

SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR MODEL MONITORING USING BATCHED INFERENCECROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims the benefit of U.S. Provisional Application No. 63/523,419, filed June 27, 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 model monitoring using batched inferences.

2. Technical Considerations

[0003] Real-time inference platforms to monitor a machine-learning model in production have several technical limitations. For example, some metrics related to a model performance may be difficult and resource-intensive to generate, since it may influence the model's latency from a service level agreement (SLA) side. Further, existing monitoring systems and dashboards lack the computational resources to run those model-based metrics.

SUMMARY

[0004] According to non-limiting embodiments or aspects, provided is a system including at least one processor configured to: receive a plurality of processing requests to execute at least one machine-learning model from a plurality of systems; execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output including a decision; communicate the first output to a corresponding system of the plurality of systems; and further process the plurality of processing requests in a batch process by: generating at least one batch including the plurality of processing requests; assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second

output for each processing request, the second output associated with a corresponding decision of the first output.

[0005] In non-limiting embodiments or aspects, the at least one processor may be further configured to: receive a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and for each follow-on request: determine the processing request of the plurality of processing requests associated with the follow-on request; retrieve the second output associated with the processing request; and communicate the second output to a corresponding system of the at least a subset of systems.

[0006] In non-limiting embodiments or aspects, the second output may include at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0007] In non-limiting embodiments or aspects, the second output may include at least one reason code based on inputted transaction data.

[0008] In non-limiting embodiments or aspects, each processing request may be associated with an authorization request for an electronic payment transaction, the decision may include an authorization decision, and the second output may include at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0009] In non-limiting embodiments or aspects, generating the at least one batch may include generating a matrix including data associated with the plurality of processing requests, where the matrix is input to the at least one machine-learning model.

[0010] In non-limiting embodiments or aspects, executing the machine-learning model may include recording model metadata to an audit log, and where executing the at least one machine-learning model for the at least one batch may be based on model metadata for the corresponding processing requests of the at least one batch.

[0011] In non-limiting embodiments or aspects, the at least one processor may be further configured to process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting a model decision tree associated with the processing request into a plurality of subtrees; and assigning each subtree to a processing thread of a plurality of parallel processing threads.

[0012] In non-limiting embodiments or aspects, the at least one processor may be further configured to: merge at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model, where executing the at least one machine-learning model for the at least one batch may include executing the merged machine-learning model.

[0013] In non-limiting embodiments or aspects, processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests may include: monitoring model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0014] In non-limiting embodiments or aspects, the at least one processor may be further configured to: generate an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0015] In non-limiting embodiments or aspects, the at least one processor may be further configured to: determine that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-train the at least one machine-learning model.

[0016] In non-limiting embodiments or aspects, generating the second output may include generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0017] In non-limiting embodiments or aspects, each of the assigned processors of the plurality of processors may include a graphics processing unit (GPU) and/or a tensor processing unit (TPU).

[0018] According to non-limiting embodiments or aspects, provided is a computer-implemented method including: receiving, with at least one processor, a plurality of processing requests to execute at least one machine-learning model from a plurality of systems; executing, with at least one processor, the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output including a decision; communicating, with at least one processor, the first output to a corresponding system of the plurality of systems; and further processing the plurality of processing requests in a batch process by: generating, with at least one processor, at least one batch including the plurality of processing requests; assigning, with at least one processor, each of the plurality of processing requests in the at least

one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing, with at least one processor, each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0019] In non-limiting embodiments or aspects, the method may further include: receiving, with at least one processor, a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and for each follow-on request: determining, with at least one processor, the processing request of the plurality of processing requests associated with the follow-on request; retrieving, with at least one processor, the second output associated with the processing request; and communicating, with at least one processor, the second output to a corresponding system of the at least a subset of systems.

[0020] In non-limiting embodiments or aspects, the second output may include at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0021] In non-limiting embodiments or aspects, the second output may include at least one reason code based on inputted transaction data.

[0022] In non-limiting embodiments or aspects, each processing request may be associated with an authorization request for an electronic payment transaction, the decision may include an authorization decision, and the second output may include at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0023] In non-limiting embodiments or aspects, generating the at least one batch may include generating a matrix including data associated with the plurality of processing requests, where the matrix may be input to the at least one machine-learning model.

[0024] In non-limiting embodiments or aspects, executing the machine-learning model may include recording model metadata to an audit log, and where executing the at least one machine-learning model for the at least one batch may be based on model metadata for the corresponding processing requests of the at least one batch.

[0025] In non-limiting embodiments or aspects, the method may further include processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting, with at least one processor, a model decision tree associated with the processing request into a plurality of subtrees; and assigning, with at least one processor, each subtree to a processing thread of a plurality of parallel processing threads.

[0026] In non-limiting embodiments or aspects, the method may further include: merging, with at least one processor, at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model, where executing the at least one machine-learning model for the at least one batch may include executing the merged machine-learning model.

[0027] In non-limiting embodiments or aspects, processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests may include: monitoring, with at least one processor, model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0028] In non-limiting embodiments or aspects, the method may further include generating, with at least one processor, an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0029] In non-limiting embodiments or aspects, the method may further include: determining, with at least one processor, that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-training, with at least one processor, the at least one machine-learning model.

[0030] In non-limiting embodiments or aspects, generating the second output may include generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0031] In non-limiting embodiments or aspects, each of the assigned processors of the plurality of processors may include a GPU and/or a TPU.

[0032] According to non-limiting embodiments or aspects, provided is a computer program product that includes 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: receive a plurality of processing requests to execute at

least one machine-learning model from a plurality of systems; execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output including a decision; communicate the first output to a corresponding system of the plurality of systems; and further process the plurality of processing requests in a batch process by: generating at least one batch including the plurality of processing requests; assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0033] In non-limiting embodiments or aspects, the program instructions may cause the at least one processor to: receive a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and for each follow-on request: determine the processing request of the plurality of processing requests associated with the follow-on request; retrieve the second output associated with the processing request; and communicate the second output to a corresponding system of the at least a subset of systems.

[0034] In non-limiting embodiments or aspects, the second output may include at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0035] In non-limiting embodiments or aspects, the second output may include at least one reason code based on inputted transaction data.

[0036] In non-limiting embodiments or aspects, each processing request may be associated with an authorization request for an electronic payment transaction, the decision may include an authorization decision, and the second output may include at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0037] In non-limiting embodiments or aspects, generating the at least one batch may include generating a matrix including data associated with the plurality of

processing requests, where the matrix may be input to the at least one machine-learning model.

[0038] In non-limiting embodiments or aspects, executing the machine-learning model may include recording model metadata to an audit log, and where executing the at least one machine-learning model for the at least one batch may be based on model metadata for the corresponding processing requests of the at least one batch.

[0039] In non-limiting embodiments or aspects, the program instructions may cause the at least one processor to process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting a model decision tree associated with the processing request into a plurality of subtrees; and assigning each subtree to a processing thread of a plurality of parallel processing threads.

[0040] In non-limiting embodiments or aspects, the program instructions may cause the at least one processor to: merge at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model, where executing the at least one machine-learning model for the at least one batch may include executing the merged machine-learning model.

[0041] In non-limiting embodiments or aspects, processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests may include: monitoring model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0042] In non-limiting embodiments or aspects, the program instructions may cause the at least one processor to: generate an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0043] In non-limiting embodiments or aspects, the program instructions may cause the at least one processor to: determine that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-train the at least one machine-learning model.

[0044] In non-limiting embodiments or aspects, generating the second output may include generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0045] In non-limiting embodiments or aspects, each of the assigned processors of the plurality of processors may include a GPU and/or a TPU.

[0046] Further non-limiting embodiments or aspects are set forth in the following numbered clauses:

[0047] Clause 1: A system comprising at least one processor configured to: receive a plurality of processing requests to execute at least one machine-learning model from a plurality of systems; execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision; communicate the first output to a corresponding system of the plurality of systems; and further process the plurality of processing requests in a batch process by: generating at least one batch comprising the plurality of processing requests; assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0048] Clause 2: The system of clause 1, the at least one processor further configured to: receive a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and for each follow-on request: determine the processing request of the plurality of processing requests associated with the follow-on request; retrieve the second output associated with the processing request; and communicate the second output to a corresponding system of the at least a subset of systems.

[0049] Clause 3: The system of clause 1 or 2, wherein the second output comprises at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0050] Clause 4: The system of any of clauses 1-3, wherein the second output comprises at least one reason code based on inputted transaction data.

[0051] Clause 5: The system of any of clauses 1-4, wherein each processing request is associated with an authorization request for an electronic payment transaction, the decision comprises an authorization decision, and the second output

comprises at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0052] Clause 6: The system of any of clauses 1-5, wherein generating the at least one batch comprises generating a matrix comprising data associated with the plurality of processing requests, wherein the matrix is input to the at least one machine-learning model.

[0053] Clause 7: The system of any of clauses 1-6, wherein executing the machine-learning model comprises recording model metadata to an audit log, and wherein executing the at least one machine-learning model for the at least one batch is based on model metadata for the corresponding processing requests of the at least one batch.

[0054] Clause 8: The system of any of clauses 1-7, wherein the at least one processor is further configured to process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting a model decision tree associated with the processing request into a plurality of subtrees; and assigning each subtree to a processing thread of a plurality of parallel processing threads.

[0055] Clause 9: The system of any of clauses 1-8, wherein the at least one processor is further configured to: merge at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model, wherein executing the at least one machine-learning model for the at least one batch comprises executing the merged machine-learning model.

[0056] Clause 10: The system of any of clauses 1-9, wherein processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests comprises: monitoring model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0057] Clause 11: The system of any of clauses 1-10, wherein the at least one processor is further configured to: generate an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0058] Clause 12: The system of any of clauses 1-11, wherein the at least one processor is further configured to: determine that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-train the at least one machine-learning model.

[0059] Clause 13: The system of any of clauses 1-12, wherein generating the second output comprises generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0060] Clause 14: The system of any of clauses 1-13, wherein each of the assigned processors of the plurality of processors comprises a GPU and/or a TPU.

[0061] Clause 15: A computer-implemented method comprising: receiving, with at least one processor, a plurality of processing requests to execute at least one machine-learning model from a plurality of systems; executing, with at least one processor, the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision; communicating, with at least one processor, the first output to a corresponding system of the plurality of systems; and further processing the plurality of processing requests in a batch process by: generating, with at least one processor, at least one batch comprising the plurality of processing requests; assigning, with at least one processor, each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing, with at least one processor, each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0062] Clause 16: The method of clause 15, further comprising: receiving, with at least one processor, a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and for each follow-on request: determining, with at least one processor, the processing request of the plurality of processing requests associated with the follow-on request; retrieving, with at least one processor, the second output associated with the processing request; and communicating, with at least one processor, the second output to a corresponding system of the at least a subset of systems.

[0063] Clause 17: The method of clause 15 or 16, wherein the second output comprises at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0064] Clause 18: The method of any of clauses 15-17, wherein the second output comprises at least one reason code based on inputted transaction data.

[0065] Clause 19: The method of any of clauses 15-18, wherein each processing request is associated with an authorization request for an electronic payment transaction, the decision comprises an authorization decision, and the second output comprises at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0066] Clause 20: The method of any of clauses 15-19, wherein generating the at least one batch comprises generating a matrix comprising data associated with the plurality of processing requests, wherein the matrix is input to the at least one machine-learning model.

[0067] Clause 21: The method of any of clauses 15-20, wherein executing the machine-learning model comprises recording model metadata to an audit log, and wherein executing the at least one machine-learning model for the at least one batch is based on model metadata for the corresponding processing requests of the at least one batch.

[0068] Clause 22: The method of any of clauses 15-21, further comprising processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting, with at least one processor, a model decision tree associated with the processing request into a plurality of subtrees; and assigning, with at least one processor, each subtree to a processing thread of a plurality of parallel processing threads.

[0069] Clause 23: The method of any of clauses 15-22, further comprising: merging, with at least one processor, at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model, wherein executing the at least one machine-learning model for the at least one batch comprises executing the merged machine-learning model.

[0070] Clause 24: The method of any of clauses 15-23, wherein processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests comprises: monitoring, with at least one processor, model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0071] Clause 25: The method of any of clauses 15-24, further comprising: generating, with at least one processor, an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0072] Clause 26: The method of any of clauses 15-25, further comprising: determining, with at least one processor, that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-training, with at least one processor, the at least one machine-learning model.

[0073] Clause 27: The method of any of clauses 15-26, wherein generating the second output comprises generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0074] Clause 28: The method of any of clauses 15-27, wherein each of the assigned processors of the plurality of processors comprises a GPU and/or a TPU.

[0075] Clause 29: 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: receive a plurality of processing requests to execute at least one machine-learning model from a plurality of systems; execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision; communicate the first output to a corresponding system of the plurality of systems; and further process the plurality of processing requests in a batch process by: generating at least one batch comprising the plurality of processing requests; assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0076] Clause 30: The computer program product of clause 29, wherein the program instructions cause the at least one processor to: receive a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request

of the plurality of processing requests; and for each follow-on request: determine the processing request of the plurality of processing requests associated with the follow-on request; retrieve the second output associated with the processing request; and communicate the second output to a corresponding system of the at least a subset of systems.

[0077] Clause 31: The computer program product of clause 29 or 30, wherein the second output comprises at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

[0078] Clause 32: The computer program product of any of clauses 29-31, wherein the second output comprises at least one reason code based on inputted transaction data.

[0079] Clause 33: The computer program product of any of clauses 29-32, wherein each processing request is associated with an authorization request for an electronic payment transaction, the decision comprises an authorization decision, and the second output comprises at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

[0080] Clause 34: The computer program product of any of clauses 29-33, wherein generating the at least one batch comprises generating a matrix comprising data associated with the plurality of processing requests, wherein the matrix is input to the at least one machine-learning model.

[0081] Clause 35: The computer program product of any of clauses 29-34, wherein executing the machine-learning model comprises recording model metadata to an audit log, and wherein executing the at least one machine-learning model for the at least one batch is based on model metadata for the corresponding processing requests of the at least one batch.

[0082] Clause 36: The computer program product of any of clauses 29-35, wherein the program instructions cause the at least one processor to process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by: splitting a model decision tree associated with the processing request into a plurality of subtrees; and assigning each subtree to a processing thread of a plurality of parallel processing threads.

[0083] Clause 37: The computer program product of any of clauses 29-36, wherein the program instructions cause the at least one processor to: merge at least two layers of the at least one machine-learning model, resulting in a merged machine-learning

model, wherein executing the at least one machine-learning model for the at least one batch comprises executing the merged machine-learning model.

[0084] Clause 38: The computer program product of any of clauses 29-37, wherein processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests comprises: monitoring model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

[0085] Clause 39: The computer program product of any of clauses 29-38, wherein the program instructions cause the at least one processor to: generate an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

[0086] Clause 40: The computer program product of any of clauses 29-39, wherein the program instructions cause the at least one processor to: determine that the model metrics fail to satisfy a threshold; and in response to determining that the model metrics fail to satisfy the threshold, re-train the at least one machine-learning model.

[0087] Clause 41: The computer program product of any of clauses 29-40, wherein generating the second output comprises generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

[0088] Clause 42: The computer program product of any of clauses 29-41, wherein each of the assigned processors of the plurality of processors comprises a GPU and/or a TPU.

[0089] 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

[0090] 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:

- [0091] FIG. 1 is a schematic diagram of a system for model monitoring using batched inferences according to some non-limiting embodiments or aspects;
- [0092] FIG. 2 is a schematic diagram of a system for model monitoring using batched inferences according to some non-limiting embodiments or aspects;
- [0093] FIG. 3 is a schematic diagram of a system for model monitoring using batched inferences according to some non-limiting embodiments or aspects;
- [0094] FIG. 4 is a model time series graph according to some non-limiting embodiments or aspects;
- [0095] FIG. 5 is a feature contribution graph according to some non-limiting embodiments or aspects;
- [0096] FIG. 6 is a schematic diagram of trees and subtrees according to some non-limiting embodiments or aspects;
- [0097] FIG. 7 is a schematic diagram of a layered neural network system according to some non-limiting embodiments or aspects;
- [0098] FIG. 8 is a flow diagram of a method for model monitoring using batched inferences according to some non-limiting embodiments or aspects;
- [0099] FIG. 9 is a schematic diagram of example components of one or more devices according to some non-limiting embodiments or aspects; and
- [0100] FIG. 10 is a schematic diagram of an electronic payment processing network according to some non-limiting embodiments or aspects.

DETAILED DESCRIPTION

[0101] 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 present disclosure 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 and non-limiting 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.

[0102] Some non-limiting embodiments or aspects are described herein in connection with thresholds. As used herein, satisfying a threshold may refer to a value being greater than the threshold, more than the threshold, higher than the threshold,

greater than or equal to the threshold, less than the threshold, fewer than the threshold, lower than the threshold, less than or equal to the threshold, equal to the threshold, etc.

[0103] 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. In addition, reference to an action being “based on” a condition may refer to the action being “in response to” the condition. For example, the phrases “based on” and “in response to” may, in some non-limiting embodiments or aspects, refer to a condition for automatically triggering an action (e.g., a specific operation of an electronic device, such as a computing device, a processor, and/or the like).

[0104] 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.

[0105] An “application program interface” (API) refers to computer code or other data stored 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.).

[0106] 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).

[0107] 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. In some non-limiting embodiments or aspects, a message may refer to a network packet (e.g., a data packet and/or the like)

that includes data. It will be appreciated that numerous other arrangements are possible.

[0108] 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.

[0109] 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.

[0110] 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.

[0111] 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).

[0112] 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.

[0113] As used herein, the term “point-of-sale (POS) device” may refer to one or more devices, which may be used by a merchant to conduct a transaction (e.g., a payment transaction) and/or process a transaction. For example, a POS device may include one or more client devices. Additionally or alternatively, a POS device may include peripheral devices, card readers, scanning devices (e.g., code scanners), Bluetooth® communication receivers, near-field communication (NFC) receivers, RFID receivers, and/or other contactless transceivers or receivers, contact-based receivers, payment terminals, and/or the like.

[0114] As used herein, the term “point-of-sale (POS) system” may refer to one or more client devices and/or peripheral devices used by a merchant to conduct a transaction. For example, a POS system may include one or more POS devices and/or other like devices that may be used to conduct a payment transaction. In some non-limiting embodiments, a POS system (e.g., a merchant POS system) may include one or more server computers programmed or configured to process online payment transactions through webpages, mobile applications, and/or the like.

[0115] 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.”

[0116] As used herein, the term “system” may refer to one or more computing devices or combinations of computing devices (e.g., processors, servers, client devices, software applications, components of such, and/or the like). Reference to “a device,” “a server,” “a processor,” and/or the like, as used herein, may refer to a previously-recited device, server, or processor that is recited as performing a previous step or function, a different device, server, or processor, and/or a combination of devices, servers, and/or processors. For example, as used in the specification and the claims, a first device, a first server, or a first processor that is recited as performing a first step or a first function may refer to the same or different device, server, or processor recited as performing a second step or a second function.

[0117] 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.

[0118] Non-limiting embodiments or aspects of the disclosed subject matter are directed to systems, methods, and computer program products for model monitoring using batched inferences. Non-limiting embodiments allow for computational efficiencies to be achieved through batching follow-on requests relating to previously-determined model outputs in a real-time decision system.

[0119] A problem arising specifically in the realm of black box machine learning models is a lack of transparency of such models in providing the reasoning behind their decisions to the user. Commonly, only the input parameters and the output decision are known. Non-limiting embodiments or aspects enable transparency in such black box models by generating explainable metrics without delaying delivery of the model decision itself to the client. The system may generate the model decision in

real-time relative to receiving the processing request and transmit the model decision to the client device in real-time to meet timing requirements of the client.

[0120] After generation of the first output, at least a subset of the processing requests may be further processed in a batch process to generate a second output associated with the first output. The second output may comprise explainable metrics providing a human-interpretable reason for the decision of the model. This second output may be made available to the client device as soon as generated and/or in response to a follow-on and/or challenge request from the client device. Generation of the second output and its availability to the client enable the problem of the lack of transparency inherent to black box models to be overcome.

[0121] Non-limiting embodiments or aspects use an algorithm to generate the second output that enables individual predictions to be evaluated to identify which individual predictions generated by the model are incorrect. Use of such an algorithm enables monitoring of the model for errors, which makes debugging thereof easier. Further, that the algorithm can monitor the health of the model over time based on a shift in the distribution of the features contributing to its decisions enables the model to automatically be re-trained in response to determining that model metrics used to monitor the model have exceeded a threshold. In some non-limiting examples, the algorithm may comprise a Shapley algorithm.

[0122] Non-limiting embodiments or aspects also enhance the processing speed of the model in generating explainable metrics (e.g., the second output). For example, the model may split a model decision tree into a plurality of subtrees and assign the subtrees to separate, parallel processing threads that process each subtree. The results of the processing of the subtrees may be summed to approximate the value of the decision tree as a whole. Splitting the model decision tree and executing the subtrees may enhance the speed with which the model decision tree is analyzed. For example, the model may merge at least two layers of a multi-layer machine-learning model to form a merged machine-learning model approximating the initial model but capable of being executed faster. The merged machine-learning model may be executed instead of the original, unmerged machine-learning model.

[0123] Referring now to FIG. 1, shown is a system for model monitoring using batched inferences, according to some non-limiting embodiments or aspects. The system comprises at least one processor configured to: receive a plurality of processing requests to execute at least one machine-learning model from a plurality

of systems; execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision; communicate the first output to a corresponding system of the plurality of systems; and further process the plurality of processing requests in a batch process by: generating at least one batch comprising the plurality of processing requests; assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

[0124] With continued reference to FIG. 1, a system 1000 for model monitoring using batched inferences 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 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 requests, authorization requests, and/or the like). For example, client devices 114, 116, 118 may each include issuer systems, and the 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.

[0125] With continued reference to FIG. 1, the server computer 100 may be in communication with a batch inference engine 102. The batch 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 batch inference engine 102 may be part of the server computer 100. In some examples, the batch inference engine 102 may be separate and/or remote from the server computer 100. The batch

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 batch inference engine 102 may also be in communication with a database 107, which may include an audit log (see FIG. 2) for each execution of a model, model metric data, and/or the like.

[0126] In operation, one or more machine-learning models (e.g., such as a model stored as model data 104) may be executed by the server computer 100 and/or an inference engine (not shown in FIG. 1) separate from batch inference engine 102 upon request from each client device 114, 116, 118. An output of the machine-learning model(s), such as a risk score and/or the like, is returned to the client device 114, 116, 118 that requested it. Model metadata, such as inputs of the request and/or output features and/or scores resulting from execution of the model, may be stored in the database 107 or elsewhere. One or more of the client devices 114, 116, 118 may, after receiving the output, request additional information (e.g., such as a reason) relating to the output (e.g., why a risk score was determined to be a particular value). This additional information may be reflective of model metadata stored in the database 107 and/or with model data 104 resulting from execution of the machine-learning model. The request for additional information may be made, for example, through one or more APIs.

[0127] The server computer 100, in response to receiving a predetermined number of requests, in response to elapsing of a predetermined time interval, and/or in response to another event, may batch a plurality of requests and communicate a batch processing request to the batch inference engine 102. The batch processing request may include, for example, a plurality of follow-on requests made by client devices 114, 116, 118. The batch inference engine 102 may then batch process the requests by assigning each follow-on request to a processor of a plurality of processors, such as a GPU and/or TPU. The batch inference engine 102 may then process each of the follow-on requests in parallel with at least one other follow-on request on different processing threads. The follow-on requests may be processed by re-executing the machine-learning model based on the model metadata stored in the database 107 from a previous model execution. For example, the model metadata may include inputs, weights, model versions, and/or the like to permit the batch inference engine 102 to re-execute the model in the manner in which it was previously executed for a corresponding real-time processing request.

[0128] The number and arrangement of systems and devices shown in FIG. 1 are provided as an example. There may be additional systems and/or devices, fewer systems and/or devices, different systems and/or devices, and/or differently arranged systems and/or devices than those shown in FIG. 1. Furthermore, two or more systems or devices shown in FIG. 1 may be implemented within a single system or device, or a single system or device shown in FIG. 1 may be implemented as multiple, distributed systems or devices. Additionally or alternatively, a set of systems (e.g., one or more systems) or a set of devices (e.g., one or more devices) of system 1000 may perform one or more functions described as being performed by another set of systems or another set of devices of system 1000.

[0129] Referring to FIG. 2, a system 2000 for model monitoring using batched inferences is shown according to some non-limiting embodiments or aspects. A real-time model inference engine 200 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. For example, a payload may include input data for a decision model. In non-limiting embodiments, the real-time model inference engine 200 may individually execute one or more models upon request. The system 2000 may provide reasoning associated with an output (e.g., after generation of the initial decision (e.g., first output)) by batching a plurality of processing requests with a batch inference engine 202. 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 batch inference engine 202. The batch inference engine 202 may assign processing tasks to different processing threads executed by different processors, such as GPUs and/or TPUs.

[0130] With continued reference to FIG. 2, the batch 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 graphical user interfaces (GUIs) to users to show graphical representations of data in the audit log 201 in addition to explainable metrics 204. 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).

[0131] In non-limiting embodiments, the follow-on requests may be processed in an accelerated manner. For example, parallel processing threads may be used for

each subtree of a plurality of subtrees forming a larger model decision tree. In such an example, the model may be split into subtrees to facilitate parallel processing. In non-limiting embodiments, the model may be merged (e.g., two or more layers may be merged) to form a merged machine-learning model. In such an example, the merged machine-learning model may facilitate an accelerated execution of the model. In non-limiting embodiments, model processing may be accelerated by truncating inputs.

[0132] With continued reference to FIG. 2, in some non-limiting embodiments or aspects, real-time model inference engine 200 may receive a plurality of processing requests from client devices 203a-n. Each processing request may cause the real-time model inference engine 200 to execute a machine-learning model. The machine learning model may be a component of the real-time model inference engine 200, or the real-time model inference engine 200 may send a request to a machine-learning model separate therefrom. Each client device 203a-n may be associated with a separate system (e.g., separate issuer systems, separate merchant systems, separate acquirer systems, separate payment gateway systems, and the like).

[0133] Each processing request may comprise a payload comprising a data packet comprising data associated with the request to be executed by the machine-learning model. In some non-limiting embodiments or aspects, the processing requests may be associated with electronic payment transactions, such as authorization requests associated with electronic payment transactions that request a determination as to whether the electronic payment transaction should be approved or declined. In such requests, the data contained in the data packet may comprise transaction data associated with the electronic payment transaction. While discussed in terms of processing requests associated with electronic payment transactions, it will be appreciated that the processing requests may be any sort of request for a machine-learning model to analyze data and generate an output.

[0134] With continued reference to FIG. 2, in response to receiving the processing requests, the machine-learning model may execute each processing request to generate a first output. Executing the processing request may comprise the machine-learning model analyzing the data in the data packet and generating the first output based on the data. The first output may comprise a decision. The first output may not comprise explainable metrics explaining and/or justifying the decision of the machine-learning model and/or how the machine-learning model arrived at its decision. In some

non-limiting embodiments or aspects in which the processing request is associated with an authorization request of an electronic payment transaction, the data packet may comprise the transaction data, and the machine-learning model may analyze the transaction data and generate the first output. The first output may comprise a decision associated with the electronic payment transaction, such as an authorization decision (e.g., to authorize and/or decline the electronic payment transaction).

[0135] In response to the machine-learning model generating the first output, the real-time model inference engine 200 may communicate the first output to the client device of the system corresponding to the system that communicated the processing request. For example, with reference to FIG. 2, client device 1 203a and client device n 203n may each communicate a processing request to real-time model inference engine 200. Real-time model inference engine 200 may generate (using the machine-learning model) a first output for the processing request received from client device 1 203a and communicate the first output to client device 1 203a, and real-time model inference engine 200 may generate (using the machine-learning model) a first output for the processing request received from client device n 203n and communicate the first output to client device n 203n.

[0136] The real-time model inference engine 200 may generate and communicate the first output in real-time relative to receiving the corresponding processing request (e.g., in real-time, in near real-time, during the event, as soon as practically available after the event, during processing and/or communication of messages related to the event, at the time of making a decision (e.g., authorization decision, authentication decision, and/or the like) related to the 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). For example, the term “real-time” may refer to performance of a task or tasks during another process or before another process is completed. In some non-limiting embodiments or aspects, the real-time model inference engine 200 may generate and communicate the first output less than 5000 μ sec, or any other time that may be imperceptible or near imperceptible to a user under the circumstance, after receiving the corresponding processing request, such as less than 1000 μ sec or less than 500 μ sec.

[0137] In some non-limiting embodiments or aspects, the real-time model inference engine 200 may record and store model metadata resulting from execution of the machine-learning model(s) in the audit log 201 such that the model metadata is

available to the batch inference engine 202. The model metadata may comprise the data from the data packet used to execute the machine-learning model, the first output (e.g., including the decision), the timing associated with execution of the machine-learning model, an identifier of the system and/or client device to which the first output is communicated, a unique identifier associated with execution of the machine-learning model, and/or the like.

[0138] With continued reference to FIG. 2, and also referring to FIG. 3, after generation of the first output by the machine-learning model, the batch inference engine 202, 302 may further process the processing requests. In some non-limiting embodiments or aspects, each processing request received by the real-time model inference engine 200 may be further processed as described herein following the generation of the first output. In some non-limiting embodiments or aspects, some subset of the processing requests received by the real-time model inference engine 200 may be further processed as described herein following the generation of the first output. In some non-limiting embodiments or aspects, processing requests received by the real-time model inference engine 200 may be further processed as described herein in response to receiving a further request (e.g., a follow-on) from the corresponding client device. For example, the follow-on request (e.g., a second request) may comprise a second payload comprising a data packet comprising data associated with the follow-on request, which data packet may identify the associated first request and/or, in scenarios in which the processing request is associated with electronic payment transactions, an identifier identifying the electronic payment transaction associated with the first request. The payload may identify the first output received in response to the first request.

[0139] Referring to FIG. 3, the processing requests may be further processed in a batch process by a dynamic batcher 303 generating at least one batch comprising a plurality of processing requests for which a first output by the machine-learning model has already been generated. The dynamic batcher 303 may include one or more computing devices and/or software applications executed by one or more computing devices. The dynamic batcher 303 may receive and/or retrieve data associated with processing requests for which a first output by the machine-learning model has already been generated from the audit log 301 and generate the batch comprising a plurality of processing requests based on the data from the audit log 301. In some non-limiting embodiments or aspects, the dynamic batcher 303 may generate batches comprising

a plurality of processing requests for further processing by generating a matrix comprising data associated with the plurality of processing requests. This matrix generated by the dynamic batcher may be input to the machine learning model (e.g., the model instance 308 described hereinafter) to cause generation of a second output. The matrix may be a data structure storing the relevant data associated with the plurality of processing requests contained in the matrix.

[0140] With continued reference to FIG. 3, in response to generating the batch comprising a plurality of processing requests, the dynamic batcher 303 may assign the matrix and/or the plurality of processing requests to at least one processor of a plurality of processors. The batch inference engine may comprise a plurality of parallel processors 304a-k configured to execute a model instance 308 (e.g., the machine-learning model) for each of the processing requests in the batch received thereby. Each of the plurality of parallel processors 304a-k may include a GPU and/or a TPU, as examples, although various types of processors may be used in non-limiting embodiments. The non-limiting example of FIG. 3 shows each of the plurality of parallel processors 304a-k comprising a GPU. This plurality of parallel processors 304a-k may process batches of processing requests in parallel with each other.

[0141] Each processor of the plurality of parallel processors 304a-k, such as the first processor 304a, may comprise a plurality of threads 306a-n, and each thread may execute a sequence of program instructions that enable the first processor 304a to perform a plurality of tasks concurrently. The first processor 304a may execute the model instance 308, which may be an instance of the machine-learning model (as previously described) and may be configured to receive an input and generate an output in the same manner as the machine-learning model. The model instance 308 may obtain the same output as the machine-learning model based on the same input. The first processor 304a may also include memory and/or be in communication with a local database 310 to store data received by and/or generated by the first processor 304a.

[0142] With continued reference to FIG. 3, the first processor 304a may receive a batch of processing requests (e.g., a matrix containing the processing requests) from the dynamic batcher 303 and may further process each processing request by executing the model instance 308 for each of the processing request in the batch. The model instance 308 may generate a second output for each processing request.

[0143] The k th processor 304k may also receive a batch of processing requests (e.g., a matrix containing the processing requests) from the dynamic batcher 303 and may further process each processing request in the same manner as and in parallel with the processing requests processed by the first processor 304a. As such, each processing request executed by the model instance 308 of the parallel processors 304a-k may be processed in parallel with another processing request executed by the model instance 308 of a different processor of the parallel processors 304a-k.

[0144] The second output generated by the model instance 308 may be associated with a corresponding decision of the first output. The decision of the first output may be based on one or more of a plurality of parameters. The second output may comprise data identifying at least one parameter of a plurality of parameters having a higher impact on the decision of the first output (compared to the other of the plurality of parameters). The second output may comprise at least one reason code based on inputted transaction data. The reason code may represent (e.g., correspond to) one or more parameters determined by the model instance 308 to have a higher impact on the decision of the first output (compared to other parameters of the plurality of parameters).

[0145] Referring to FIG. 5, a feature contribution graph 5000 is shown according to some non-limiting embodiments or aspects. The second output may comprise the feature contribution graph 5000. The feature contribution graph 5000 may show a relative contribution of each of the plurality of features to the decision of the first output. In this exemplary feature contribution graph 5000 of FIG. 5, there are at least 20 features in the plurality of features, and 4 of the features contribute to the decision of the first output, with two features having a significantly higher contribution compared to the rest, and one feature clearly having the highest contribution. However, it will be appreciated that the feature contribution graph 5000 may show the relative contribution of each of the features for each different processing request for which a first output is generated.

[0146] Referring again to FIGS. 2-3, the second output generated by the batch inference engine 202, 302 may be transmitted to an explainable metrics processor 204. The explainable metrics processor 204 may include one or more computing devices and/or software applications executed by one or more computing devices. The explainable metrics processor 204 may receive the second output and generate a human interpretable output that enables a human to understand the parameter and/or

the reason code from the second output. Thus, the explainable metrics processor 204 may convert the computer-interpretable second output to a human-interpretable format, such as readable text, that enables a human to understand the reason(s) for the decision of the machine-learning model in the first input. The human-interpretable format may be transmitted to the client device 203a that submitted the follow-on request and displayed (e.g., on a user interface thereof) by the client device 203a in the human-interpretable format.

[0147] With continued reference to FIGS. 2-3, the system 2000, 3000 may receive a plurality of follow-on requests from at least a subset of client devices of the plurality of client devices 203a-n. Each follow-on request may be associated with an (initial) processing request of the plurality of processing requests. For example, a client device 203a-n may transmit a follow-on request to challenge and/or request further information about the model decision generated in the first output.

[0148] For each follow-on request, the system 2000, 3000 may determine the processing request of the plurality of processing requests associated with the follow-on request (e.g., the processing request for which the follow-on request is requesting further information). For each follow-on request, the system 2000, 3000 may retrieve the second output associated with the processing request. In some non-limiting embodiments, the second output may be generated in response to receiving the follow-on request associated with the processing request. In other embodiments, the second output may have already been generated when the follow-on request is received. For each follow-on request, the system 2000, 3000 may communicate the second output to the corresponding client device 203a.

[0149] Referring to FIG. 2, the second output may be transmitted to a monitoring system dashboard 206. The monitoring system dashboard 206 may also obtain (e.g., receive and/or retrieve) data from the audit log 201 associated with the processing requests processed by the system 2000. The monitoring system dashboard 206 may comprise a graphical user interface configured to display data. In some non-limiting embodiments or aspects, further processing each processing request of the plurality of processing requests in parallel with one another may comprise monitoring, with the monitoring system dashboard 206, model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed. Monitoring the model metrics may comprise monitoring relative contribution of the

each of the plurality of parameters to the decisions of the first output (e.g., based on the second output) over time.

[0150] FIG. 4 shows a model time series graph 4000 according to some non-limiting embodiments or aspects. The model time series graph 4000 may be used to monitor the model metrics over time. The model time series graph 4000 may track feature contribution (y-axis) for each of the plurality of features (x-axis) over time. For example, the model time series graph 4000 of FIG. 4 shows the contribution of the plurality of features over time (e.g., as a distribution) at the time of training the machine-learning model, at a first time T1 after training, and at a second time T2 after T1. Deviation of the distribution over time by more than a threshold amount may indicate deteriorating and/or poor health of the machine learning model, such that the accuracy of the output of the model may be in question. Deviation of the distribution over time by more than a threshold amount may automatically trigger re-training of the model. Deviation of the distribution over time by more than a threshold amount (e.g., an alert threshold) may automatically generate an alert. The alert may notify a user to take a corrective action with respect to the model, such as initiating re-training of the model.

[0151] While the model time series graph 4000 is shown as a line graph having the design shown in FIG. 4, it will be appreciated that any representation that enables the monitoring system dashboard 206 to track the distribution of the contribution of the plurality of features over time may be used, such as a bar graph format tracking the same features over time (e.g. similar to FIG. 5 but at multiple times).

[0152] Referring to FIG. 3, the model instance 308 may generate the second output (which may include generating explainable metrics) associated with a corresponding decision of the first output in any suitable manner. For example, the model instance 308 may execute a Shapley algorithm to generate the second output. The second output may comprise a Shapley value for each parameter of a plurality of parameters associated with the processing request. The Shapley algorithm may determine how the features affect an individual prediction (e.g. a score) and how much that feature affected (e.g., increased and/or decreased) that prediction score compared to average. Use of a Shapley algorithm in this context may improve debugging and/or re-training because a user can study incorrect predictions individually and why the prediction was incorrect. Further, use of a Shapley algorithm in this context may improve transparency of the model (e.g., including model logic impacting decisions

and/or output), which is a problem specifically arising with “black box” machine-learning models. A Shapley algorithm may include one or more formulae for determining the contribution of each parameter to each possible subset of parameters and/or the plurality of parameters, considering each parameter.

[0153] For example, the model instance 308 may be configured to further process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by splitting a model decision tree 600 associated with the processing request into a plurality of subtrees 600a-c, as shown in FIG. 6. Each subtree 600a-c may be assigned to a processing thread (e.g., a separate processing thread) of a plurality of processing threads 306a-n. The splitting of the model decision tree 600 into a plurality of subtrees 600a-c may result in acceleration in the processing of the second output.

[0154] Referring to FIG. 6, a model decision tree 600 is shown according to some non-limiting embodiments. The model decision tree 600 has a plurality of subcomponents (subcomponents A-C). The model decision tree 600 may be separated into a plurality of subtrees 600a-c. Each subtree 600a-c may start from the root to a single leaf (each of A-C). Each subtree's 600a-c value may be computed by the first processor 304a using different threads 306a-n in parallel, leading to accelerated processing of the model decision tree 600. In some examples the threads may be simultaneously processed by different cores of the first processor. The model decision tree's 600 value (e.g. Shapely value when using a Shapley algorithm) may be approximated as a sum of the subtrees 600a-c:

$$\varphi_i(x) = \sum_{j=A}^k \varphi_{i,j}(x).$$

[0155] Referring to FIGS. 3 and 7, the model instance 308 may generate the second output (e.g., including explainable metrics) associated with a corresponding decision of the first output in any suitable manner. For example, the model instance 308 may comprise a plurality of model layers and may be configured to further process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by processing an input based on the request in accordance with each of the plurality of layers in succession. In some non-limiting embodiments or aspects, executing the model instance 308 may comprise merging at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model. The model instance

308 may then execute the model for the at least one batch by executing the merged machine-learning model.

[0156] Referring to FIG. 7 layered neural network systems 700a-b, 702 are shown according to some non-limiting embodiments or aspects. Non-limiting examples of an unmerged layered system 700a-b are shown. A non-limiting example of a merged layered system 702 is shown.

[0157] The unmerged layered systems 700a-b may comprise one or more rectified linear unit (ReLU) layers, one or more batch normalization layers, and one or more convolutional and/or fully connected layers. In unmerged layer system 700a, a first ReLU layer is followed by a batch normalization layer, which is followed by a convolutional and/or fully connected layer, which is followed by a second ReLU layer, the opposite direction of which may be executed during back propagation. In unmerged layer system 700b, a first ReLU layer is followed by a convolutional and/or fully connected layer, which is followed by a batch normalization layer, which is followed by a second ReLU layer, the opposite direction of which may be executed during back propagation.

[0158] For unmerged layer system 700a, the unmerged layer system 700a may generate a forward inference ($\forall i, j$) (e.g. forward propagation) using the batch normalization layer followed by the convolutional and/or fully connected layer as follows:

Batch Normalization layer:

$$\text{norm}(x_i^{(l)}) = \lambda_i \frac{x_i^{l+} - \mu_i}{\sigma_i} + \beta_i$$

Convolutional and/or fully connected layer:

$$x_j^{(l+1)+} = \sum_i w_{i,j} \left(\text{norm}(x_i^{(l)}) \right) + b_{i,j}$$

[0159] For unmerged layer system 700a, the unmerged layer system 700a may generate a backward inference ($\forall i, j$) (e.g. backward propagation) as follows:

Backwards gradient:

$$\frac{\partial x_j^{(l+1)+}}{\partial x_i^{l+}} = \frac{\partial x_j^{(l+1)+}}{\partial \text{norm}(x_i^{(l)})} \circ \frac{\partial \text{norm}(x_i^{(l)})}{\partial x_i^{l+}}$$

[0160] For merged layered system 702, the batch normalization layer and the convolutional and/or fully connected layer may be merged to form a convolutional and/or fully connected layer representative of the unmerged layers. Applying the layer merging to both forward and backward propagation during inference time may make execution of the model faster. This faster processing speed may be realized by utilizing a Shapley algorithm, as the gradients from the layer merging may be simpler and may avoid applying auto-gradient and chain-rule through merged layers. The forward and backward inference for the merged layered system 702 may be as follows: Forward inference:

$$\begin{aligned} x_j^{(l+1)+} &= \sum_i w_{i,j} \left(\lambda_i \frac{x_i^{l+} - \mu_i}{\sigma_i} + \beta_i \right) + b_{i,j} \\ &= \sum_i \left(w_{i,j} \frac{\lambda_i}{\sigma_i} \right) x_i^{l+} + \left(\sum_i w_{i,j} \left(\lambda_i \frac{-\mu_i}{\sigma_i} + \beta_i \right) \right) + b_{i,j} \end{aligned}$$

Backward gradient during inference after layer merging:

$$\frac{\partial x_j^{(l+1)+}}{\partial x_i^{l+}} = w_{i,j} \frac{\lambda_i}{\sigma_i}$$

[0161] Referring again to FIGS. 2-3, as previously described, the systems 2000, 3000 may generate a first output and may also generate a separate second output for at least a subset of the processing requests of the plurality of processing requests. The first output may comprise the decision of the machine-learning model and not comprise explainable metrics explaining and/or justifying the decision of the machine-learning model and/or how the machine-learning model arrived at its decision. The first output may be generated in real-time relative (as previously defined herein) to

receiving the corresponding processing request. The second output may be associated with the first output and may comprise explainable metrics explaining and/or justifying the decision of the machine-learning model and/or how the machine-learning model arrived at its decision. The second output may not be generated in real-time relative to receiving the corresponding initial processing request, and in some non-limiting embodiments or aspects, the second output may be transmitted to a corresponding client device 203a in response to the client device 203a transmitting a follow-on request to the system 2000, 3000. In some non-limiting embodiments or aspects, the second output may not be generated until a follow-on request is received, while in other embodiments, the second output may automatically be generated before receiving the follow-on request and may be transmitted to the client device 203a in response to receiving the follow-on request or without receiving a follow-on request. The second output may take longer to generate than the first output due to the explainable metrics taking longer for the model to generate compared to just the decision itself.

[0162] Separating the first output and the second output in this manner may enable the system to more quickly respond to a processing request of a client device 203a with the decision in real-time to meet real-time processing requirements of the client, while still following-on with explainable metrics of the decision (e.g., the second output) at a later time and/or only if requested (e.g., via a challenge request and/or follow-on request). For example, a first system as described herein was prepared to, in a first scenario, return only the model decision (without explainable metrics) in response to a processing request as soon as possible, while a second system was prepared to, in a second scenario, return the model decision together with explainable metrics (e.g., the first and second output together) in response to a processing request as soon as possible. In the first scenario, the model decision (without explainable metrics) was generated and transmitted to the client device 203a in response to a processing request within 355 μ sec. In contrast, in the second scenario, the model decision together with explainable metrics were generated and transmitted to the client device 203a in response to a processing request within 8640 μ sec. As can be seen from this non-limiting experimental example, generating and returning the decision without explainable metrics can be completed faster compared to if the explainable metrics are transmitted together with the decisions. As such, it can be seen that systems described herein lower the latency time for a client device 203a to receive the model

decision, and enable the explainable metrics to be provided later, and optionally only if challenged and/or requested.

[0163] In some non-limiting embodiments or aspects, the processing request may be associated with an authorization request for an electronic payment transaction. The decision of the first output may be an authorization decision to approve or decline the electronic payment transaction. For example, a transaction processing system of a transaction service provider and/or an issuer system of an issuer may generate the authorization decision. The second output may comprise at least one parameter and/or reason code associated with transaction data of the electronic payment transaction. The at least one parameter and/or reason code may comprise the parameter and/or reasons for the authorization decision (e.g., why the decision was to approve and/or decline the electronic payment transaction). As used herein, a “payment transaction decision” generated herein refers to a determination made in connection with the processing of an electronic payment transaction (e.g., a credit or debit card transaction), such as a payment authorization decision, a generated risk score, a fraud determination, and the like.

[0164] While the processing requests described herein are discussed in association with authorization requests associated with electronic payment transactions, it will be appreciated that the processing requests may be associated with other requests submitted to black box machine-learning models for decisions, recommendations, predictions, score, answers, determinations, outputs, and the like. As used herein, the terms “black box model” and “black box machine-learning model” refer to a model that receives an input and generates an output based on the input and one or more processes unknown to an end-user.

[0165] Referring now to FIG. 8, shown is a flow diagram for a method for model monitoring using batched inferences, according to some non-limiting embodiments or aspects. The steps shown in FIG. 8 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.

[0166] The method 8000 may comprise a step 802 of receiving, with at least one processor, a plurality of processing requests to execute at least one machine-learning model from a plurality of systems. In some non-limiting embodiments or aspects, the

real-time model inference engine 200 may receive the plurality of processing requests to execute at least one machine-learning model from the plurality of client devices 203a-n.

[0167] The method 8000 may comprise a step 804 of executing, with at least one processor, the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision. In some non-limiting embodiments or aspects, the real-time model inference engine 200 may execute the machine-learning model for each processing request.

[0168] The method 8000 may comprise a step 806 of communicating, with at least one processor, the first output to a corresponding system of the plurality of systems. In some non-limiting embodiments or aspects, the real-time model inference engine 200 may communicate the first output to the corresponding client device 203a-n.

[0169] The method 8000 may comprise a step 808 of further processing the plurality of processing requests in a batch process as described herein.

[0170] The method 8000 may comprise a step 810 of generating, with at least one processor, at least one batch comprising the plurality of processing requests. In some non-limiting embodiments or aspects, the dynamic batcher 303 of the batch inference engine 302 may generate the at least one batch.

[0171] The method 8000 may comprise a step 812 of assigning, with at least one processor, each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors. In some non-limiting embodiments or aspects, the dynamic batcher 303 of the batch inference engine 302 may assign each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of parallel processors 304a-k.

[0172] The method 8000 may comprise a step 814 of further processing, with at least one processor, each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output. In some non-limiting embodiments or aspects, each model instance 308 of each processor of a plurality of parallel processors 304a-k may be executed thereby to generate the second output.

[0173] Referring now to FIG. 9, shown is a diagram of example components of a device 900 according to non-limiting embodiments. Device 900 may correspond to server computer 100, batch inference engine 102, model data database 104, database 107, client devices 114, 116, 118 from FIG. 1; real-time model inference engine 200, audit log 201, batch inference engine 202, client device 203a-n, explainable metrics 204, monitoring system dashboard 206 from FIG. 2; and audit log 301, batch inference engine 302, dynamic batcher 303, GPUs/TPUs 304a-k, threads 306a-n, model instance 308, and/or local database 310 from FIG. 3, as examples. In some non-limiting embodiments, such systems or devices may include at least one device 900 and/or at least one component of device 900. The number and arrangement of components shown are provided as an example. In some non-limiting embodiments, device 900 may include additional components, fewer components, different components, or differently arranged components than those shown. Additionally, or alternatively, a set of components (e.g., one or more components) of device 900 may perform one or more functions described as being performed by another set of components of device 900.

[0174] As shown in FIG. 9, device 900 may include a bus 902, a processor 904, memory 906, a storage component 908, an input component 910, an output component 912, and a communication interface 914. Bus 902 may include a component that permits communication among the components of device 900. In some non-limiting embodiments, processor 904 may be implemented in hardware, firmware, or a combination of hardware and software. For example, processor 904 may include a processor (e.g., a central processing unit (CPU), a graphics processing unit (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 906 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 904.

[0175] With continued reference to FIG. 9, storage component 908 may store information and/or software related to the operation and use of device 900. For example, storage component 908 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 910 may include a component that permits device 900 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 910 may include a sensor for sensing information (e.g., a global positioning system (GPS) component, an accelerometer, a gyroscope, an actuator, etc.). Output component 912 may include a component that provides output information from device 900 (e.g., a display, a speaker, one or more light-emitting diodes (LEDs), etc.). Communication interface 914 may include a transceiver-like component (e.g., a transceiver, a separate receiver and transmitter, etc.) that enables device 900 to communicate with other devices, such as via a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication interface 914 may permit device 900 to receive information from another device and/or provide information to another device. For example, communication interface 914 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.

[0176] Device 900 may perform one or more processes described herein. Device 900 may perform these processes based on processor 904 executing software instructions stored by a computer-readable medium, such as memory 906 and/or storage component 908. 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 906 and/or storage component 908 from another computer-readable medium or from another device via communication interface 914. When executed, software instructions stored in memory 906 and/or storage component 908 may cause processor 904 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.

[0177] FIG. 10 shows an electronic payment processing network 10000 according to non-limiting embodiments or aspects. The payment processing network may be used in conjunction with the systems and methods described herein. It will be appreciated that the particular arrangement of electronic payment processing network 10000 shown is for example purposes only, and that various arrangements are possible. Transaction processing system 1001 (e.g., a transaction handler) is shown to be in communication with one or more issuer systems (e.g., such as issuer system 1006) and one or more acquirer systems (e.g., such as acquirer system 1008). Although only a single issuer system 1006 and single acquirer system 1008 are shown, it will be appreciated that transaction processing system 1001 may be in communication with a plurality of issuer systems and/or acquirer systems. In some embodiments, transaction processing system 1001 may also operate as an issuer system such that both transaction processing system 1001 and issuer system 1006 are a single system and/or controlled by a single entity.

[0178] In some non-limiting embodiments or aspects, transaction processing system 1001 may communicate with merchant system 1004 directly through a public or private network connection. Additionally or alternatively, transaction processing system 1001 may communicate with merchant system 1004 through payment gateway 1002 and/or acquirer system 1008. In some non-limiting embodiments or aspects, an acquirer system 1008 associated with merchant system 1004 may operate as payment gateway 1002 to facilitate the communication of transaction requests from merchant system 1004 to transaction processing system 1001. Merchant system 1004 may communicate with payment gateway 1002 through a public or private network connection. For example, a merchant system 1004 that includes a physical POS device may communicate with payment gateway 1002 through a public or private network to conduct card-present transactions. As another example, a merchant system 1004 that includes a server (e.g., a web server) may communicate with payment gateway 1002 through a public or private network, such as a public Internet connection, to conduct card-not-present transactions.

[0179] In some non-limiting embodiments or aspects, transaction processing system 1001, after receiving a transaction request from merchant system 1004 that identifies an account identifier of a payor (e.g., such as an account holder) associated

with an issued payment device 1010 (e.g., consumer device), may generate an authorization request message to be communicated to the issuer system 1006 that issued the payment device 1010 and/or account identifier. Issuer system 1006 may then approve or decline the authorization request and, based on the approval or denial, generate an authorization response message that is communicated to transaction processing system 1001. Transaction processing system 1001 may communicate an approval or denial to merchant system 1004. When issuer system 1006 approves the authorization request message, it may then clear and settle the payment transaction between the issuer system 1006 and acquirer system 1008.

[0180] In some non-limiting embodiments or aspects, the systems from FIG. 1-3 may be a part of, for example, transaction processing system 1001, and the processing requests may be received from payment device 1010, merchant system 1004, payment gateway 1002, acquirer system 1008, and/or issuer system 1006. In some non-limiting embodiments or aspects, the systems from FIG. 1-3 may be a part of, for example, issuer system 1006, and the processing requests may be received from payment device 1010, merchant system 1004, payment gateway 1002, acquirer system 1008, and/or transaction processing system 1001.

[0181] 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 processor configured to:
 - receive a plurality of processing requests to execute at least one machine-learning model from a plurality of systems;
 - execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision;
 - communicate the first output to a corresponding system of the plurality of systems; and
 - further process the plurality of processing requests in a batch process by:
 - generating at least one batch comprising the plurality of processing requests;
 - assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and
 - further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.
2. The system of claim 1, the at least one processor further configured to:
 - receive a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and
 - for each follow-on request:
 - determine the processing request of the plurality of processing requests associated with the follow-on request;
 - retrieve the second output associated with the processing request; and

communicate the second output to a corresponding system of the at least a subset of systems.

3. The system of claim 1, wherein the second output comprises at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

4. The system of claim 1, wherein the second output comprises at least one reason code based on inputted transaction data.

5. The system of claim 1, wherein each processing request is associated with an authorization request for an electronic payment transaction, the decision comprises an authorization decision, and the second output comprises at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

6. The system of claim 1, wherein generating the at least one batch comprises generating a matrix comprising data associated with the plurality of processing requests, wherein the matrix is input to the at least one machine-learning model.

7. The system of claim 1, wherein executing the machine-learning model comprises recording model metadata to an audit log, and wherein executing the at least one machine-learning model for the at least one batch is based on model metadata for the corresponding processing requests of the at least one batch.

8. The system of claim 1, wherein the at least one processor is further configured to process each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by:

splitting a model decision tree associated with the processing request into a plurality of subtrees; and

assigning each subtree to a processing thread of a plurality of parallel processing threads.

9. The system of claim 1, wherein the at least one processor is further configured to:

merge at least two layers of the at least one machine-learning model, resulting in a merged machine-learning model,

wherein executing the at least one machine-learning model for the at least one batch comprises executing the merged machine-learning model.

10. The system of claim 1, wherein processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests comprises:

monitoring model metrics associated with the at least one machine-learning model while the at least one machine-learning model is executed.

11. The system of claim 10, wherein the at least one processor is further configured to:

generate an alert while monitoring the model metrics based on the model metrics satisfying at least one alert threshold.

12. The system of claim 10, wherein the at least one processor is further configured to:

determine that the model metrics fail to satisfy a threshold; and

in response to determining that the model metrics fail to satisfy the threshold, re-train the at least one machine-learning model.

13. The system of claim 1, wherein generating the second output comprises generating a Shapley value for each parameter of a plurality of parameters associated with the processing request.

14. The system of claim 1, wherein each of the assigned processors of the plurality of processors comprises a graphics processing unit (GPU) and/or a tensor processing unit (TPU).

15. A computer-implemented method comprising:

receiving, with at least one processor, a plurality of processing requests to execute at least one machine-learning model from a plurality of systems;

executing, with at least one processor, the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision;

communicating, with at least one processor, the first output to a corresponding system of the plurality of systems; and

further processing the plurality of processing requests in a batch process by:

generating, with at least one processor, at least one batch comprising the plurality of processing requests;

assigning, with at least one processor, each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and

further processing, with at least one processor, each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.

16. The method of claim 15, further comprising:

receiving, with at least one processor, a plurality of follow-on requests from at least a subset of systems of the plurality of systems, each follow-on request of the plurality of follow-on requests associated with a processing request of the plurality of processing requests; and

for each follow-on request:

determining, with at least one processor, the processing request of the plurality of processing requests associated with the follow-on request;

retrieving, with at least one processor, the second output associated with the processing request; and

communicating, with at least one processor, the second output to a corresponding system of the at least a subset of systems.

17. The method of claim 15, wherein the second output comprises at least one parameter of a plurality of parameters having a higher impact on the decision of the first output.

18. The method of claim 15, wherein the second output comprises at least one reason code based on inputted transaction data.

19. The method of claim 15, wherein each processing request is associated with an authorization request for an electronic payment transaction, the decision comprises an authorization decision, and the second output comprises at least one parameter and/or reason code associated with transaction data of the electronic payment transaction.

20. 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:

receive a plurality of processing requests to execute at least one machine-learning model from a plurality of systems;

execute the at least one machine-learning model for each processing request of the plurality of processing requests to generate a first output of the machine-learning model for each processing request, the first output comprising a decision;

communicate the first output to a corresponding system of the plurality of systems; and

further process the plurality of processing requests in a batch process by:

generating at least one batch comprising the plurality of processing requests;

assigning each of the plurality of processing requests in the at least one batch and/or a batch of the at least one batch to a processor of a plurality of processors; and

further processing each processing request of the plurality of processing requests in parallel with at least one other processing request of the plurality of processing requests by executing the at least one machine-learning model for the at least one batch to generate a second output for each processing request, the second output associated with a corresponding decision of the first output.