{ec. 34}$$

$c_0 + c_1 = u*b - u*v + a*b - u*b + u*v$ ec. 35

La FIG. 6 representa un ejemplo de un proceso 600 para entrenar un SLRM de múltiples partes utilizando un procedimiento de modelado seguro interactivo impulsado por eventos de acuerdo con implementaciones de esta memoria descriptiva.

En algunas implementaciones, diversas etapas del procedimiento 600 pueden ejecutarse en paralelo, de manera combinada, en bucle o en cualquier otro orden. Para facilitar la presentación, la siguiente descripción describe de manera genérica el procedimiento 600 en el contexto de las otras figuras en esta descripción. Sin embargo, se entenderá que el método 600 puede realizarse, por ejemplo, mediante cualquier sistema, entorno, software y hardware adecuados, o una combinación de sistemas, entornos, software y hardware, según sea apropiado. Por ejemplo, el método 600 puede realizarse por uno o más aparatos de procesamiento de datos que están configurados para ejecutar algoritmos de aprendizaje automático utilizando SS. El aparato de procesamiento de datos puede incluir o estar implementado por uno o más de, por ejemplo, CPU de uso general o aceleradores de hardware tales como GPU, FPGA e incluso procesadores ASIC personalizados.

En algunas implementaciones, como se ilustra en la FIG. 6, el proceso 600a (incluidos los pasos 602a-622a) se realiza por el Nodo A junto con una base de datos, mientras que el proceso 600b (incluidos los pasos 602b-624b) se realiza por el Nodo B en conjunto con la base de datos. La base de datos puede ser un servidor de diccionario remoto (Redis) y soporta la transmisión y el almacenamiento temporal de datos para los nodos A, B y B. En algunas implementaciones, los datos se pueden transmitir entre los SCN A y B 112, 114 sin pasar por una base de datos y las colas de suscripción pueden almacenarse y actualizarse localmente en cada uno de los nodos A y nodos B. Antes de que comience el proceso de iteración, cada uno de los nodos A y nodos B entra en un proceso de modelado independiente e inicia una iteración por lotes. Cada uno de los nodos A y nodos B puede publicar y suscribir datos, por lo tanto, cada uno de los Nodo A y nodos B es un editor (o "productor") y también un suscriptor (o "consumidor"). Bajo el modelo dirigido por eventos en este ejemplo, incluso el Nodo A y el Nodo B se responden entre sí, su proceso 600a y 600b de modelado son independientes y se desarrollan por separado.

En 602a y 602b, el Nodo A y el Nodo B entran cada uno en una iteración de lotes independiente. 602a y 602b pueden producirse de forma concurrente o consecutiva. Después de 602a, el proceso 600a pasa a 604a. Después de 602b, el proceso 600b pasa a 604b.

En 604a, el Nodo A publica los datos del paso 1 al Nodo B. En algunas implementaciones, los datos del paso 1 se publican enviando datos de muestra cifrados a una base de datos, de modo que la base de datos puede almacenar y actualizar la cola de suscriptores. Una vez que el Nodo A ha completado este paso, se puede enviar una notificación al Nodo B para informar al Nodo B que se ha completado la publicación de los datos del paso 1 del Nodo A. Después de 604a, el proceso 600a pasa a 606a.

En 606 a, el Nodo A se suscribe a los datos del paso 1 del Nodo B. En algunos casos, debido a que el SLRM se entrena en base al SS, el Nodo A no puede completar el entrenamiento en sí mismo y ambos datos de los nodos A y B se utilizan como entrada al modelo. En algunas implementaciones, el Nodo A realiza dicha suscripción solicitando los datos del paso 1 del Nodo B al Nodo B. Si la cola de datos en la base de datos se actualiza con los datos del paso 1 del Nodo B, el Nodo A recibirá una notificación. Debido a que en este momento el Nodo A no ha recibido la notificación del Nodo B, el siguiente paso de 600a no se puede desencadenarse y el proceso 600a se suspende temporalmente.

Mientras tanto, en 604b, el Nodo B publica los datos del paso 1 al Nodo A. En algunas implementaciones, los datos del paso 1 se publican enviando datos de muestra cifrados a una base de datos, por lo que la base de datos puede almacenar y actualizar la cola de suscriptores. Una vez que el Nodo B completa este paso, se puede enviar una notificación al Nodo A para informar al Nodo A que se ha completado la publicación de datos del paso 1 del Nodo B. Después de 604b, el proceso 600b pasa a 606b.

En 606b, el Nodo B se suscribe a los datos del paso 1 del Nodo B. Debido a que en este momento el Nodo A ya publicó los datos del paso 1 y el Nodo B ya recibió una notificación del Nodo A, se desencadena el siguiente paso y, después de 606b, el proceso 600b continúa a 608b.

En 608b, el Nodo B calcula localmente un resultado del paso 1 utilizando sus propios datos y los datos del paso 1 recibidos del Nodo A. Las ecuaciones y los pasos detallados de este paso se puede hacer referencia a la FIG. 2B. Después de 608b, el proceso 600b pasa a 610b.

Debido a que el Nodo B puede realizar los pasos 610b y 612b de forma independiente, el Nodo B puede realizar estos dos pasos juntos o consecutivamente sin depender de la acción del Nodo A. Después de los pasos 610b y 612b, la cola de suscriptores en la base de datos se actualiza utilizando los datos del paso 2 publicados por el Nodo B, y se envía una notificación de publicación al Nodo A. Después del paso 612b, debido a que el Nodo B no ha recibido la notificación del Nodo A, el siguiente paso no se puede desencadenar y el proceso 600b se suspende temporalmente.

Volviendo al paso 606a del proceso 600a. Si en el momento en que el nodo B ya realizó el paso 604b, y el nodo A ya ha recibido notificaciones del nodo B, se desencadena el siguiente paso del proceso 600a y, después de 606a, el proceso 600a pasa a 608a.

Debido a que los pasos 608a-612a pueden realizarse de forma independiente por el Nodo A sin datos del Nodo B, el Nodo A puede realizar estos pasos consecutivamente. Después del paso 610a, la cola de suscriptores en la base de datos se actualiza utilizando los datos del paso 2 publicados por el Nodo A, y se envía una notificación de publicación al Nodo B. Después del paso 612a, si el Nodo B ya realizó el paso 610b y el Nodo A recibió la notificación correspondiente, se desencadena el siguiente paso de 600a y el proceso 600a pasa a 614a. De lo contrario, el proceso 600a se suspende temporalmente hasta que se realiza el paso 610b.

Dado que los pasos 614a -618a pueden realizares de forma independiente por el Nodo A, el Nodo A puede realizar estos pasos de forma consecutiva. Después de los pasos 614a y 616a, la cola de suscriptores en la base de datos se actualiza utilizando los datos del paso 3 publicados por el Nodo A, y se envía una notificación de publicación al Nodo B. Después del paso 618a, debido a que el Nodo A no ha recibido notificación del Nodo B, el proceso 600a puede suspenderse temporalmente hasta que se reciba la notificación del Nodo B.

Volviendo al proceso 600b. Después del paso 610a en el que el nodo A publicó sus datos del paso 2 y envió la notificación correspondiente, se desencadena el siguiente paso del proceso 600b y el proceso 600b puede pasar a 614b. En 614b, el Nodo B calcula localmente el resultado del paso 2 utilizando sus propios datos y los datos del paso 2 recibidos del Nodo A. Las ecuaciones y los pasos detallados del paso de cálculo se puede hacer referencia a la FIG. 3. Después de 614b, el proceso 600b pasa a 616b.

Debido a que el Nodo B puede realizar cada uno de los pasos de 616b-620b de forma independiente, el Nodo B puede realizar estos tres pasos consecutivamente. En 616b, se calcula un valor de sigmoide (wx) - y utilizando la ecuación (20). Después de los pasos 618b y 620b, la cola de suscriptores en la base de datos se actualiza utilizando los datos del paso 3 publicados por el Nodo B, y se envía una notificación de publicación al Nodo A. Después del paso 620b, si en ese momento el Nodo A ya realizó 616a y el nodo B ha recibido una notificación del nodo A, se desencadena el siguiente paso de 600b y el proceso 600b pasa a 622b.

En 622b, el Nodo B calcula localmente los resultados del paso 3 utilizando sus propios datos y 3 los datos del paso recibidos del Nodo A. Las ecuaciones y los pasos detallados del cálculo se detalla en la FIG. 4B previamente descrita. Después de 622b, debido a que el Nodo A y el Nodo B determinan conjuntamente si entrar en la siguiente iteración, después de 622b y, debido a que el Nodo B no ha recibido notificación del Nodo B, el proceso 600b se suspende temporalmente.

Volviendo al proceso 600a, después del paso 618b en el que el Nodo B ha publicado sus datos del paso 3 y enviado la notificación correspondiente, se desencadena el siguiente paso del proceso 600a y el proceso 600a puede pasar a 620a. En 620a, el Nodo A calcula localmente el resultado del paso 3 utilizando sus propios datos y los datos del paso 3 recibidos del nodo A. Las ecuaciones y los pasos detallados del paso de cálculo se puede hacer referencia a la FIG. 3. Después de 620a, tanto el Nodo A como el Nodo B han completado un círculo de iteración.

Como se describió previamente en la FIG. 4, si se termina el círculo de iteración depende de si se satisface una condición predeterminada. En caso afirmativo, ambos procesos 600a y 600b pasan a 622a y 624a y repiten los pasos de 602a-620a y 602b-622b nuevamente. En caso negativo, la iteración se termina y ambos procesos 600a y 600b se detienen en 618a y 622b, respectivamente.

La FIG. 7 representa un ejemplo de un proceso 700 que puede ejecutarse de acuerdo con implementaciones de la presente memoria descriptiva. En algunas implementaciones, diversas etapas del procedimiento 700 pueden ejecutarse en paralelo, de manera combinada, en bucle o en cualquier otro orden. Para facilitar la presentación, la siguiente descripción describe de manera genérica el procedimiento 700 en el contexto de las otras figuras en esta descripción. Sin embargo, se entenderá que el método 700 puede realizarse, por ejemplo, mediante cualquier sistema, entorno, software y hardware adecuados, o una combinación de sistemas, entornos, software y hardware, según sea apropiado. Por ejemplo, el método 700 puede realizarse por uno o más aparatos de procesamiento de datos que están configurados para ejecutar algoritmos de aprendizaje automático utilizando SS. El aparato de procesamiento de datos puede incluir o estar implementado por uno o más de, por ejemplo, CPU de uso general o aceleradores de hardware tales como GPU, FPGA e incluso procesadores ASIC personalizados.

En 702, los datos de entrenamiento de muestra para un SLRM se dividen en dos participaciones utilizando SS y cada una de las participaciones se distribuye a un SCN. En algunas implementaciones, una matriz X de datos de muestra se divide en X^L y X^R, y la matriz X^L de datos de muestra pertenece a un SCN 1 y la matriz X^R de datos de muestra pertenece a SCN 2. Después de 702, el proceso 700 pasa a 704.

En 704, los parámetros del SLRM se actualizan iterativamente utilizando cada una de las participaciones de los datos de entrenamiento de muestra. En algunas implementaciones, una función objetivo del SLRM se optimiza utilizando un método de descenso de gradiente estocástico (SGD) de mini lotes.

En algunas implementaciones, el SLRM se basa en una función de regresión logística que se puede expresar como:

$$\theta[j] := \theta[j] - \frac{\alpha}{m}\sum_{i=1}^{m}(sigmoide(X_i^L \cdot \theta^L + X_i^R \cdot \theta^R) - Y_i)X_i[j] - \frac{\lambda}{m}\theta[j]$$

donde m representa el tamaño de la muestra del SGD del mini lotes. X representa una matriz de muestra de m $*$ k. Cada una de las filas de la matriz X representa una muestra, y X_i representa la $i^{-ésima}$ fila de la matriz X, [j] representa el $j^{-ésimo}$ elemento del vector X_i. La matriz X de datos de muestra se divide en X^L y ^{XR}, y los datos X^L de muestra pertenecen a SCN P_0, mientras que X^R pertenecen a SCN P_1. Θ representa un vector columna de parámetros y el vector θ columna puede dividirse verticalmente a θ^L y θ^R.

En algunas implementaciones, entrenar el SLRM incluye el uso de un modelo dirigido por eventos. Después de 704, el proceso 700 pasa a 706.

En 706, para cada uno de los SCN, se calcula una participación de A en función de:

$$A = X\Theta = X^L\theta^L + X^R\theta^R$$

En algunas implementaciones, el cálculo de $X^L\theta^L$ y $X^R\theta^R$ incluye, ocultar datos originales proporcionadas desde cada uno de los SCN utilizando números aleatorios, y el intercambio de datos ocultos entre los SCN. En algunas implementaciones, antes de ocultar los datos originales, cada uno de los SCN obtiene números aleatorios. En algunas implementaciones, el número aleatorio puede ser un número, un vector o una matriz. En alguna implementación, el SCN puede generar los números aleatorios. En algunas implementaciones, el número aleatorio se puede solicitar y recibir de una agencia de terceros. Después de 706, el proceso 700 pasa a 708.

En 708, se calcula un vector columna de un error de predicción en función de:

$$E = g(A) - Y_i$$

donde g es una función de ajuste de una función sigmoide, que puede ser un ajuste de función polinómica. En algunas implementaciones, si el orden o g(x) es 7, g(x) se puede expresar mediante la ecuación:

$$g(x) = 0.5 + 1.73496*(x/8) - 4.19407*(x/8)^3 + 5.43402*(x/8)^5 - 2.50739*(x/8)^7$$

donde g(A) representa el vector columna compuesto por g(A[1]), g(A[2]),... g(A[m]). Después de 708, el proceso 700 pasa a 710.

En 710, el vector θ columna actualizado se puede expresar como:

$$\theta := (1 - \frac{\lambda}{m})\theta - \frac{\alpha}{m}X^T E$$

donde $X^T E = (X^L)^T E \parallel (X^R)^T E$

Después de 710, el proceso 700 pasa a 712.

En 712, la iteración finaliza si se satisface una condición predeterminada. En algunas implementaciones, las iteraciones se pueden terminar cuando el número de ciclos de iteración completados ha alcanzado un número predeterminado.

En algunas implementaciones, se predetermina un umbral, y cuando la diferencia entre dos resultados de iteraciones consecutivas es menor que ese umbral, la iteración finaliza. Después de 712, el proceso 700 puede detenerse.

La FIG. 8 representa un ejemplo de un proceso 800 que puede ejecutarse de acuerdo con implementaciones de la presente memoria descriptiva. En algunas implementaciones, diversas etapas del procedimiento 800 pueden ejecutarse en paralelo, de manera combinada, en bucle o en cualquier otro orden. Para facilitar la presentación, la siguiente descripción describe de manera genérica el procedimiento 800 en el contexto de las otras figuras en esta descripción. Sin embargo, se entenderá que el método 800 puede realizarse, por ejemplo, mediante cualquier sistema, entorno, software y hardware adecuados, o una combinación de sistemas, entornos, software y hardware, según sea apropiado. Por ejemplo, el método 800 puede realizarse por uno o más aparatos de procesamiento de datos que están configurados para ejecutar algoritmos de aprendizaje automático utilizando SS. El aparato de procesamiento de datos puede incluir o estar implementado por uno o más de, por ejemplo, CPU de uso general o aceleradores de hardware tales como GPU, FPGA e incluso procesadores ASIC personalizados.

En 802, un usuario inicializa el SLRM determinando una serie de parámetros asociados con el SLRM. En algunas implementaciones, los parámetros pueden incluir el tamaño de los datos de entrenamiento, características asociadas con los datos de entrenamiento, correspondientes a los ajustes del SLRM, hiperparámetros del SLRM

En algunas implementaciones, después de que el usuario ha inicializado el SLRM, la información asociada con el SLRM se envía a un agente de gestión de seguridad. En algunas implementaciones, el agente de gestión de seguridad procesa el SLRM y los datos de entrenamiento que se alimentan al modelo. En algunas implementaciones, un agente de gestión de nodos está configurado para seleccionar y gestionar los SCN que participan en el entrenamiento. En algunas implementaciones, tanto el agente de gestión de seguridad como el agente de gestión de nodos pueden estar configurados como agentes de terceros. Después de 802, el proceso 800 pasa a 804.

En 804, los números aleatorios se solicitan por al menos dos SCN de un proveedor de números aleatorios, y los números aleatorios solicitados se utilizan para cifrar los datos almacenados en cada uno de los SCN. En algunas implementaciones, el número aleatorio puede ser un número, un vector o una matriz, etc. En algunas implementaciones, los números aleatorios pueden generarse por al menos un SCN. En algunas implementaciones, los números aleatorios pueden proporcionarse por un tercero independiente.

En algunas implementaciones, cifrar los datos incluye realizar cálculos utilizando los datos y los números aleatorios recibidos. En algunas implementaciones, los cálculos realizados son al menos uno de suma, resta y multiplicación. Después de 802, el proceso 800 pasa a 804.

En 806, los datos cifrados se utilizan como entrada para actualizar iterativamente los parámetros de un SLRM en base a SS. Los detalles de este paso se describirán con más detalle en la FIG. 7. Después de 806, el proceso 800 pasa a 808.

En 808, la salida del SLRM entrenado se utiliza para realizar un servicio para cada uno de los SCN. En algunas implementaciones, el servicio puede ser un servicio de predicción o un servicio de clasificación. Después de 808, el proceso 800 se detiene.

La FIG. 9 representa un ejemplo de un diagrama 900 que ilustra módulos de un aparato de acuerdo con implementaciones de la memoria descriptiva. El aparato 900 puede ser una implementación de ejemplo de un aparato para entrenar modelos de regresión logística seguros de múltiples partes. El aparato 900 puede corresponder a las implementaciones descritas anteriormente, y el aparato 900 incluye lo siguiente: un receptor o unidad 902 de recepción para recibir, en una pluralidad de nodos de cálculo seguro (SCN), una pluralidad de números aleatorios de un proveedor de números aleatorios; un cifrador o unidad 904 de cifrado para cifrar, en cada uno de los SCN, los datos almacenados en el SCN utilizando los números aleatorios recibidos; y actualizador o unidad de 906 actualización para actualizar iterativamente un modelo de regresión logística seguro (SLRM) utilizando los datos cifrados de cada uno de los SCN; y un generador o unidad 908 de salida para generar un resultado del SLRM, en donde el resultado está configurado para permitir que cada uno de los SCN realice un servicio después de actualizar iterativamente el SLRM.

En una implementación opcional, cada uno de los números aleatorios es al menos uno de un número, un vector o una matriz.

En una implementación opcional, al menos uno de los números aleatorios se genera por un agente externo.

En una implementación opcional, el actualizador o unidad 906 de actualización se utiliza para actualizar iterativamente el SLRM en base a un esquema de intercambio de secretos (SS).

En una implementación opcional, el actualizador o unidad 906 de actualización se utiliza para actualizar iterativamente el SLRM en base a un modelo dirigido por eventos.

En una implementación opcional, el actualizador o unidad 906 de actualización se utiliza para actualizar iterativamente el SLRM determinando un vector columna de un error de predicción en base a la ecuación (19).

En una implementación opcional, el aparato 900 incluye además un inicializador o unidad de inicialización para inicializar parámetros asociados con el SLRM antes de recibir una pluralidad de números aleatorios del proveedor de números aleatorios.

En una implementación opcional, la actualización iterativa del SLRM continúa hasta que la diferencia entre dos resultados de iteraciones consecutivas es menor que un umbral predeterminado.

El sistema, aparato, módulo o unidad ilustrado en las implementaciones anteriores puede implementarse utilizando un chip informático o una entidad, o puede implementarse utilizando un producto que tenga una determinada función. Un dispositivo de implementación típico es una computadora, y la computadora puede ser una computadora personal, una computadora portátil, un teléfono móvil, un teléfono con cámara, un teléfono inteligente, un asistente digital personal, un reproductor multimedia, un dispositivo de navegación, un dispositivo de recepción y envío de correo electrónico, una consola de juegos, una computadora tableta, un dispositivo ponible o cualquier combinación de estos dispositivos.

Para un proceso de implementación de funciones y roles de cada una de las unidades en el aparato, se pueden hacer referencias a un proceso de implementación de los pasos correspondientes en el método anterior. Los detalles se omiten aquí por simplicidad.

Debido a que la implementación de un aparato corresponde básicamente a la implementación de un método, para partes relacionadas, se pueden hacer referencias a descripciones relacionadas en la implementación del método. La implementación del aparato descrita anteriormente es simplemente un ejemplo. Las unidades descritas como partes separadas pueden estar o no físicamente separadas, y las partes mostradas como unidades pueden o no ser unidades físicas, pueden estar ubicadas en una posición o pueden estar distribuidas en varias unidades de red. Algunos o todos los módulos pueden seleccionarse en base a las demandas reales para lograr los objetivos de las soluciones de la memoria descriptiva. Un experto en la técnica puede entender y poner en práctica las formas de realización de la presente solicitud sin esfuerzos creativos.

Haciendo referencia nuevamente a la FIG. 9, se puede interpretar como que ilustra un módulo funcional interno y una estructura de un aparato para procesar datos utilizando un marco de trabajo de entrenamiento de modelo de seguridad distribuido de múltiples partes para la protección de privacidad. El aparato de ejecución puede ser un ejemplo de un aparato configurado para permitir el procesamiento de datos.

Las implementaciones de la materia objeto y las acciones y operaciones descritas en esta memoria descriptiva se pueden implementar en circuitería electrónica digitales, en software o firmware de computadora incorporados de manera tangible, en hardware de computadora, incluidas las estructuras descritas en esta memoria descriptiva y sus equivalentes estructurales, o en combinaciones de uno o más de ellos. Las implementaciones de la materia objeto descrita en esta memoria descriptiva se pueden implementar como uno o más programas informáticos, p. ej., uno o más módulos de instrucciones de programa informático, codificados en un soporte de programa informático, para su ejecución por, o para controlar la operación de, aparatos de procesamiento de datos. El soporte puede ser un medio de almacenamiento informático tangible no transitorio. Alternativa o adicionalmente, el soporte puede ser una señal propagada generada artificialmente, p. ej., una señal eléctrica, óptica o electromagnética generada por una máquina, que se genera para codificar información para la transmisión a un aparato receptor adecuado para la ejecución por un aparato de procesamiento de datos. El medio de almacenamiento informático puede ser, o ser parte de, un dispositivo de almacenamiento legible por máquina, un sustrato de almacenamiento legible por máquina, un dispositivo de memoria de acceso aleatorio o en serie, o una combinación de uno o más de ellos. Un medio de almacenamiento informático no es una señal propagada.

El término "aparato de procesamiento de datos" abarca todo tipo de aparatos, dispositivos y máquinas para procesar datos, incluyendo, a modo de ejemplo, un procesador programable, una computadora o múltiples procesadores u computadoras. El aparato de procesamiento de datos puede incluir circuitería lógica de propósito especial, p. ej., una FPGA (matriz de puertas programables en campo), un ASIC (circuito integrado de aplicación específica) o una GPU (unidad de procesamiento de gráficos). El aparato también puede incluir, además del hardware, código que crea un entorno de ejecución para programas informáticos, p. ej., código que constituye el firmware del procesador, una pila de protocolo, un sistema de gestión de bases de datos, un sistema operativo o una combinación de uno o más de ellos.

Un programa informático, que también puede denominarse o describirse como un programa, software, una aplicación de software, una app, un módulo, un módulo de software, un motor, una secuencia de comandos o código, se puede escribir en cualquier forma de lenguaje de programación, incluidos los lenguajes compilados o interpretados, o los lenguajes declarativos o procedimentales; y se puede desplegar en cualquier forma, incluso como un programa independiente o como un módulo, componente, motor, subrutina u otra unidad adecuada para ejecutarse en un entorno informático, cuyo entorno puede incluir una o más computadoras interconectadas por una red de comunicaciones de datos en una o más ubicaciones.

Un programa informático puede, pero no necesariamente, corresponder a un archivo en un sistema de archivos. Un programa informático puede estar almacenado en una porción de un archivo que contiene otros programas o datos, p. ej., una o más secuencias de comandos almacenadas en un documento de lenguaje de marcado, en un solo archivo dedicado al programa en cuestión, o en múltiples archivos coordinados, p. ej., archivos que almacenan uno o más módulos, subprogramas o porciones de código.

Los procesos y flujos lógicos descritos en esta memoria descriptiva pueden realizarse por una o más computadoras que ejecutan uno o más programas informáticos para realizar operaciones operando con datos de entrada y generando salida. Los procesos y flujos lógicos también se pueden realizar mediante circuitería lógica de propósito especial, p. ej., una FPGA, un ASIC o una GPU, o mediante una combinación de circuitería lógica de propósito especial y uno o más computadoras programadas.

Las computadoras adecuadas para la ejecución de un programa informático pueden estar basadas en microprocesadores de propósito general o especial o ambos, o cualquier otro tipo de unidad central de procesamiento. Generalmente, una unidad central de procesamiento recibirá instrucciones y datos de una memoria de solo lectura o una memoria de acceso aleatorio o ambas. Los elementos de una computadora pueden incluir una unidad central de

procesamiento para ejecutar instrucciones y uno o más dispositivos de memoria para almacenar instrucciones y datos. La unidad central de procesamiento y la memoria pueden complementarse o incorporarse en circuitería lógica de propósito especial.

Generalmente, una computadora también incluirá, o estará operativamente acoplada para recibir datos o transferir datos a uno o más dispositivos de almacenamiento masivo. Los dispositivos de almacenamiento masivo pueden ser, por ejemplo, discos magnéticos, magneto-ópticos u ópticos, o unidades de estado sólido. Sin embargo, una computadora no necesita tener tales dispositivos. Además, una computadora puede integrarse en otro dispositivo, p. ej., un teléfono móvil, un asistente digital personal (PDA), un reproductor de audio o vídeo móvil, una consola de juegos, un receptor del sistema de posicionamiento global (GPS) o un dispositivo de almacenamiento portátil, p. ej., una unidad flash de bus serie universal (USB), por nombrar solo algunas.

Para facilitar la interacción con un usuario, las implementaciones de la materia objeto descrita en esta memoria descriptiva se pueden implementar o configurar para comunicarse con una computadora que tenga un dispositivo de visualización, p. ej., un monitor LCD (pantalla de cristal líquido), para visualizar información al usuario, y un dispositivo de entrada mediante el cual el usuario puede proporcionar entrada a la computadora, p. ej., un teclado y un dispositivo señalador, p. ej., un ratón, una bola de seguimiento o un panel táctil. También se pueden utilizar otros tipos de dispositivos para proporcionar la interacción con un usuario; por ejemplo, la retroalimentación proporcionada al usuario puede ser cualquier forma de retroalimentación sensorial, p. ej., retroalimentación visual, retroalimentación auditiva o retroalimentación táctil; y la entrada del usuario se puede recibir de cualquier forma, incluida la entrada acústica, de voz o táctil. Además, una computadora puede interactuar con un usuario enviando documentos a y recibiendo documentos desde un dispositivo que se utiliza por el usuario; por ejemplo, enviando páginas web a un navegador web en el dispositivo de un usuario en respuesta a solicitudes recibidas desde el navegador web, o interactuando con una app que se ejecuta en un dispositivo de usuario, p. ej., un teléfono inteligente o tableta electrónica. Además, una computadora puede interactuar con un usuario enviando mensajes de texto u otras formas de mensaje a un dispositivo personal, p. ej., un teléfono inteligente que está ejecutando una aplicación de mensajería, y recibiendo de vuelta mensajes de respuesta del usuario.

Si bien esta memoria descriptiva contiene muchos detalles de implementación específica, estos no deben interpretarse como limitaciones en el alcance de lo que se reivindica, que está definido por las propias reivindicaciones, sino más bien como descripciones de características que pueden ser específicas de implementaciones particulares. Ciertas características que se describen en esta memoria descriptiva en el contexto de implementaciones separadas también se pueden realizar en combinación en una sola implementación. A la inversa, diversas características que se describen en el contexto de una sola implementación también se pueden realizar en múltiples implementaciones por separado o en cualquier subcombinación adecuada. Además, aunque las características pueden describirse anteriormente como que actúan en ciertas combinaciones e incluso inicialmente reivindicarse como tales, una o más características de una combinación reivindicada pueden en algunos casos eliminarse de la combinación, y la reivindicación puede estar dirigida a una subcombinación o variación de una subcombinación.

De manera similar, si bien las operaciones se describen en los dibujos y se enumeran en las reivindicaciones en un orden particular, esto no debe entenderse como que requiere que tales operaciones se realicen en el orden particular mostrado o en orden secuencial, o que se realicen todas las operaciones ilustradas, para lograr resultados deseables. En determinadas circunstancias, la multitarea y el procesamiento en paralelo pueden resultar ventajosos. Además, la separación de diversos módulos y componentes del sistema en las implementaciones descritas anteriormente no debe entenderse que requiera dicha separación en todas las implementaciones, y debe entenderse que los componentes y sistemas del programa descritos generalmente se pueden integrar juntos en un solo producto de software o empaquetarse en múltiples productos de software.

Se han descrito implementaciones particulares de la materia objeto. Otras implementaciones están dentro del alcance de las siguientes reivindicaciones. Por ejemplo, las acciones enumeradas en las reivindicaciones se pueden realizar en un orden diferente y aún así lograr resultados deseables. Como ejemplo, los procesos representados en las figuras adjuntas no requieren necesariamente el orden particular mostrado, o el orden secuencial, para lograr resultados deseables. En algunos casos, la multitarea y el procesamiento en paralelo pueden resultar ventajosos.

REIVINDICACIONES

1. Un método implementado por computadora, que comprende:
recibir, mediante un primer nodo de cálculo seguro, SCN, de una pluralidad de nodos de cálculo seguro, SCN, un primer número aleatorio de una pluralidad de números aleatorios de un proveedor de números aleatorios;
cifrar, por el primer SCN, datos almacenados en el SCN utilizando el primer número aleatorio recibido para generar un primer dato cifrado;
transmitir, por el primer SCN, los primeros datos cifrados a un segundo SCN de la pluralidad de SCN;
recibir, por el primer SCN, un segundo dato cifrado del segundo SCN, los segundos datos cifrados, cifrados utilizando un segundo número aleatorio de la pluralidad de números aleatorios del proveedor de números aleatorios;
actualizar iterativamente, por el primer SCN, un modelo de regresión logística seguro, SLRM, utilizando los primeros datos cifrados y los segundos datos cifrados en base a un modelo dirigido por eventos que es un modelo de publicación-suscripción, en donde un editor envía una notificación cuando su servicio se completa, en donde un suscriptor supervisa la notificación del editor, y en donde se activa un servicio del suscriptor al recibir la notificación del editor,
En donde la actualización iterativa del SLRM comprende determinar un vector columna de un error de predicción en base a la ecuación:
$$E = g(A) - Y_i;$$
en donde g representa una función de ajuste de una función sigmoide, en donde A representa una participación del vector columna calculada utilizando un vector de muestra de mini lotes y en donde Y_i representa una etiqueta de la $i^{ésima}$ muestra del vector de muestra de mini lotes; y
después de actualizar iterativamente el SLRM, publicar un resultado del SLRM, en donde el resultado está configurado para permitir que cada uno de los SCN de la pluralidad de SCN realice un servicio.

2. El método implementado por computadora de la reivindicación 1, en donde cada uno de los números aleatorios es al menos uno de un número, un vector o una matriz.

3. El método implementado por computadora de la reivindicación 1, en donde al menos uno de los números aleatorios se genera por un agente externo.

4. El método implementado por computadora de la reivindicación 1, en donde la actualización iterativa del SLRM es en basa a un esquema de intercambio de secretos, SS.

5. El método implementado por computadora de la reivindicación 1, que además comprende:
antes de recibir una pluralidad de números aleatorios del proveedor de números aleatorios, inicializar los parámetros asociados con el SLRM.

6. El método implementado por computadora de la reivindicación 1, en donde la actualización iterativa del SLRM continúa hasta que una diferencia entre dos resultados de iteraciones consecutivas es menor que un umbral predeterminado.

7. Un medio de almacenamiento legible por computadora no transitorio acoplado a una o más computadoras y que tiene instrucciones sobre los mismos ejecutables por la una o más computadoras para realizar un método de acuerdo con una cualquiera de las reivindicaciones anteriores.

8. Un sistema que comprende:
una o más computadoras; y
una o más memorias legibles por computadora acopladas a la una o más computadoras y que tienen instrucciones sobre las mismas ejecutables por la una o más computadoras para realizar un método de acuerdo con una cualquiera de las reivindicaciones 1 a 6.

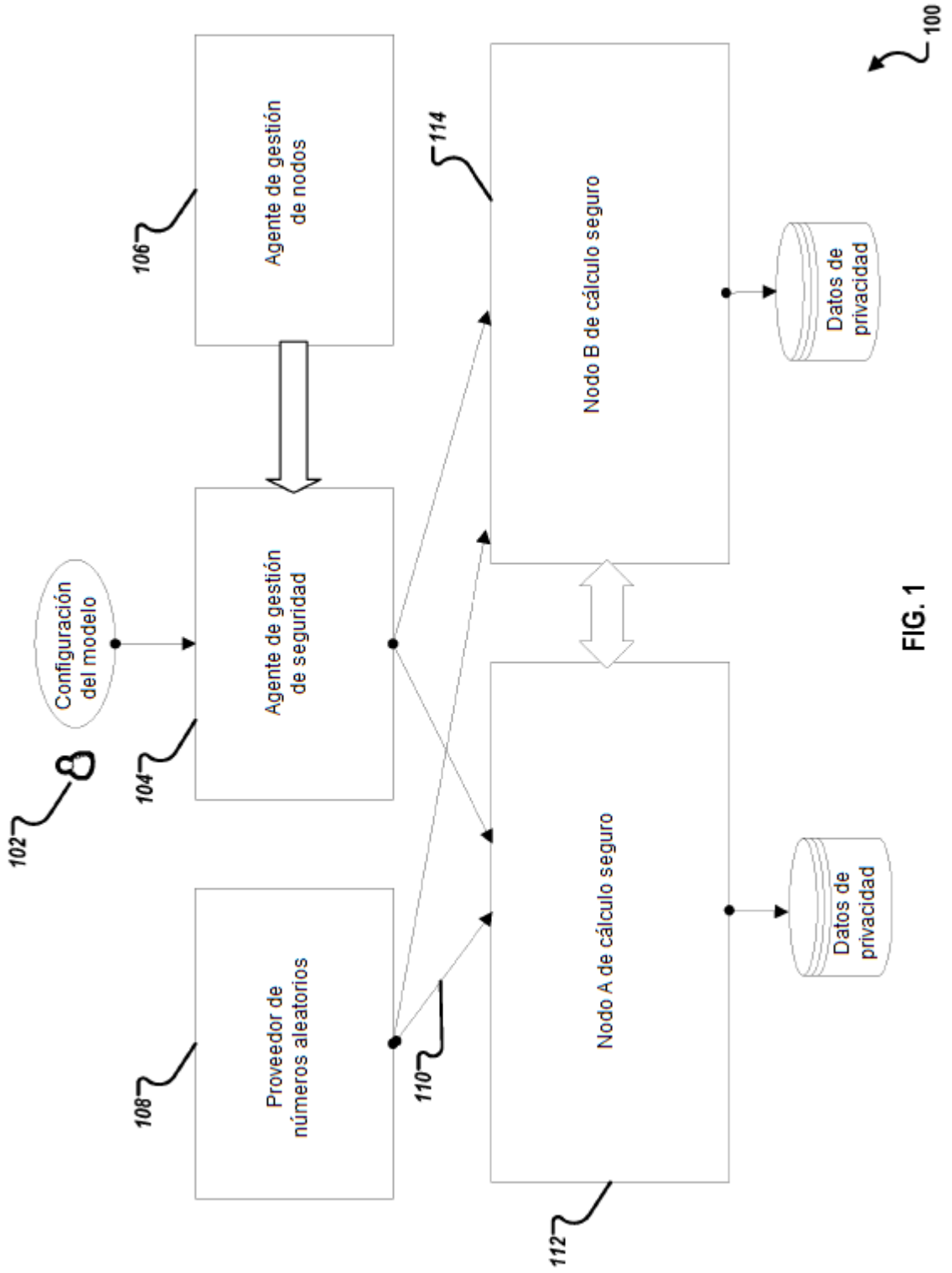
100
Agente de gestión de nodos
106
114
Nodo B de cálculo seguro
Datos de privacidad
Configuración del modelo
102
Agente de gestión de seguridad
104
Proveedor de números aleatorios
108
110
Nodo A de cálculo seguro
Datos de privacidad
112

FIG. 1

P0
P1
participación predeterminada de uv
$z_0 + z_1 = uv = u(v_0 + v_1)$
predeterminado
u, v_0, z_0
predeterminado
v_1, z_1
suponer que, $a = X^t$, $b_0 = \langle \theta^L \rangle_0$
suponer que, $b_1 = \langle \theta^L \rangle_1$
$e = a - u, b_0 - v_0$
$b_1 - v_1$
$f = b - v$
$f = b - v$
$c_0 = uf + eb_0 + z_0$
$c_0 + c_1 = uf + eb + uv$
$c_1 = eb_1 + z_1$
$c_0 + c_1 = ub - uv + ab - ub + uv$
$c_0 + c_1 = ab$
200a

FIG. 2A

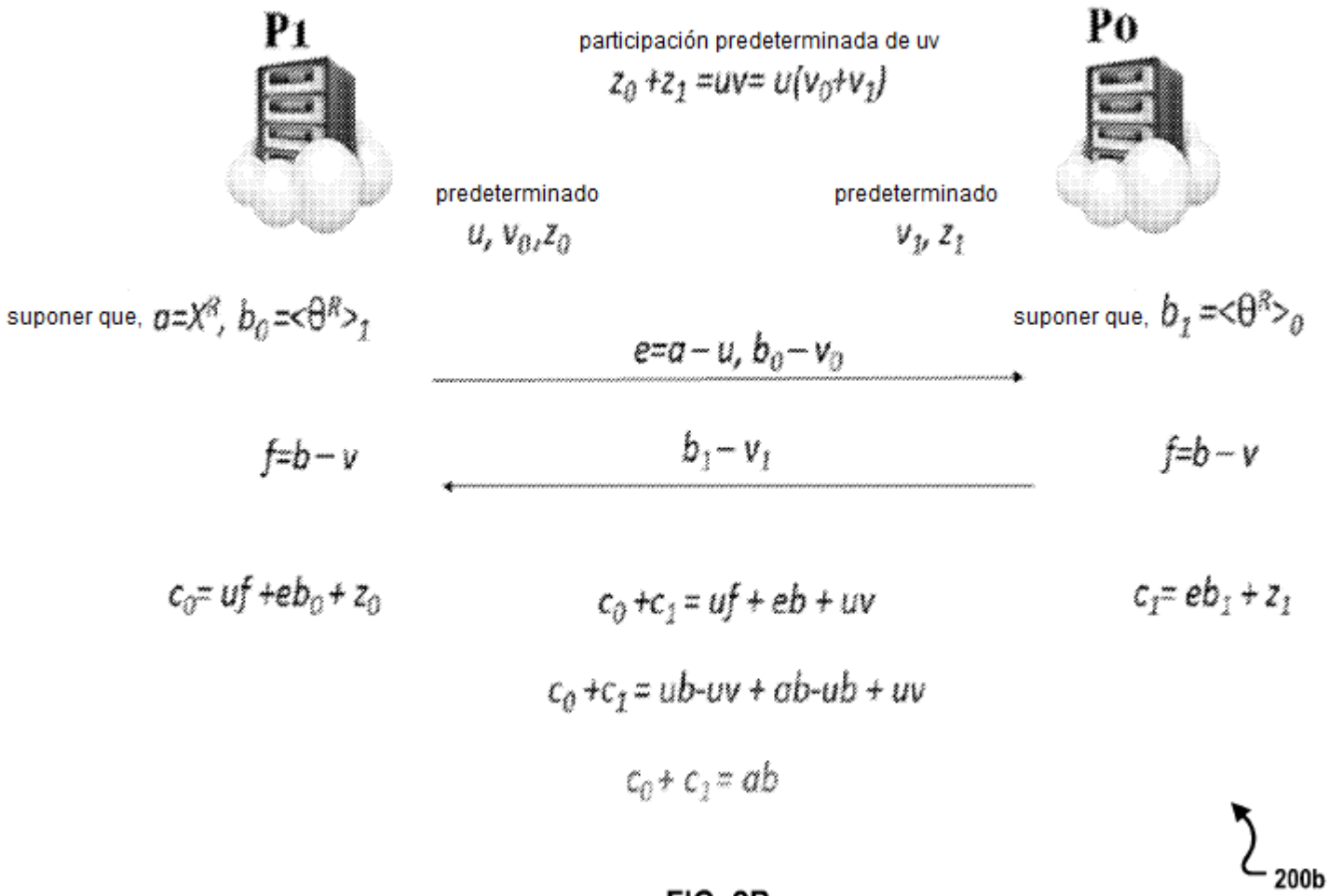
P1
participación predeterminada de uv
$z_0 + z_1 = uv = u(v_0 + v_1)$
P0
predeterminado
u, v_0, z_0
predeterminado
v_1, z_1
suponer que, $a = X^R$, $b_0 = \langle \theta^R \rangle_1$
suponer que, $b_1 = \langle \theta^R \rangle_0$
$e = a - u, b_0 - v_0$
$f = b - v$
$b_1 - v_1$
$f = b - v$
$c_0 = uf + eb_0 + z_0$
$c_0 + c_1 = uf + eb + uv$
$c_1 = eb_1 + z_1$
$c_0 + c_1 = ub - uv + ab - ub + uv$
$c_0 + c_1 = ab$
200b

FIG. 2B

P0
P1
participación predeterminada de uv
$z_0 + z_1 = u \bullet v$
predeterminado
u, z_0
predeterminado
v, z_1
a
b
$e = a - u$
$f = b - v$
$c_0 = u \bullet f + z_0$
$c_1 = e \bullet b + z_1$
$c_0 + c_1 = u \bullet f + e \bullet b + u \bullet v$
$c_0 + c_1 = u \bullet b - u \bullet v + a \bullet b - u \bullet b + u \bullet v$
$c_0 + c_1 = a \bullet b$
300

FIG. 3

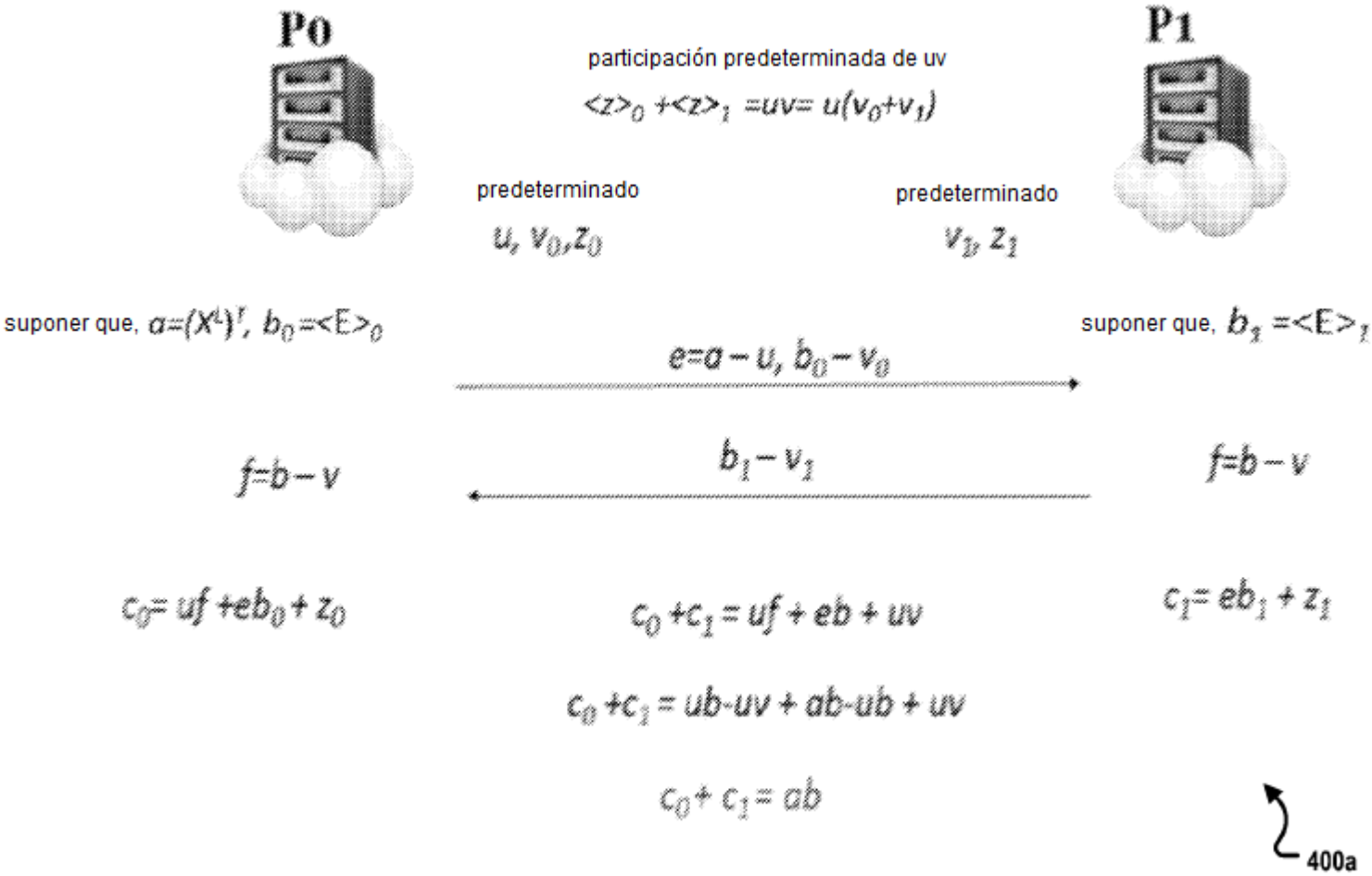

FIG. 4A

ES 2 870 706 T3

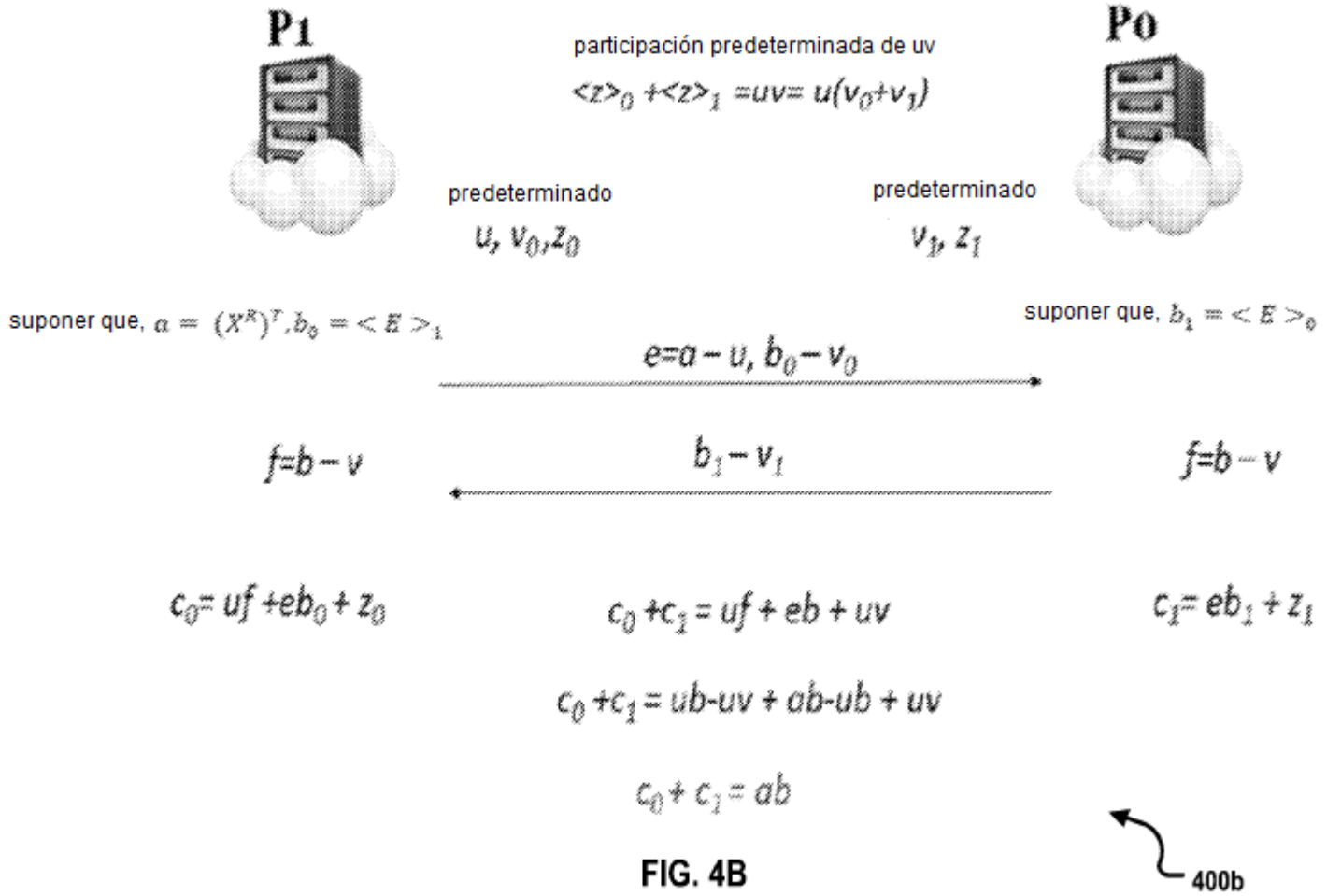
P1
P0
participación predeterminada de uv
$<z>_0 + <z>_1 = uv = u(v_0+v_1)$
predeterminado
u, v_0, z_0
predeterminado
v_1, z_1
suponer que, $a = (X^R)^T, b_0 = < E >_1$
suponer que, $b_1 = < E >_0$
$e=a-u, b_0-v_0$
$f=b-v$
b_1-v_1
$f=b-v$
$c_0 = uf + eb_0 + z_0$
$c_0 + c_1 = uf + eb + uv$
$c_1 = eb_1 + z_1$
$c_0 + c_1 = ub - uv + ab - ub + uv$
$c_0 + c_1 = ab$
400b

FIG. 4B

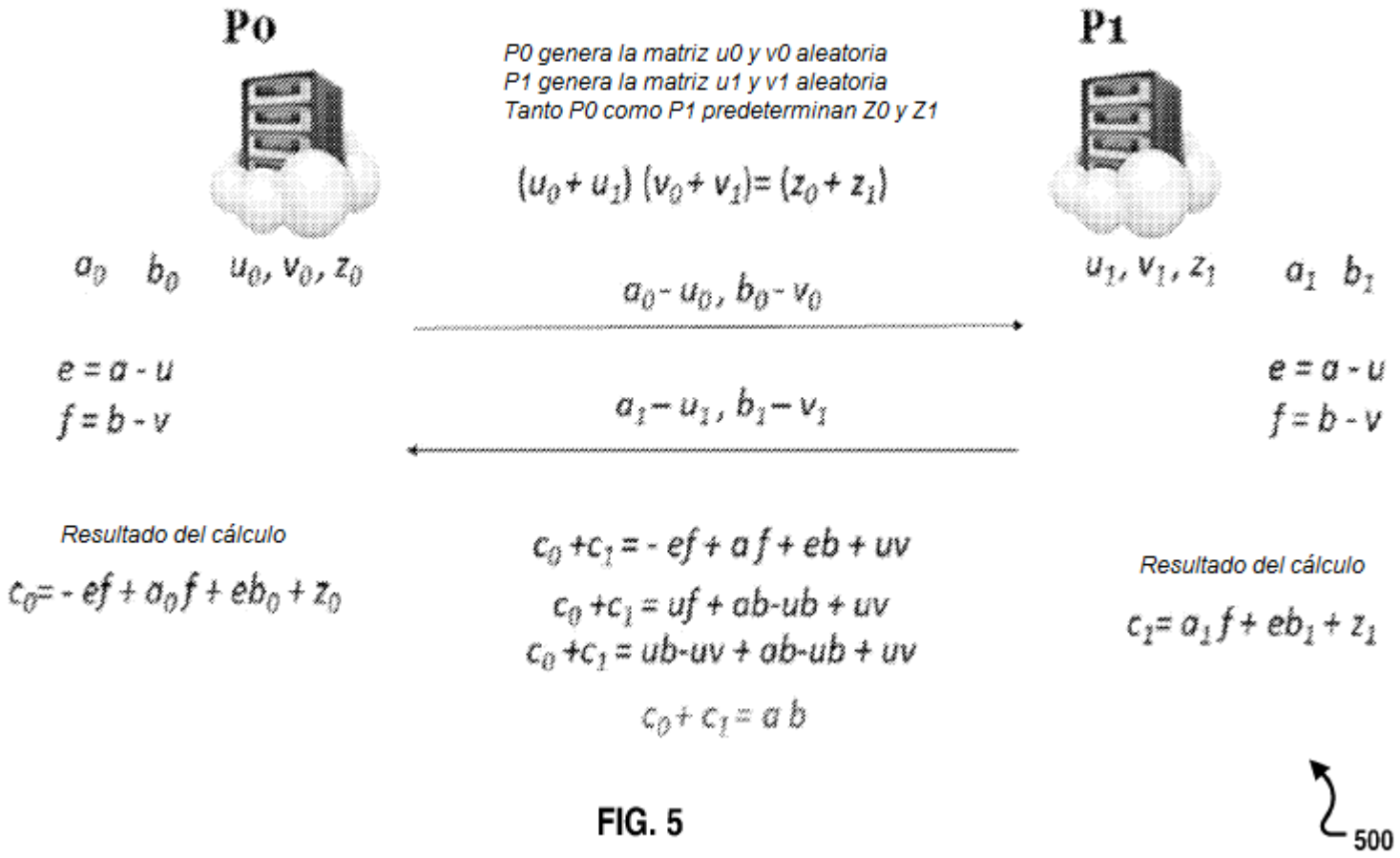

FIG. 5

ES 2 870 706 T3

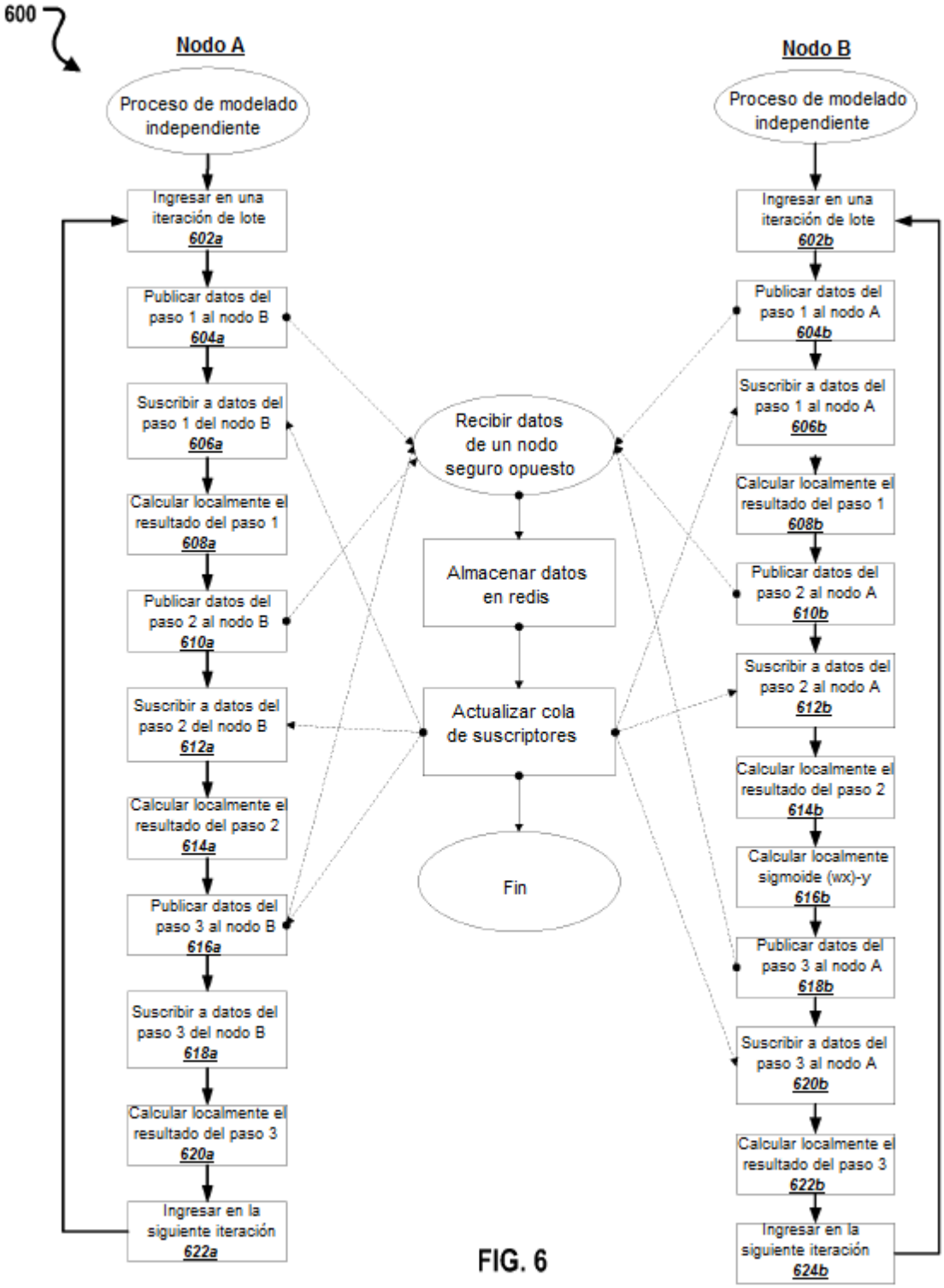
600
Nodo A
Proceso de modelado independiente
Ingresar en una iteración de lote 602a
Publicar datos del paso 1 al nodo B 604a
Suscribir a datos del paso 1 del nodo B 606a
Calcular localmente el resultado del paso 1 608a
Publicar datos del paso 2 al nodo B 610a
Suscribir a datos del paso 2 del nodo B 612a
Calcular localmente el resultado del paso 2 614a
Publicar datos del paso 3 al nodo B 616a
Suscribir a datos del paso 3 del nodo B 618a
Calcular localmente el resultado del paso 3 620a
Ingresar en la siguiente iteración 622a
Recibir datos de un nodo seguro opuesto
Almacenar datos en redis
Actualizar cola de suscriptores
Fin
Nodo B
Proceso de modelado independiente
Ingresar en una iteración de lote 602b
Publicar datos del paso 1 al nodo A 604b
Suscribir a datos del paso 1 al nodo A 606b
Calcular localmente el resultado del paso 1 608b
Publicar datos del paso 2 al nodo A 610b
Suscribir a datos del paso 2 al nodo A 612b
Calcular localmente el resultado del paso 2 614b
Calcular localmente sigmoide (wx)-y 616b
Publicar datos del paso 3 al nodo A 618b
Suscribir a datos del paso 3 al nodo A 620b
Calcular localmente el resultado del paso 3 622b
Ingresar en la siguiente iteración 624b

FIG. 6

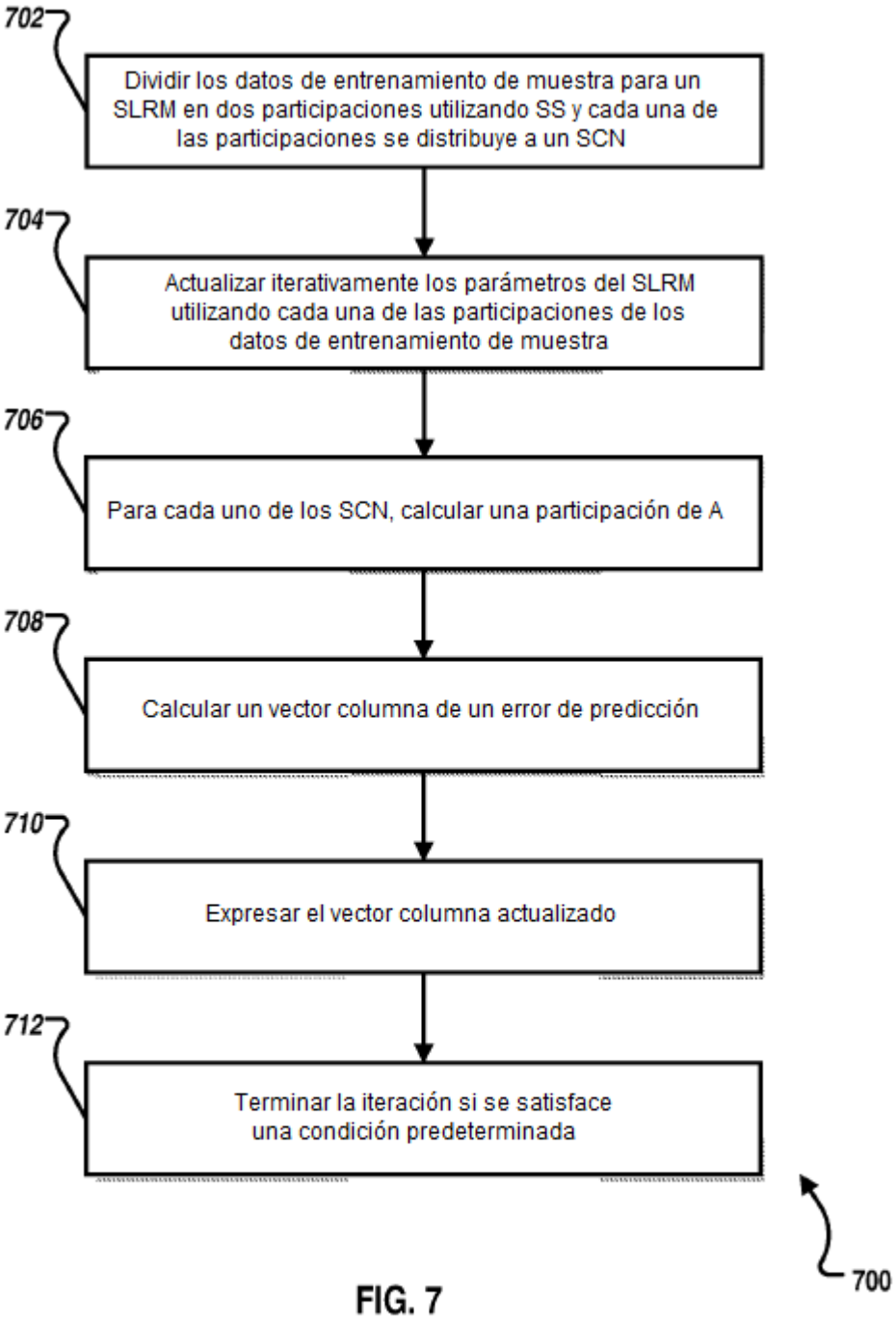
702
Dividir los datos de entrenamiento de muestra para un SLRM en dos participaciones utilizando SS y cada una de las participaciones se distribuye a un SCN
704
Actualizar iterativamente los parámetros del SLRM utilizando cada una de las participaciones de los datos de entrenamiento de muestra
706
Para cada uno de los SCN, calcular una participación de A
708
Calcular un vector columna de un error de predicción
710
Expresar el vector columna actualizado
712
Terminar la iteración si se satisface una condición predeterminada
700

FIG. 7

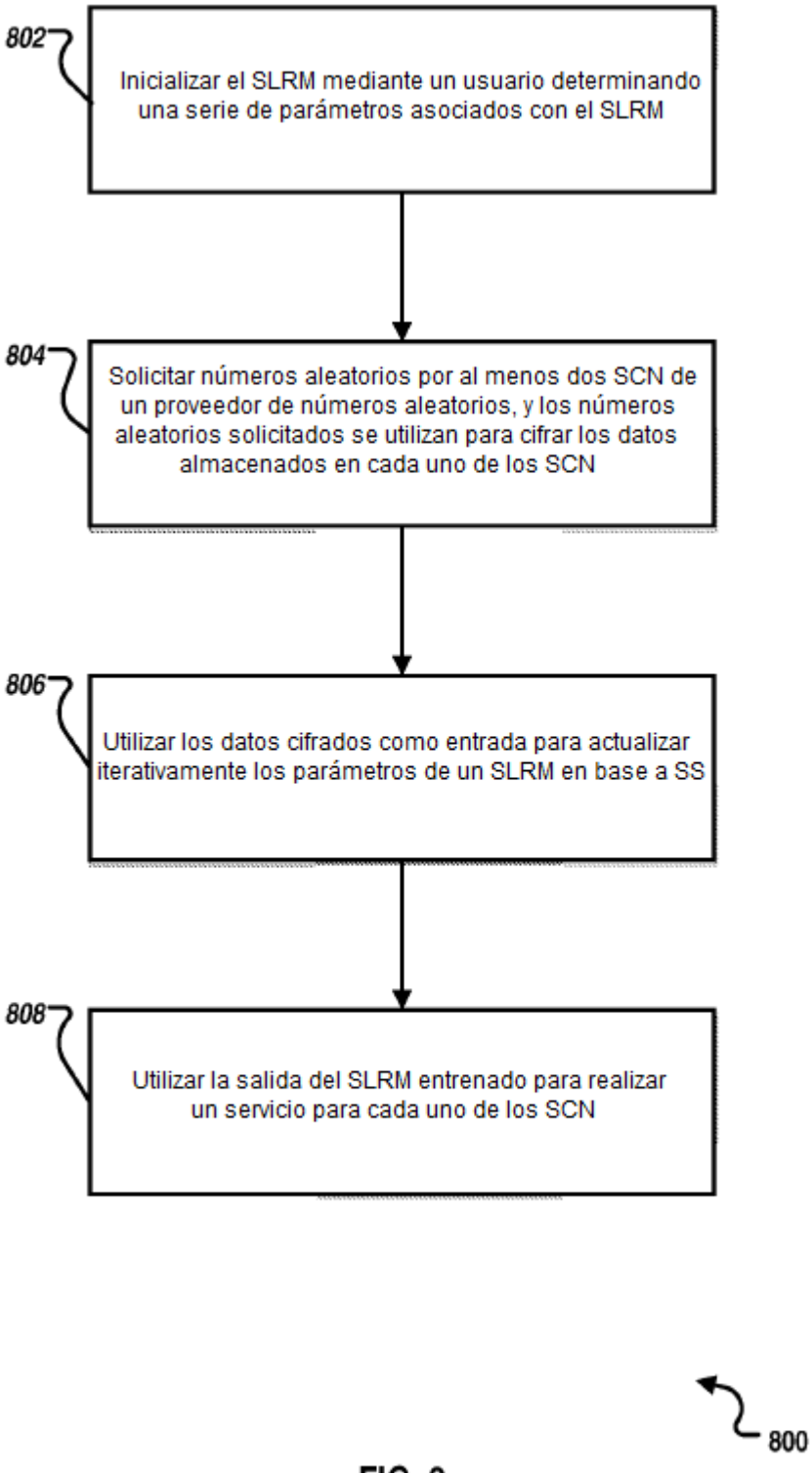
802
Inicializar el SLRM mediante un usuario determinando una serie de parámetros asociados con el SLRM
804
Solicitar números aleatorios por al menos dos SCN de un proveedor de números aleatorios, y los números aleatorios solicitados se utilizan para cifrar los datos almacenados en cada uno de los SCN
806
Utilizar los datos cifrados como entrada para actualizar iterativamente los parámetros de un SLRM en base a SS
808
Utilizar la salida del SLRM entrenado para realizar un servicio para cada uno de los SCN
800

FIG. 8

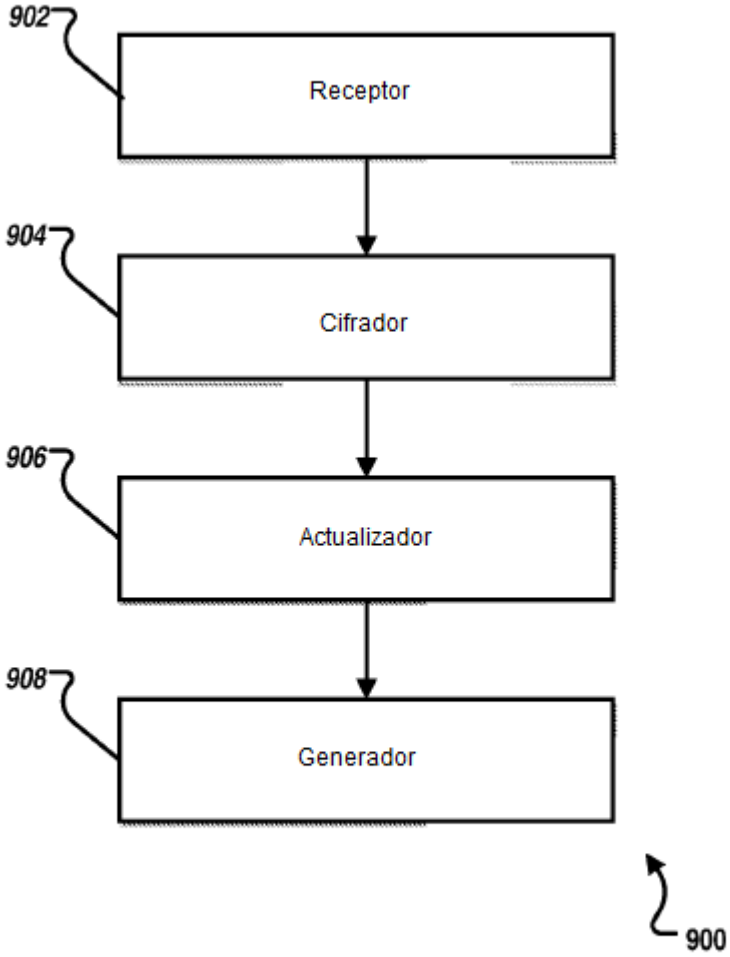
902
Receptor
904
Cifrador
906
Actualizador
908
Generador
900

FIG. 9