(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
02 January 2025 (02.01.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/006729 A2

(51) International Patent Classification:
Not classified

(21) International Application Number:
PCT/US2024/035789

(22) International Filing Date:
27 June 2024 (27.06.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/523,956 29 June 2023 (29.06.2023) US

(71) Applicant: **VISA INTERNATIONAL SERVICE ASSOCIATION** [US/US]; P.O. Box 8999, San Francisco, California 94128 (US).

(72) Inventors: **HE, Runxin**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **LOU, Mingji**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **KERSTING, Nicholas, Stephen**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **LI, Songshan**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **HO, Iat, Kei**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **OKOCHU, Raphael**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US). **GU, Yu**; Visa International Service Association, P.O. Box 8999, San Francisco, California 94128 (US).

(74) Agent: **PREPELKA, Nathan, J.** et al.; The Webb Law Firm, One Gateway Center, 420 Ft. Duquesne Blvd., Suite 1200, Pittsburgh, Pennsylvania 15222 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG,

(54) Title: SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR INCREMENTAL LEARNING

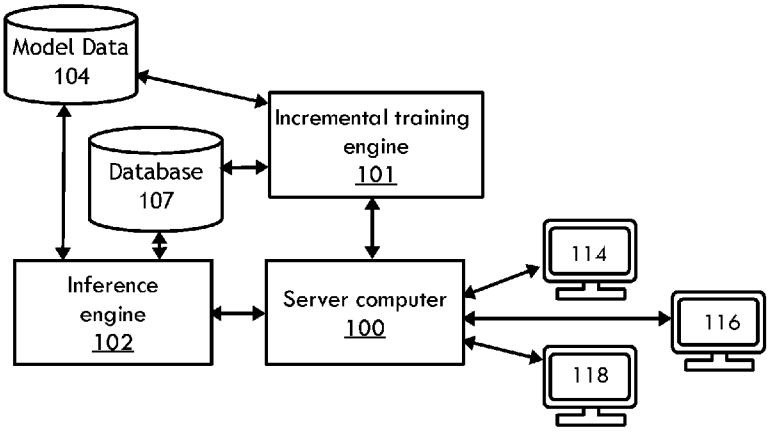

FIG. 1

1000

(57) Abstract: Systems, methods, and computer program products for incremental learning are provided. A system includes at least one processor programmed or configured to execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, determine to train the first machine-learning model based on at least one rule, in response to determining to train the first machine-learning model, creating a second machine-learning model including weights from the first machine-learning model, train the second-machine learning model with the model data stored in the at least one data storage device, determine whether to replace the first machine-learning model with the second machine-learning model, in response to determining to replace the first machine-learning model with the second machine-learning model, and replace the first machine-learning model with the second machine-learning model in the production environment.

WO 2025/006729 A2

[Continued on next page]

KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR INCREMENTAL LEARNING

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application claims the benefit of U.S. Provisional Patent Application No. 63/523,956, filed on June 29, 2023, the disclosure of which is hereby incorporated by reference in its entirety.

BACKGROUND

1. Field

[0002] This disclosure relates generally to machine-learning and, in some non-limiting embodiments or aspects, to systems, methods, and computer program products for incremental learning for a model in a production environment.

2. Technical Considerations

[0003] Inference platforms to monitor a machine-learning model in production have several technical limitations. For example, training is a separate process and retraining the model may take a considerable amount of resources (e.g., memory, processing, time, etc.). The production and training environments both are associated with workloads and overheads. Further, during training of such models there are limited opportunities for production tests. Moreover, the training process is often slow in response to data distribution during the in-production generation of inferences.

SUMMARY

[0004] According to non-limiting embodiments or aspects, provided is a system comprising: at least one data storage device; and at least one processor programmed or configured to: execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; store, in the at least one data storage device, model data for each execution of the first machine-learning model; determine to train the first machine-learning model based on at least one rule; in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model; train the second-machine learning model with the model data stored in the at least one data storage device; determine whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in

the production environment such that a next plurality of requests are input to the second machine-learning model. In non-limiting embodiments or aspects, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

[0005] In non-limiting embodiments or aspects, the plurality of requests are processed by a batch processor, and the at least one processor is further programmed or configured to generate a dashboard interface configured to communicate with the batch processor. In non-limiting embodiments or aspects, determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model. In non-limiting embodiments or aspects, determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy. In non-limiting embodiments or aspects, the at least one rule is based on at least one of model score and feature distribution. In non-limiting embodiments or aspects, the inference comprises a score.

[0006] According to non-limiting embodiments or aspects, provided is a computer-implemented method comprising: executing, with at least one processor, a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; storing, in at least one data storage device, model data for each execution of the first machine-learning model; determining, with at least one processor, to train the first machine-learning model based on at least one rule; in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model; training, with at least one processor, the second-machine learning model with the model data stored in the at least one data storage device; determining, with at least one processor, whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replacing the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

[0007] In non-limiting embodiments or aspects, the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests. In non-limiting

embodiments or aspects, the plurality of requests are processed by a batch processor, the method includes generating a dashboard interface configured to communicate with the batch processor. In non-limiting embodiments or aspects, determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model. In non-limiting embodiments or aspects, determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy. In non-limiting embodiments or aspects, the at least one rule is based on at least one of model score and feature distribution. In non-limiting embodiments or aspects, the inference comprises a score.

[0008] According to non-limiting embodiments or aspects, provided is a computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; store, in at least one data storage device, model data for each execution of the first machine-learning model; determine to train the first machine-learning model based on at least one rule; in response to determining to train the first machine-learning model, create a second machine-learning model comprising weights from the first machine-learning model; train the second-machine learning model with the model data stored in the at least one data storage device; determine whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

[0009] Other non-limiting embodiments or aspects will be set forth in the following numbered clauses:

[0010] Clause 1: A system comprising: at least one data storage device; and at least one processor programmed or configured to: execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; store, in the at least one data storage device, model data for

each execution of the first machine-learning model; determine to train the first machine-learning model based on at least one rule; in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model; train the second-machine learning model with the model data stored in the at least one data storage device; determine whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

[0011] Clause 2: The system of clause 1, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

[0012] Clause 3: The system of clause 1 or 2, wherein the plurality of requests are processed by a batch processor, and wherein the at least one processor is further programmed or configured to generate a dashboard interface configured to communicate with the batch processor.

[0013] Clause 4: The system of any of clauses 1-3, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

[0014] Clause 5: The system of any of clauses 1-4, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

[0015] Clause 6: The system of any of clauses 1-5, wherein the at least one rule is based on at least one of model score and feature distribution.

[0016] Clause 7: The system of any of clauses 1-6, wherein the inference comprises a score.

[0017] Clause 8: A computer-implemented method comprising: executing, with at least one processor, a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; storing, in at least one data storage device, model data for each execution of the first machine-learning model; determining, with at least one processor, to train the first machine-

learning model based on at least one rule; in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model; training, with at least one processor, the second-machine learning model with the model data stored in the at least one data storage device; determining, with at least one processor, whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replacing the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

[0018] Clause 9: The method of clause 8, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

[0019] Clause 10: The method of clause 8 or 9, wherein the plurality of requests are processed by a batch processor, further comprising generating a dashboard interface configured to communicate with the batch processor.

[0020] Clause 11: The method of any of clauses 8-10, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

[0021] Clause 12: The method of any of clauses 8-11, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

[0022] Clause 13: The method of any of clauses 8-12, wherein the at least one rule is based on at least one of model score and feature distribution.

[0023] Clause 14: The method of any of clauses 8-13, wherein the inference comprises a score.

[0024] Clause 15: A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first-machine learning model configured to output an inference for each input; store, in at least one data storage device, model data for each execution of the first machine-learning model; determine

to train the first machine-learning model based on at least one rule; in response to determining to train the first machine-learning model, create a second machine-learning model comprising weights from the first machine-learning model; train the second-machine learning model with the model data stored in the at least one data storage device; determine whether to replace the first machine-learning model with the second machine-learning model; and in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

[0025] Clause 16: The computer program product of clause 15, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

[0026] Clause 17: The computer program product of clause 15 or 16, wherein the plurality of requests are processed by a batch processor, and wherein the at least one processor is further programmed or configured to generate a dashboard interface configured to communicate with the batch processor.

[0027] Clause 18: The computer program product of any of clauses 15-17, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

[0028] Clause 19: The computer program product of any of clauses 15-18, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

[0029] Clause 20: The computer program product of any of clauses 15-19, wherein the at least one rule is based on at least one of model score and feature distribution.

[0030] These and other features and characteristics of the present disclosure, as well as the methods of operation and functions of the related elements of structures and the combination of parts and economies of manufacture, will become more apparent upon consideration of the following description and the appended claims with reference to the accompanying drawings, all of which form a part of this specification, wherein like reference numerals designate corresponding parts in the various figures. It is to be expressly understood, however, that the drawings are for the purpose of

illustration and description only and are not intended as a definition of the limits of the disclosed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0031] Additional advantages and details are explained in greater detail below with reference to the non-limiting, exemplary embodiments that are illustrated in the accompanying schematic figures, in which:

[0032] FIG. 1 is a schematic diagram of a system for incremental learning according to some non-limiting embodiments or aspects;

[0033] FIG. 2 is a schematic diagram of a system for incremental learning according to some non-limiting embodiments or aspects;

[0034] FIG. 3 is a flow diagram of a method for incremental learning according to some non-limiting embodiments or aspects; and

[0035] FIG. 4 is a schematic diagram of example components of one or more devices according to some non-limiting embodiments or aspects.

DETAILED DESCRIPTION

[0036] For purposes of the description hereinafter, the terms “end,” “upper,” “lower,” “right,” “left,” “vertical,” “horizontal,” “top,” “bottom,” “lateral,” “longitudinal,” and derivatives thereof shall relate to the embodiments as they are oriented in the drawing figures. However, it is to be understood that the embodiments may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings, and described in the following specification, are simply exemplary embodiments or aspects of the disclosed subject matter. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0037] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly described as such. Also, as used herein, the articles “a” and “an” are intended to include one or more items and may be used interchangeably with “one or more” and “at least one.” Furthermore, as used herein, the term “set” is intended to include one or more items (e.g., related items, unrelated items, a combination of related and unrelated items, and/or the like) and may be used interchangeably with “one or more” or “at least one.” Where only one item is intended, the term “one” or similar language is used. Also, as used herein, the terms “has,” “have,” “having,” or the like are intended

to be open-ended terms. Further, the phrase “based on” is intended to mean “based at least partially on” unless explicitly stated otherwise.

[0038] As used herein, the term “account identifier” may include one or more primary account numbers (PANs), tokens, or other identifiers associated with a customer account. The term “token” may refer to an identifier that is used as a substitute or replacement identifier for an original account identifier, such as a PAN. Account identifiers may be alphanumeric or any combination of characters and/or symbols. Tokens may be associated with a PAN or other original account identifier in one or more data structures (e.g., one or more databases, and/or the like) such that they may be used to conduct a transaction without directly using the original account identifier. In some examples, an original account identifier, such as a PAN, may be associated with a plurality of tokens for different individuals or purposes.

[0039] An “application program interface” (API) refers to computer code or other data sorted on a computer-readable medium that may be executed by a processor to facilitate the interaction between software components, such as a client-side front-end and/or server-side back-end for receiving data from the client. An “interface” refers to a generated display, such as one or more graphical user interfaces (GUIs) with which a user may interact, either directly or indirectly (e.g., through a keyboard, mouse, etc.).

[0040] As used herein, the term “communication” may refer to the reception, receipt, transmission, transfer, provision, and/or the like of data (e.g., information, signals, messages, instructions, commands, and/or the like). For one unit (e.g., a device, a system, a component of a device or system, combinations thereof, and/or the like) to be in communication with another unit means that the one unit is able to directly or indirectly receive information from and/or transmit information to the other unit. This may refer to a direct or indirect connection (e.g., a direct communication connection, an indirect communication connection, and/or the like) that is wired and/or wireless in nature. Additionally, two units may be in communication with each other even though the information transmitted may be modified, processed, relayed, and/or routed between the first and second unit. For example, a first unit may be in communication with a second unit even though the first unit passively receives information and does not actively transmit information to the second unit. As another example, a first unit may be in communication with a second unit if at least one intermediary unit processes information received from the first unit and communicates the processed information to the second unit.

[0041] As used herein, the term "computing device" may refer to one or more electronic devices configured to process data. A computing device may, in some examples, include the necessary components to receive, process, and output data, such as a processor, a display, a memory, an input device, a network interface, and/or the like. A computing device may be a mobile device. As an example, a mobile device may include a cellular phone (e.g., a smartphone or standard cellular phone), a portable computer, a wearable device (e.g., watches, glasses, lenses, clothing, and/or the like), a personal digital assistant (PDA), and/or other like devices. A computing device may also be a desktop computer or other form of non-mobile computer.

[0042] As used herein, the term "issuer institution" may refer to one or more entities, such as a bank, that provide accounts to customers for conducting transactions (e.g., payment transactions), such as initiating credit and/or debit payments. For example, an issuer institution may provide an account identifier, such as a PAN, to a customer that uniquely identifies one or more accounts associated with that customer. The account identifier may be embodied on a portable financial device, such as a physical financial instrument, e.g., a payment card, and/or may be electronic and used for electronic payments. The term "issuer system" refers to one or more computer devices operated by or on behalf of an issuer institution, such as a server computer executing one or more software applications. For example, an issuer system may include one or more authorization servers for authorizing a transaction.

[0043] As used herein, the term "merchant" may refer to an individual or entity that provides goods and/or services, or access to goods and/or services, to customers based on a transaction, such as a payment transaction. The term "merchant" or "merchant system" may also refer to one or more computer systems operated by or on behalf of a merchant, such as a server computer executing one or more software applications.

[0044] As used herein, the terms "client" and "client device" may refer to one or more client-side devices or systems (e.g., remote from a transaction service provider) used to initiate or facilitate a transaction (e.g., a payment transaction). As an example, a "client device" may refer to one or more POS devices used by a merchant, one or more acquirer host computers used by an acquirer, one or more mobile devices used by a user, and/or the like. In some non-limiting embodiments or aspects, a client device may be an electronic device configured to communicate with one or more networks and initiate or facilitate transactions. For example, a client device may include one or

more computers, portable computers, laptop computers, tablet computers, mobile devices, cellular phones, wearable devices (e.g., watches, glasses, lenses, clothing, and/or the like), PDAs, and/or the like. Moreover, a "client" may also refer to an entity (e.g., a merchant, an acquirer, and/or the like) that owns, utilizes, and/or operates a client device for initiating transactions (e.g., for initiating transactions with a transaction service provider).

[0045] As used herein, the term "payment device" may refer to a payment card (e.g., a credit or debit card), a gift card, a smartcard, smart media, a payroll card, a healthcare card, a wristband, a machine-readable medium containing account information, a keychain device or fob, an RFID transponder, a retailer discount or loyalty card, a cellular phone, an electronic wallet mobile application, a personal digital assistant (PDA), a pager, a security card, a computing device, an access card, a wireless terminal, a transponder, and/or the like. In some non-limiting embodiments or aspects, the payment device may include volatile or non-volatile memory to store information (e.g., an account identifier, a name of the account holder, and/or the like).

[0046] As used herein, the term "payment gateway" may refer to an entity and/or a payment processing system operated by or on behalf of such an entity (e.g., a merchant service provider, a payment service provider, a payment facilitator, a payment facilitator that contracts with an acquirer, a payment aggregator, and/or the like), which provides payment services (e.g., transaction service provider payment services, payment processing services, and/or the like) to one or more merchants. The payment services may be associated with the use of payment devices managed by a transaction service provider. As used herein, the term "payment gateway system" may refer to one or more computer systems, computer devices, servers, groups of servers, and/or the like, operated by or on behalf of a payment gateway.

[0047] As used herein, the term "server" may refer to or include one or more computing devices that are operated by or facilitate communication and processing for multiple parties in a network environment, such as the internet, although it will be appreciated that communication may be facilitated over one or more public or private network environments and that various other arrangements are possible. Further, multiple computing devices (e.g., servers, point-of-sale (POS) devices, mobile devices, etc.) directly or indirectly communicating in the network environment may constitute a "system." Reference to "a server" or "a processor," as used herein, may refer to a previously-recited server and/or processor that is recited as performing a

previous step or function, a different server and/or processor, and/or a combination of servers and/or processors. For example, as used in the specification and the claims, a first server and/or a first processor that is recited as performing a first step or function may refer to the same or different server and/or a processor recited as performing a second step or function.

[0048] As used herein, the term "transaction service provider" may refer to an entity that receives transaction authorization requests from merchants or other entities and provides guarantees of payment, in some cases through an agreement between the transaction service provider and an issuer institution. For example, a transaction service provider may include a payment network such as Visa® or any other entity that processes transactions. The term "transaction processing system" may refer to one or more computer systems operated by or on behalf of a transaction service provider, such as a transaction processing server executing one or more software applications. A transaction processing server may include one or more processors and, in some non-limiting embodiments or aspects, may be operated by or on behalf of a transaction service provider.

[0049] Non-limiting embodiments or aspects of the disclosed subject matter are directed to systems, methods, and computer program products for incremental learning. Non-limiting embodiments allow for an inference model to be trained while it is being used to execute real-time and/or batched processing requests. For example, non-limiting embodiments allow for a model to be trained while deployed in a production environment, using smaller subsets of data each time as opposed to training at a longer interval with a larger amount of data. Non-limiting embodiments provide for the model improvement in a production environment, thus avoiding disruption of service, delay, and other undesirable effects of training a model. Non-limiting embodiments may be used in conjunction with a batch inference platform as described in PCT Application No. PCT/US2024/35341, filed June 25, 2024, which is incorporated herein by reference in its entirety.

[0050] Non-limiting embodiments may be implemented within an electronic payment processing network, although it will be appreciated that non-limiting embodiments may be implemented within various different types of environments, not limited to payment networks, where one or more models are executed as a service and benefit from regular and/or periodic training.

[0051] Referring to FIG. 1, a system 1000 for incremental learning is shown according to some non-limiting embodiments or aspects. A server computer 100 may include one or more computing devices, such as a web server, private server, authentication server (e.g., such as a 3D Secure system), and/or any other computing device configured to communicate with one or more client devices (e.g., 114, 116, 118). The client devices 114, 116, 118 may include computing devices that make requests from the server computer 100, such as processing requests (e.g., transaction processing and/or the like). For example, client devices 114, 116, 118 may each include issuer systems, and server computer 100 may be associated with a transaction processing system and/or payment gateway. It will be appreciated, however, that various entities may control and/or operate the devices and systems shown in FIG. 1 in various non-limiting embodiments. The client devices 114, 116, 118 may communicate with the server computer 100 through one or more APIs, as an example, although various communication methods may be used.

[0052] With continued reference to FIG. 1, the server computer 100 may be in communication with an inference engine 102. The inference engine 102 may include one or more computing devices and/or software applications executed by one or more computing devices. In some examples, the inference engine 102 may be part of the server computer 100. In some examples the inference engine 102 may be separate and/or remote from the server computer 100. The inference engine 102 may be configured to control execution of one or more machine-learning models stored as model data 104 in one or more data storage devices. The inference engine 102 may also be in communication with a database 107, which may include an audit log for each execution of a model, model metric data, and/or the like. In some examples in which client devices 114, 116, 118 are issuer systems, the machine-learning models may include scoring models (e.g., fraud or risk scoring models) used by issuer systems to make decisions (e.g., authorization decisions).

[0053] An incremental training engine 101 is also in communication with the server computer 100. The incremental training engine 101 may include one or more computing devices and/or software applications executed by one or more computing devices. In operation, the inference engine 102 may execute a first machine-learning model (e.g., a production model) for inputs received in real-time or in batch from the server computer 100 based on requests from client devices 114, 116, 118. The inference and model data resulting from the model execution may be stored in the

database 107. The server computer 100 determines when to train the first machine-learning model. For example, training may be performed based on a predetermined time interval, a number of transactions processed, user input, and/or the like. In non-limiting embodiments, determining to train the model is based on one or more rules, which may include thresholds for model scores, feature distribution, and/or the like.

[0054] As used herein, the term "real-time" may refer to performance of a task or tasks during another process or before another process is completed. For example, the inference engine 102 may execute the first machine-learning model in real-time relative to processing a transaction (e.g., before or during authorization of a transaction or the like), during processing and/or communication of messages related to an event, at the time of making a decision (e.g., authorization decision, authentication decision and/or the like) related to an event (e.g., receiving an authorization request, at least a portion of which is included in the data sample, and determining an authorization decision based thereon), and/or the like.

[0055] Non-limiting embodiments provide for a faster, more computational efficient training process by adding a smaller (e.g., incremental) subset of data as compared to non-incremental approaches, and training on top of the previous (e.g., current) model weights. By using smaller subsets of data and incrementally training, an accurate model may be maintained in a live production environment.

[0056] With continued reference to FIG. 1, when the server computer 100 determines to train the first machine-learning model while the model is in production, it may communicate with the incremental training engine 101. The incremental training engine 101 may be separate from and/or a part of the server computer 100. The incremental training engine 101 may then create a second machine-learning model based on the first machine-learning model that is in production. The second machine-learning model may be stored as model data 104 and/or be stored in temporary memory. The second machine-learning model may include weights from the first machine-learning model. The second machine-learning model may be duplicated from the first machine-learning model. The incremental training engine 101 may then train the second machine-learning model based on the audit log stored in the database 107 from prior executions within a time period. In this manner, the second machine-learning model may be trained while the first machine-learning model is being executed in the production environment.

[0057] Still referring to FIG. 1, in response to the training of the second machine-learning model, the incremental training engine 101 and/or server computer 100 may then determine whether to replace the first machine-learning model with the second machine-learning model. In response to determining to replace the first machine-learning model with the second machine-learning model, the incremental training engine 101 and/or server computer 100 may then replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model. For example, the first machine-learning model in the model data 104 may be replaced, removed, and/or the like such that the second machine-learning model is used moving forward.

[0058] In non-limiting embodiments, determining whether to replace the first machine-learning model with the second machine-learning model involves comparing the two models (e.g., comparing efficiency, accuracy, and/or the like). Replacing the first model may occur if differences in model metrics between the newly trained model and the production model satisfy a predetermined threshold (e.g., meet or exceed a threshold). It will be appreciated that other parameters may be considered to determine when to replace the existing model with a newly trained model.

[0059] Referring to FIG. 2, a system 2000 for incremental learning is shown according to some non-limiting embodiments or aspects. An inference engine 202 may include one or more computing devices and/or software applications executed by one or more computing devices for executing one or more machine-learning models in response to a request. The inference engine 202 may be a real-time inference engine and/or a batch inference engine. For example, a payload may include input data for a risk scoring model. In non-limiting embodiments, the model inference engine 202 may execute one or more models upon request. In non-limiting embodiments, in response to receiving multiple requests to provide reasoning associated with an output (e.g., a request for explainable machine-learning or artificial intelligence metrics), the model inference engine 202 may batch process the requests. An audit log 201 may store model metadata resulting from execution of the machine-learning model(s) such that the model metadata is available to the inference engine 202.

[0060] With continued reference to FIG. 2, the inference engine 202 may output explainable machine-learning or artificial intelligence metrics relating to the model and reasons (e.g., influences) on the model output. A monitoring system dashboard 206

may present one or more GUIs to users to show graphical representations of data in the audit log 201 in addition to explainable metrics. For example, the monitoring system dashboard 206 may display alerts, notifications, and/or the like based on real-time and/or batched executions of the machine-learning model(s). In non-limiting embodiments, the monitoring system dashboard 206 is in communication with the inference engine 202 and audit log 201. The dashboard 206 may display some or all of the model metric data and/or facilitate user interaction with the inference engine 202 and/or audit log 201. For example, a user may access the dashboard 206 through a web browser or application and use it to configure a training interval time, view model metric data, and/or the like. In response to determining to train one or more models used by the inference engine, the inference engine and/or dashboard 206 may communicate with a feature generation engine 209. The feature generation engine 209 may include one or more computing devices and/or software applications executed by one or more computing devices for generating features (e.g., feature vectors) from the model data stored in the audit log 201 from prior executions.

[0061] With continued reference to FIG. 2, an incremental training engine 208 may be configured to train a second model that is based on one or more of the models in production by the inference engine 202. The incremental training engine 208 outputs a challenger model 205, which is analyzed by a model evaluation engine 207 to compare it with the model currently in production. The model evaluation engine 207 may include one or more computing devices and/or software applications executed by one or more computing devices for performing such a comparison. The production model(s) executed by the inference engine 202 may be replaced with the challenger model 205 in response to the model evaluation engine determining to replace it based on a comparison of model metrics.

[0062] In non-limiting embodiments, the machine-learning models discussed herein may be unsupervised models. However, it will be appreciated that in non-limiting embodiments, supervised learning models may also be used, for example in instances in which the model data (e.g., labels generated with the model, observations, and/or the like) is updated.

[0063] Non-limiting embodiments provide for incremental learning that avoids a need to retrain a model from a beginning state (e.g., "from scratch") each time it is trained. By iterating through multiple versions of models over time as the model data

grows and provides more training data, incremental training is provided for each stage and/or threshold.

[0064] Referring now to FIG. 3, shown is a flow diagram for a method for incremental learning according to some non-limiting embodiments or aspects. The steps shown in FIG. 3 are for example purposes only. It will be appreciated that additional, fewer, different, and/or a different order of steps may be used in some non-limiting embodiments or aspects. In some non-limiting embodiments or aspects, a step may be automatically performed in response to performance and/or completion of a prior step. At step 300, a first machine-learning model may be executed in a production environment. For example, the first machine-learning model may be configured in an electronic payment network to provide real-time inferences that are used by the payment network, such as but not limited to fraud determinations used for authorization and/or the like.

[0065] At step 302, model data from the inference processed at step 300 may be stored in a data structure, such as a model database, audit log, and/or the like. The model data may include a model input, a model output, model parameters (e.g., weights of nodes and/or edges of a network, transformations, and/or the like), and/or model metrics (e.g., Shapley values or other metrics representing how one or more features affect an individual prediction and/or how much that feature affected that prediction score compared to other features).

[0066] At step 304, it is determined whether to train the machine-learning model based on the model data stored at step 302. Such a determination may be based on one or more rules, such as one or more thresholds associated with model performance metrics. For example, the score and/or feature distribution may be compared with one or more thresholds to determine whether training and/or retraining should be triggered at step 304. In some examples, feature distribution drift, score distribution drift, and/or the like may be used to trigger training and/or retraining at step 304. In some non-limiting embodiments, Shapley values or other metrics representing how one or more features affect an individual prediction (e.g. a score) and/or how much that feature affected (e.g., increased and/or decreased) that prediction score compared to other features may be used to monitor the model and trigger and/or retrigger retraining. Inputs continue to be provided to the first machine-learning model that is live in the production environment.

[0067] Once training and/or retraining is triggered, the method may proceed to step 306 and a second model (e.g., a replica model) may be created (e.g., generated) based on the first model. For example, a copy of the model with the same weights and other parameters may be generated so that it can be trained while the first model is still being executed in the production environment. At step 308, the second model is trained based on the model data stored at step 302. Once the second model is trained using the model data, at step 310 it may be determined whether to replace the first model that is currently live in the production environment with the second model. The trained second model becomes a challenger model that is compared to the first model. For example, at step 310 the models may be evaluated by computation efficiency and/or accuracy based on a cross-validation data set. Other comparison methods may be performed. In response to determining to replace the model, the method proceeds to step 312 and the first model is replaced with the second model as the live model in the production environment. If it is determined not to replace the model, for example if the second model does not perform as well as the first model, the method may proceed back to step 300 and continue with the first model still in the production environment.

[0068] Referring now to FIG. 4, shown is a diagram of example components of a device 400 according to non-limiting embodiments or aspects. Device 400 may correspond to at least one of the computing devices (e.g., server computer 100, inference engine 102, incremental training engine 101, and/or the like) in FIG. 1. In some non-limiting embodiments or aspects, such systems or devices may include at least one device 400 and/or at least one component of device 400. The number and arrangement of components shown in FIG. 4 are provided as an example. In some non-limiting embodiments or aspects, device 400 may include additional components, fewer components, different components, or differently arranged components than those shown in FIG. 4. Additionally, or alternatively, a set of components (e.g., one or more components) of device 400 may perform one or more functions described as being performed by another set of components of device 400.

[0069] As shown in FIG. 4, device 400 may include bus 402, processor 404, memory 406, storage component 408, input component 410, output component 412, and communication interface 414. Bus 402 may include a component that permits communication among the components of device 400. In some non-limiting embodiments or aspects, processor 404 may be implemented in hardware, firmware, or a combination of hardware and software. For example, processor 404 may include

a processor (e.g., a central processing unit (CPU), a GPU, an accelerated processing unit (APU), etc.), a microprocessor, a digital signal processor (DSP), and/or any processing component (e.g., a field-programmable gate array (FPGA), an application-specific integrated circuit (ASIC), etc.) that can be programmed to perform a function. Memory 406 may include random access memory (RAM), read only memory (ROM), and/or another type of dynamic or static storage device (e.g., flash memory, magnetic memory, optical memory, etc.) that stores information and/or instructions for use by processor 404.

[0070] With continued reference to FIG. 4, storage component 408 may store information and/or software related to the operation and use of device 400. For example, storage component 408 may include a hard disk (e.g., a magnetic disk, an optical disk, a magneto-optic disk, a solid state disk, etc.) and/or another type of computer-readable medium. Input component 410 may include a component that permits device 400 to receive information, such as via user input (e.g., a touch screen display, a keyboard, a keypad, a mouse, a button, a switch, a microphone, etc.). Additionally, or alternatively, input component 410 may include a sensor for sensing information (e.g., a global positioning system (GPS) component, an accelerometer, a gyroscope, an actuator, etc.). Output component 412 may include a component that provides output information from device 400 (e.g., a display, a speaker, one or more light-emitting diodes (LEDs), etc.). Communication interface 414 may include a transceiver-like component (e.g., a transceiver, a separate receiver and transmitter, etc.) that enables device 400 to communicate with other devices, such as via a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication interface 414 may permit device 400 to receive information from another device and/or provide information to another device. For example, communication interface 414 may include an Ethernet interface, an optical interface, a coaxial interface, an infrared interface, a radio frequency (RF) interface, a universal serial bus (USB) interface, a Wi-Fi® interface, a cellular network interface, and/or the like.

[0071] Device 400 may perform one or more processes described herein. Device 400 may perform these processes based on processor 404 executing software instructions stored by a computer-readable medium, such as memory 406 and/or storage component 408. A computer-readable medium may include any non-transitory memory device. A memory device includes memory space located inside of a single

physical storage device or memory space spread across multiple physical storage devices. Software instructions may be read into memory 406 and/or storage component 408 from another computer-readable medium or from another device via communication interface 414. When executed, software instructions stored in memory 406 and/or storage component 408 may cause processor 404 to perform one or more processes described herein. Additionally, or alternatively, hardwired circuitry may be used in place of or in combination with software instructions to perform one or more processes described herein. Thus, embodiments described herein are not limited to any specific combination of hardware circuitry and software. The term "configured to," as used herein, may refer to an arrangement of software, device(s), and/or hardware for performing and/or enabling one or more functions (e.g., actions, processes, steps of a process, and/or the like). For example, "a processor configured to" may refer to a processor that executes software instructions (e.g., program code) that cause the processor to perform one or more functions.

[0072] Although embodiments have been described in detail for the purpose of illustration, it is to be understood that such detail is solely for that purpose and that the disclosure is not limited to the disclosed embodiments or aspects, but, on the contrary, is intended to cover modifications and equivalent arrangements that are within the spirit and scope of the appended claims. For example, it is to be understood that the present disclosure contemplates that, to the extent possible, one or more features of any embodiment or aspect can be combined with one or more features of any other embodiment or aspect.

WHAT IS CLAIMED IS:

1. A system comprising:

at least one data storage device; and

at least one processor programmed or configured to:

execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first machine-learning model configured to output an inference for each input;

store, in the at least one data storage device, model data for each execution of the first machine-learning model;

determine to train the first machine-learning model based on at least one rule;

in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model;

train the second machine-learning model with the model data stored in the at least one data storage device;

determine whether to replace the first machine-learning model with the second machine-learning model; and

in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

2. The system of claim 1, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

3. The system of claim 2, wherein the plurality of requests are processed by a batch processor, and wherein the at least one processor is further programmed or configured to generate a dashboard interface configured to communicate with the batch processor.

4. The system of claim 1, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

5. The system of claim 4, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

6. The system of claim 1, wherein the at least one rule is based on at least one of model score and feature distribution.

7. The system of claim 1, wherein the inference comprises a score.

8. A computer-implemented method comprising:

executing, with at least one processor, a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first machine-learning model configured to output an inference for each input;

storing, in at least one data storage device, model data for each execution of the first machine-learning model;

determining, with at least one processor, to train the first machine-learning model based on at least one rule;

in response to determining to train the first machine-learning model, creating a second machine-learning model comprising weights from the first machine-learning model;

training, with at least one processor, the second machine-learning model with the model data stored in the at least one data storage device;

determining, with at least one processor, whether to replace the first machine-learning model with the second machine-learning model; and

in response to determining to replace the first machine-learning model with the second machine-learning model, replacing the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

9. The method of claim 8, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

10. The method of claim 9, wherein the plurality of requests are processed by a batch processor, further comprising generating a dashboard interface configured to communicate with the batch processor.

11. The method of claim 8, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

12. The method of claim 11, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

13. The method of claim 8, wherein the at least one rule is based on at least one of model score and feature distribution.

14. The method of claim 8, wherein the inference comprises a score.

15. A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to:

execute a first machine-learning model for each input of a plurality of inputs associated with a plurality of requests in a production environment, the first machine-learning model configured to output an inference for each input;

store, in at least one data storage device, model data for each execution of the first machine-learning model;

determine to train the first machine-learning model based on at least one rule;

in response to determining to train the first machine-learning model, create a second machine-learning model comprising weights from the first machine-learning model;

train the second machine-learning model with the model data stored in the at least one data storage device;

determine whether to replace the first machine-learning model with the second machine-learning model; and

in response to determining to replace the first machine-learning model with the second machine-learning model, replace the first machine-learning model with the second machine-learning model in the production environment such that a next plurality of requests are input to the second machine-learning model.

16. The computer program product of claim 15, where the plurality of requests are processed as a batch by executing the first machine-learning model for each input of the plurality of inputs associated with the plurality of requests.

17. The computer program product of claim 16, wherein the plurality of requests are processed by a batch processor, and wherein the at least one processor is further programmed or configured to generate a dashboard interface configured to communicate with the batch processor.

18. The computer program product of claim 15, wherein determining whether to replace the first machine-learning model with the second machine-learning model comprises comparing the first machine-learning model to the second machine-learning model.

19. The computer program product of claim 18, wherein determining whether to replace the first machine-learning model with the second machine-learning model is based on at least one of computation efficiency and accuracy.

20. The computer program product of claim 15, wherein the at least one rule is based on at least one of model score and feature distribution.

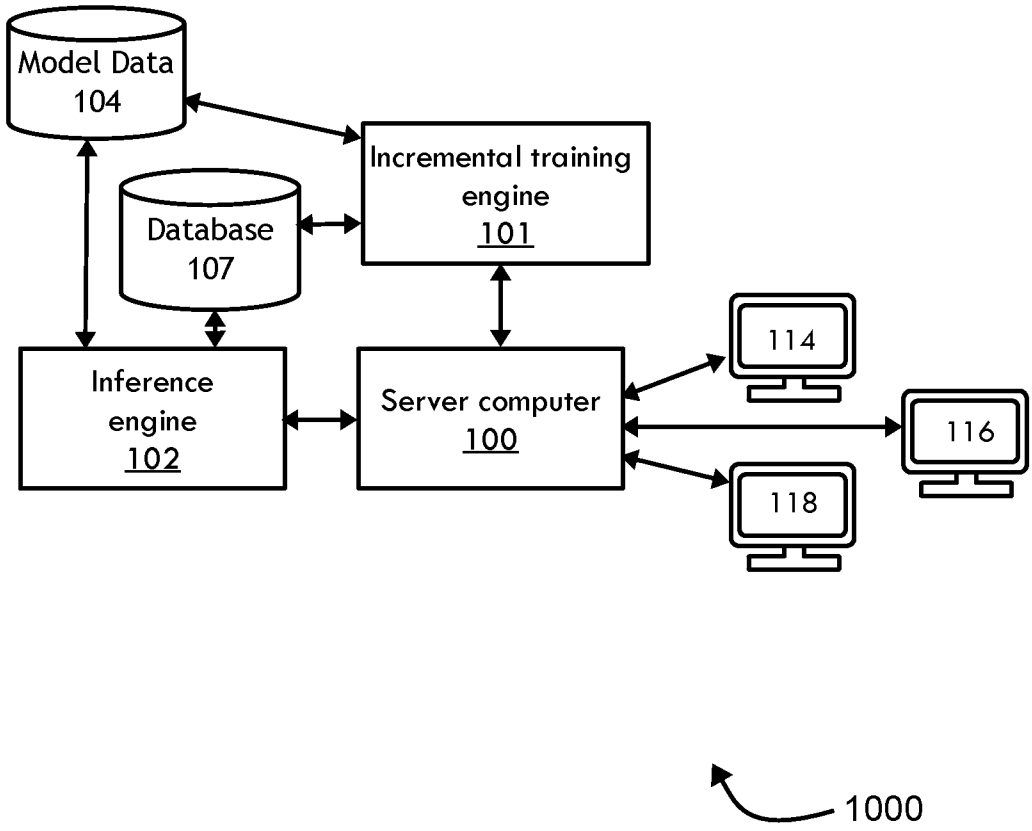
Model Data
104
Incremental training
engine
101
Database
107
Inference
engine
102
Server computer
100
114
116
118
1000

FIG. 1

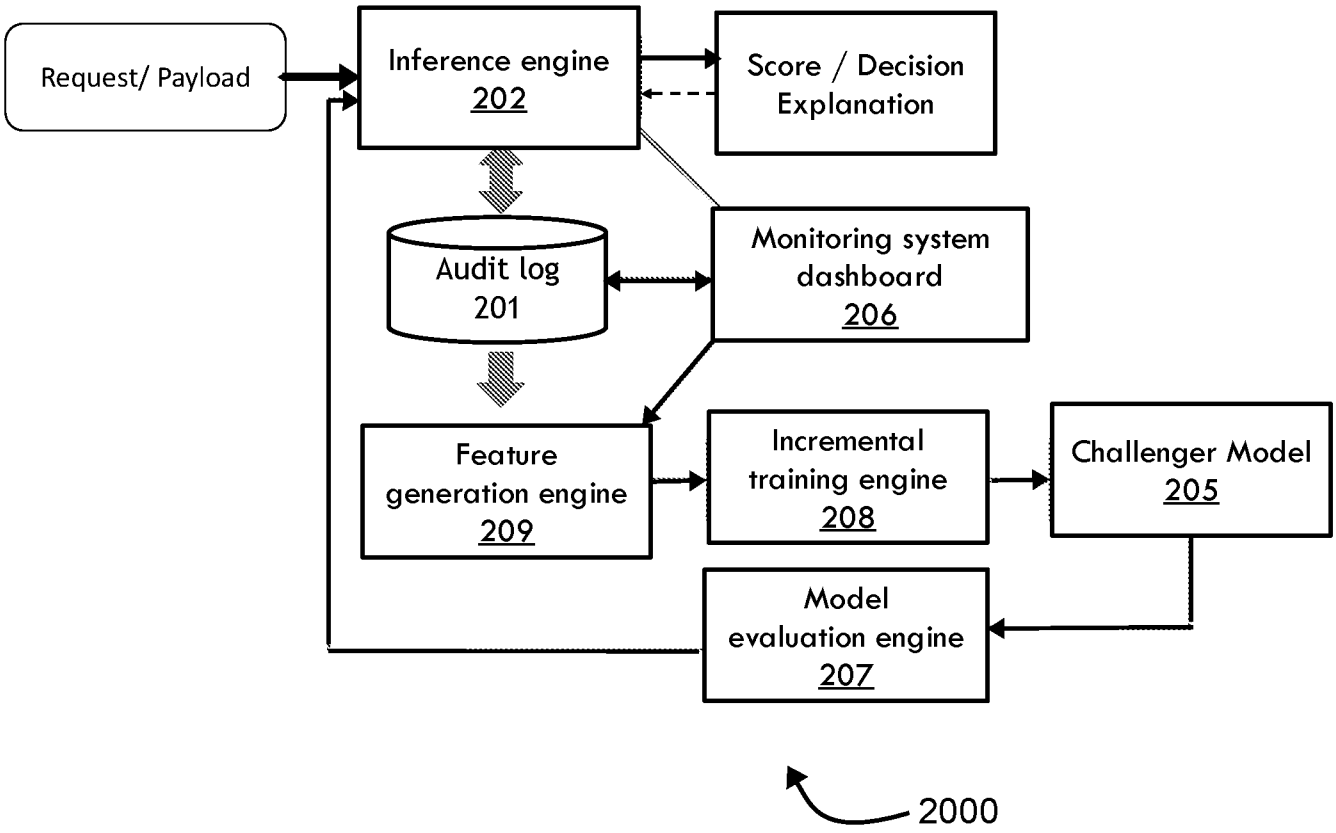
Request/ Payload
Inference engine
202
Score / Decision
Explanation
Audit log
201
Monitoring system
dashboard
206
Feature
generation engine
209
Incremental
training engine
208
Challenger Model
205
Model
evaluation engine
207
2000

FIG. 2

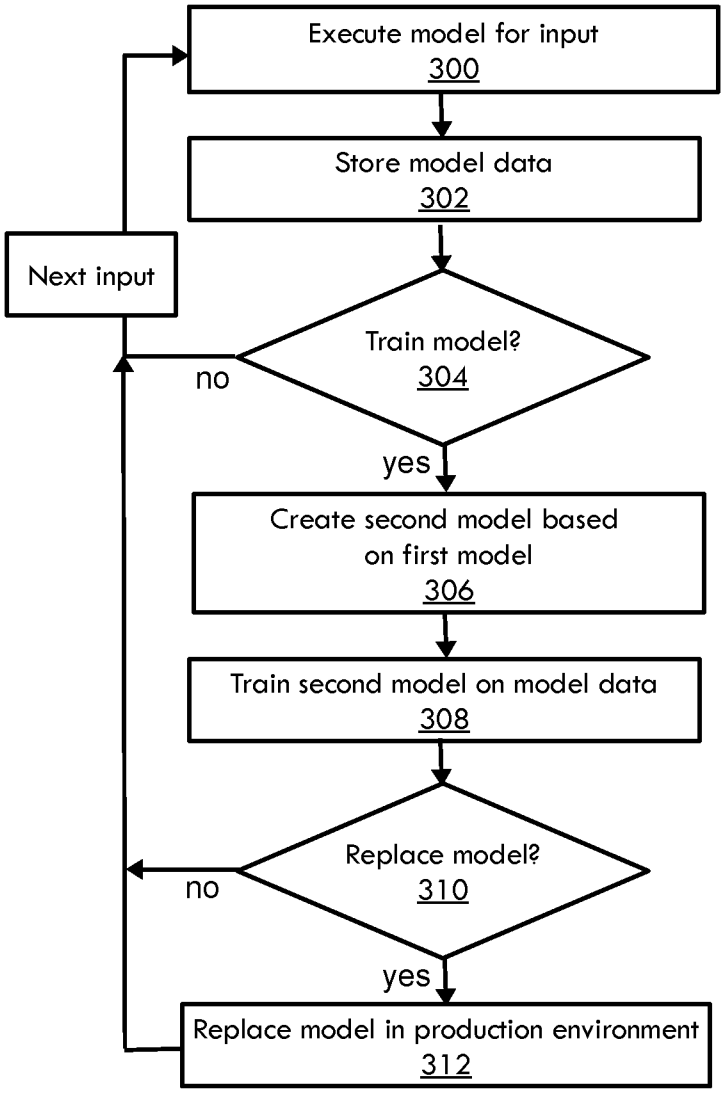
Execute model for input
300
Store model data
302
Next input
Train model?
304
no
yes
Create second model based on first model
306
Train second model on model data
308
Replace model?
310
no
yes
Replace model in production environment
312

FIG. 3

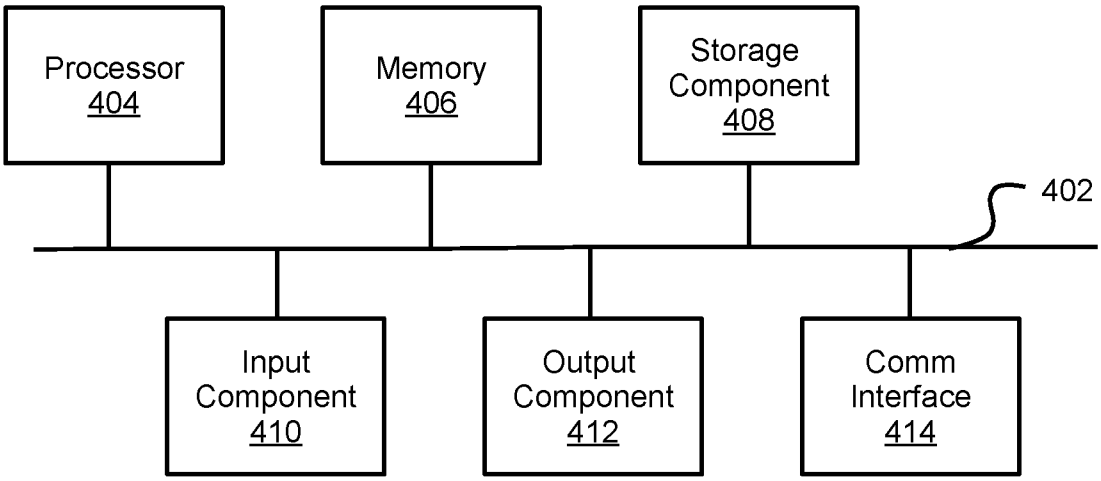
Processor
404
Memory
406
Storage
Component
408
402
Input
Component
410
Output
Component
412
Comm
Interface
414

FIG. 4


400