

NEURAL NETWORKS FOR MIXED DATA TYPES

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] The present application is a PCT application of and claims priority to U.S. Provisional Application 63/621,951, filed on January 17, 2024, which is incorporated herein
5 by reference in its entirety.

BACKGROUND

[0002] Many real-world datasets contain both quantitative and categorical features. Quantitative features can include weight, distance, area, transaction amount, accumulated transaction volume, etc. Categorical features can include features such as school grades, card
10 type, device, email domain, IP address, shipping address, resource provider identifier, etc.

[0003] Existing deep neural networks have limitations on dealing with these mixed-types of data. Current deep neural networks are not able to handle categorical features directly, especially those with large number of categories. Current deep neural networks are not able to capture the interactions among all quantitative and categorical features. These
15 problems with deep neural networks result in poor performance when compared with tree-based models (e.g., XGBoost).

[0004] Embodiments of the disclosure address this problem and other problems individually and collectively.

SUMMARY

20 [0005] One embodiment is related to a method performed by a computer system. The method includes the computer system obtaining input data comprising quantitative data and categorical data. The categorical data can correspond to a plurality of categories. The computer system can determine one or more quantitative embeddings for the quantitative data. The computer system can determine a plurality of categorical embeddings for the
25 plurality of categories using a plurality of embedding layers. The computer system can determine a combined embedding based on the one or more quantitative embeddings and the

plurality of categorical embeddings. The computer system can generate a prediction based on the combined embedding for the input data.

[0006] In some embodiments, the computer system can perform a method of generating additional categorical data in a hierarchical manner. Prior to determining the plurality of categorical embeddings, the computer system can identify categorical data for a category based on a predetermined criteria. The computer system can generate one or more additional data graphs based on data derived from the categorical data for the category. The computer system can then include the one or more additional data graphs into the categorical data. The data derived from the category can be one or more new categories in the plurality of categories. In some embodiments, the predetermined criteria is that the categorical data for the category includes a sparse graph of data. The category that includes the sparse graph of data can be an IP address and the data derived from the category can include a first category of classless inter-domain routing and a second category of autonomous system number.

[0007] In some embodiments, the computer system can perform a method of generating time indexed quantitative embeddings. The one or more quantitative embeddings can be one or more time indexed quantitative embeddings. The computer system can determine the one or more indexed quantitative embeddings using the following steps. The computer system can obtain time data associated with the quantitative data from a data storage. The computer system can combine the quantitative data with time data. The computer system can determine one or more time indexed quantitative embeddings using the combination of the quantitative data with the time data using a machine learning model. In some embodiments, the computer system can then compress the one or more time indexed quantitative embeddings into a compressed quantitative embedding.

[0008] In some embodiments, the computer system can determine interactions between quantitative embeddings and categorical embeddings using a pair-wise dot product process. The computer system can determine the combined embedding using the following steps. The computer system can determine a dot product for each pair of each categorical embedding of the plurality of categorical embeddings and each quantitative embedding of the one or more quantitative embeddings. The computer system can combine each dot product to form an interacted embedding. The computer system can then generate the combined

embedding by concatenating the one or more quantitative embeddings and the interacted embedding.

[0009] In some embodiments, the computer system can determine interactions between quantitative embeddings and categorical embeddings using a transformer process.

- 5 The computer system can generate the combined embedding using a transformer. The one or more quantitative embeddings and the plurality of categorical embeddings can be utilized as inputs to the transformer.

[0010] In some embodiments, the computer system can determine interactions between quantitative embeddings and categorical embeddings using a concatenation process.

- 10 The computer system can concatenate each categorical embedding of the plurality of categorical embeddings and each quantitative embedding of the one or more quantitative embeddings to form the combined embedding.

[0011] Another embodiment is related to a machine learning computer comprising a processor and a computer-readable medium coupled to the processor. The computer-readable medium can comprise code executable by the processor for implementing a method. The method can include obtaining input data comprising quantitative data and categorical data. The categorical data can correspond to a plurality of categories. The method includes determining one or more quantitative embeddings for the quantitative data. The method includes determining a plurality of categorical embeddings for the plurality of categories using a plurality of embedding layers. The method also includes determining a combined embedding based on the one or more quantitative embeddings and the plurality of categorical embeddings. The method includes generating a prediction based on the combined embedding for the input data.

- 20 [0012] Further details regarding embodiments of the disclosure can be found in the Detailed Description and the Figures.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] FIG. 1 shows a block diagram of a system for neural networks for mixed data types according to embodiments.

- [0014] FIG. 2 shows a diagram illustrating an overall framework according to embodiments.
- [0015] FIG. 3 shows a block diagram of components of a machine learning computer according to embodiments.
- 5 [0016] FIG. 4 shows a flow diagram illustrating a method of generating a prediction using mixed data types according to embodiments.
- [0017] FIG. 5 shows a diagram illustrating hierarchical data enrichment according to embodiments.
- [0018] FIG. 6 shows a flow diagram illustrating a method of additional categorical
10 data in a hierarchical manner according to embodiments.
- [0019] FIG. 7 shows a diagram illustrating a time indexed quantitative embeddings creation method according to embodiments.
- [0020] FIG. 8 shows a flow diagram illustrating a method of generating time indexed quantitative embeddings according to embodiments.
- 15 [0021] FIG. 9 shows a diagram illustrating a transformer according to embodiments.
- [0022] FIG. 10 shows a diagram illustrating a convolutional neural network according to embodiments.
- [0023] FIG. 11 shows a diagram illustrating a computer system according to embodiments.

20

TERMS

- [0024] Prior to discussing embodiments of the disclosure, some terms can be described in further detail.
- [0025] A “machine learning computer” can include a device that creates, trains, and/or otherwise manipulates models. A machine learning computer can train a machine
25 learning model.

[0026] A “machine learning model” (ML model) can refer to a software module configured to be run on one or more processors to provide a classification or numerical value of a property of one or more samples. An ML model can include various parameters (e.g., for coefficients, weights, thresholds, functional properties of function, such as activation functions). As examples, an ML model can include at least 10, 100, 1,000, 5,000, 10,000, 50,000, 100,000, or one million parameters. An ML model can be generated using sample data (e.g., training samples) to make predictions on test data. Various number of training samples can be used, e.g., at least 10, 100, 1,000, 5,000, 10,000, 50,000, 100,000, or at least 200,000 training samples. One example is an unsupervised learning model such as hidden Markov model (HMM), clustering (e.g., hierarchical clustering, k-means, mixture models, model-based clustering, density-based spatial clustering of applications with noise (DBSCAN), and OPTICS algorithm), approaches for learning latent variable models such as Expectation–maximization algorithm (EM), method of moments, and blind signal separation techniques (e.g., principal component analysis, independent component analysis, non-negative matrix factorization, singular value decomposition), and anomaly detection (e.g., local outlier factor and isolation forest). Another example type of model is supervised learning that can be used with embodiments of the present disclosure. Example supervised learning models may include different approaches and algorithms including analytical learning, statistical models, artificial neural network (e.g. including convolutional and/or transformer layers) that may have 1-10 layers as examples, recurrent neural network (e.g., long short term memory, LSTM), boosting (meta-algorithm), bootstrap aggregating (bagging) such as random forests, support vector machine (SVM), support vector (SVR), Bayesian statistics, case-based reasoning, decision tree learning, inductive logic programming, linear regression, logistic regression, Gaussian process regression, genetic programming, group method of data handling, kernel estimators, learning automata, learning classifier systems, minimum message length (decision trees, decision graphs, etc.), multilinear subspace learning, naive Bayes classifier, maximum entropy classifier, conditional random field, nearest neighbor algorithm, probably approximately correct learning (PAC) learning, ripple down rules, a knowledge acquisition methodology, symbolic machine learning algorithms, subsymbolic machine learning algorithms, minimum complexity machines (MCM), ordinal classification, data pre-processing, handling imbalanced datasets, statistical relational learning, or Proaftn (a multicriteria classification algorithm), or an ensemble of any of these

types. Supervised learning models can be trained in various ways using various cost/loss functions that define the error from the known label (e.g., least squares and absolute difference from known classification) and various optimization techniques, e.g., using backpropagation, steepest descent, conjugate gradient, and Newton and quasi-Newton techniques.

[0027] A “deep neural network (DNN)” may be a neural network in which there are multiple layers between an input and an output. Each layer of the deep neural network may represent a mathematical manipulation used to turn the input into the output. In particular, a “recurrent neural network (RNN)” may be a deep neural network in which data can move forward and backward between layers of the neural network.

[0028] A “model database” may include a database that can store machine learning models. Machine learning models can be stored in a model database in a variety of forms, such as collections of parameters or other values defining the machine learning model. Models in a model database may be stored in association with keywords that communicate some aspect of the model. For example, a model used to evaluate news articles may be stored in a model database in association with the keywords “news,” “propaganda,” and “information.” A machine learning computer can access a model database and retrieve models from the model database, modify models in the model database, delete models from the model database, or add new models to the model database.

[0029] A “feature vector” may include a set of measurable properties (or “features”) that represent some object or entity. A feature vector can include collections of data represented digitally in an array or vector structure. A feature vector can also include collections of data that can be represented as a mathematical vector, on which vector operations such as the scalar product can be performed. A feature vector can be determined or generated from input data. A feature vector can be used as the input to a machine learning model, such that the machine learning model produces some output or classification. The construction of a feature vector can be accomplished in a variety of ways, based on the nature of the input data. For example, for a machine learning classifier that classifies words as correctly spelled or incorrectly spelled, a feature vector corresponding to a word such as “LOVE” could be represented as the vector (12, 15, 22, 5), corresponding to the alphabetical index of each letter in the input data word. For a more complex “input,” such as a human

entity, an exemplary feature vector could include features such as the human's age, height, weight, a quantitative representation of relative happiness, etc. Feature vectors can be represented and stored electronically in a feature store. Further, a feature vector can be normalized (i.e., be made to have unit magnitude). As an example, the feature vector (12, 15, 22, 5) corresponding to "LOVE" could be normalized to approximately (0.40, 0.51, 0.74, 0.17).

[0030] "Quantitative data" can include data related to a quantity of something. Quantitative data can include continuous data. Quantitative data can include a numerical value. Quantitative data can include heights, weights, sizes, amounts, volumes, distances, charges, etc. For example, quantitative data for transaction data can include transaction amounts, accumulated transaction volume, etc.

[0031] "Categorical data" can include data related to classifications or divisions of things having particular shard characteristics. Categorical data can qualitative data. Categorical data can include data represented by numerical values (e.g., categories represented by integer values). Categorical data can include types, subtypes, addresses, colors, shapes, identifiers, etc. For example, categorical data for transaction data can include card types, devices, email domains, IP addresses, shipping addresses, resource provider identifiers, transport computer identifiers, network processing identifiers, authorizing entity identifiers, etc.

[0032] An "embedding" can include a value that represents an object. An embedding can represent an object such as quantitative data, categorical data, text, images, audio, etc. An embedding can be in a particular dimensional space. An embedding can represent semantically meaningful information of the originating object. An embedding can be numerical value such as a vector.

[0033] An "embedding layer" can include a part of machine learning model that allows for processing and conversion of data into a different dimensional space. An embedding layer can transform input data into vectors of fixed size. An embedding layer can generate embeddings based on input data.

[0034] A "prediction" can include a generated estimate of something. A prediction can be a machine learning prediction that is the output generated by a trained machine

learning model. A prediction can be a predicted classification of something. A prediction can be derived by a machine learning model based on analyzed patterns and relationships within a dataset that the machine learning model has been trained on.

[0035] “Access request” can include a request to access a resource. An access request
5 can include a request from a first device or a first entity to access a resource that is provided by a second device or a second entity. For example, an access request can be a request generated by a user device that is requesting access to a resource provided by a resource provider computer. An access request may include authorization information, such as a user name, account number, password, credential, etc. The access request may also include an
10 access request identifier, a resource identifier, a timestamp, a date, a device or computer identifier, a geo-location, or any other suitable information. Example access requests include a transaction between two parties and a data exchange between two devices. In some embodiments, an access request can include a user requesting access to secure data, a secure webpage, a secure location, a resource, a service, etc. In other embodiments, an access
15 request can include a payment transaction in which two devices can interact to facilitate a payment. Access request data can include data related to and/or recorded during an access request. In some embodiments, access request data can be transaction data that includes a primary account number, a user device identifier, an amount (e.g., a transaction amount), a resource provider computer identifier, a transport computer identifier, a network processing
20 computer identifier, an authorizing entity computer identifier, a resource identifier, an IP address, a mailing address, a billing address, and/or other information related to the devices involved, the steps performed, or the interaction itself.

[0036] A “user” may include an individual. In some embodiments, a user may be associated with one or more personal accounts and/or mobile devices. The user may also be
25 referred to as a cardholder, account holder, or consumer in some embodiments.

[0037] A “user device” may be a device that is operated by a user. Examples of user devices may include a mobile phone, a smart phone, a card, a personal digital assistant (PDA), a laptop computer, a desktop computer, a server computer, a vehicle such as an automobile, a thin-client device, a tablet PC, etc. Additionally, user devices may be any type
30 of wearable technology device, such as a watch, earpiece, glasses, etc. The user device may include one or more processors capable of processing user input. The user device may also

include one or more input sensors for receiving user input. As is known in the art, there are a variety of input sensors capable of detecting user input, such as accelerometers, cameras, microphones, etc. The user input obtained by the input sensors may be from a variety of data input types, including, but not limited to, audio data, visual data, or biometric data. The user
5 device may comprise any electronic device that may be operated by a user, which may also provide remote communication capabilities to a network. Examples of remote communication capabilities include using a mobile phone (wireless) network, wireless data network (e.g., 3G, 4G, 5G, or similar networks), Wi-Fi, Wi-Max, or any other communication medium that may provide access to a network such as the Internet or a
10 private network.

[0038] A “user identifier” can include any piece of data that can identify a user. A user identifier can comprise any suitable alphanumeric string of characters. In some embodiments, the user identifier may be derived from user identifying information. In some embodiments, a user identifier can include an account identifier associated with the user.

15 [0039] An “access device” may be any suitable device that provides access to a remote system. An access device may also be used for communicating with a coordination computer, a communication network, or any other suitable system. An access device may generally be located in any suitable location, such as at the location of a merchant. An access device may be in any suitable form. Some examples of access devices include POS or point
20 of sale devices (e.g., POS terminals), cellular phones, personal digital assistants (PDAs), personal computers (PCs), tablet PCs, hand-held specialized readers, set-top boxes, electronic cash registers (ECRs), vending machines, automated teller machines (ATMs), virtual cash registers (VCRs), kiosks, security systems, access systems, and the like.

[0040] An access device may use any suitable contact or contactless mode of
25 operation to send or receive data from, or associated with, a mobile communication or payment device. For example, access devices can have card readers that can include electrical contacts, radio frequency (RF) antennas, optical scanners, bar code readers, or magnetic stripe readers to interact with portable devices such as payment cards.

[0041] The term “resource” generally refers to any asset that may be used or
30 consumed. For example, the resource may be an electronic resource (e.g., stored data,

received data, a computer account, a network-based account, an email inbox), a physical resource (e.g., a tangible object, a building, a safe, or a physical location), or other electronic communications between computers (e.g., a communication signal corresponding to an account for performing a transaction).

5 **[0042]** A “resource provider” may be an entity that can provide a resource such as goods, services, information, and/or access. Examples of resource providers includes merchants, data providers, transit agencies, governmental entities, venue and dwelling operators, etc.

10 **[0043]** An “authorization request message” may be an electronic message that requests authorization for an interaction. In some embodiments, it is sent to a transaction processing computer and/or an issuer of a payment card to request authorization for a transaction. An authorization request message according to some embodiments may comply with International Organization for Standardization (ISO) 8583, which is a standard for systems that exchange electronic transaction information associated with a payment made by
15 a user using a payment device or payment account. The authorization request message may include an issuer account identifier that may be associated with a payment device or payment account. An authorization request message may also comprise additional data elements corresponding to “identification information” including, by way of example only: a service code, a CVV (card verification value), a dCVV (dynamic card verification value), a PAN
20 (primary account number or “account number”), a payment token, a user name, an expiration date, etc. An authorization request message may also comprise “transaction information,” such as any information associated with a current transaction, such as the transaction value, merchant identifier, merchant location, acquirer bank identification number (BIN), card acceptor ID, information identifying items being purchased, etc., as well as any other
25 information that may be utilized in determining whether to identify and/or authorize a transaction.

30 **[0044]** An “authorization response message” may be a message that responds to an authorization request. In some cases, it may be an electronic message reply to an authorization request message generated by an issuing financial institution or a transaction processing computer. The authorization response message may include, by way of example only, one or more of the following status indicators: Approval -- transaction was approved;

Decline -- transaction was not approved; or Call Center -- response pending more information, resource provider is to call the toll-free authorization phone number. The authorization response message may also include an authorization code, which may be a code that a credit card issuing bank returns in response to an authorization request message in an electronic message (either directly or through the transaction processing computer) to the merchant's access device (e.g., POS equipment) that indicates approval of the transaction. The code may serve as proof of authorization.

[0045] An "authorizing entity" may be an entity that authorizes a request. Examples of an authorizing entity may be an issuer, a governmental agency, a document repository, an access administrator, etc. An authorizing entity may operate an authorizing entity computer. An "issuer" may refer to a business entity (e.g., a bank) that issues and optionally maintains an account for a user. An issuer may also issue payment credentials stored on a user device, such as a cellular telephone, smart card, tablet, or laptop to the consumer, or in some embodiments, a portable device.

15 [0046] A "processor" may include a device that processes something. In some embodiments, a processor can include any suitable data computation device or devices. A processor may comprise one or more microprocessors working together to accomplish a desired function. The processor may include a CPU comprising at least one high-speed data processor adequate to execute program components for executing user and/or system-generated requests. The CPU may be a microprocessor such as AMD's Athlon, Duron and/or Opteron; IBM and/or Motorola's PowerPC; IBM's and Sony's Cell processor; Intel's Celeron, Itanium, Pentium, Xeon, and/or XScale; and/or the like processor(s).

25 [0047] A "memory" may be any suitable device or devices that can store electronic data. A suitable memory may comprise a non-transitory computer readable medium that stores instructions that can be executed by a processor to implement a desired method. Examples of memories may comprise one or more memory chips, disk drives, etc. Such memories may operate using any suitable electrical, optical, and/or magnetic mode of operation.

30 [0048] A "server computer" may include a powerful computer or cluster of computers. For example, the server computer can be a large mainframe, a minicomputer

cluster, or a group of servers functioning as a unit. In one example, the server computer may be a database server coupled to a Web server. The server computer may comprise one or more computational apparatuses and may use any of a variety of computing structures, arrangements, and compilations for servicing the requests from one or more client computers.

5

DETAILED DESCRIPTION

[0049] Embodiments solve the technical problem of how to effectively extract information from the quantitative features, categorical features, as well as their interactions to obtain improved prediction performance.

[0050] Various embodiments can provide for a complete solution for training and
10 maintaining a prediction model. A framework, provided by embodiment, includes feature engineering, data loading, and model building. In further detail, in the feature engineering stage, embodiments define several novel and informative categorical features to enrich the data, and embodiments provide for a data loading pipeline to handle large-scale real-world transaction data. Moreover, embodiments provide for a network structure to further extract
15 knowledge for prediction from the features. Embodiments can 1) enrich data (e.g., transaction data, etc.) with additional informative categorical features, 2) have an ability to handle large-scale real-world transaction data, and 3) extract representative interaction knowledge from the data to make accurate predictions with the resulting machine learning model.

[0051] Embodiments can enrich data using with additional informative categorical
20 features. A machine learning computer can enrich the categorical data by evaluating the categorical data at different hierarchical levels. The machine learning computer can determine a plurality of hierarchical categories for each category in the categorical data. The machine learning computer can utilize a hierarchical structure of a particular category to construct additional categories of data. For example, the categorical data can include a
25 category of phone number. The machine learning computer can generate an additional category of categorical data using area codes from the phone numbers. The area codes can be in a hierarchical structure with the phone numbers. For example, machine learning computer can generate one or more additional data graphs (e.g., an area code data graph) based on data derived from the categorical data (e.g., area code) for the category (e.g., phone number). The

machine learning computer can include the one or more additional data graphs into the categorical data.

[0052] Embodiments can efficiently introduce and combine time data into quantitative data. The machine learning computer can obtain time data that is associated with
5 the quantitative data during a data preprocessing phase. The machine learning computer can combine the quantitative data with the time data. For example, the machine learning computer can combine the quantitative data and the time data using a first machine learning model or a concatenation process. The machine learning model can determine time indexed quantitative embeddings using the combination of the quantitative data with the time data
10 using a second machine learning model.

[0053] Embodiments can extract information from the interactions between quantitative data and categorical data. The machine learning computer can evaluate how quantitative embeddings derived from the quantitative data interact with categorical embeddings derived from the categorical data. The machine learning computer can utilize
15 one or more interaction determination methods. These methods can include a pair-wise dot product process, a concatenation process, a transformer process, and a convolutional neural network process. Each method can allow the machine learning computer to evaluate how the embeddings affect one another.

I. SYSTEM HARDWARE

20 [0054] Embodiments can utilize the systems described herein to, at least, create, maintain, and use neural networks for mixed data types. Systems can evaluate input data of mixed data types comprising quantitative data and categorical data to determine predictions that relate to the input data. For example, the input data can include data from an access request involving a user of a user device requesting access to a resource from a resource
25 provider of a resource provider computer. Systems described herein can evaluate quantitative data and categorical data from the access request to determine a prediction that is a classification of “authorized” or “not authorized.” The prediction can indicate whether or not the user is authorized to access the resource, e.g., to prevent cyberattacks.

A. Example network architecture

[0055] Embodiments provide for systems that can determine predictions using a neural network architecture based on mixed data types. The mixed data types can include quantitative data (e.g., amounts, transaction volumes, etc.) and categorical data (e.g.,
5 identifiers, account numbers, IP addresses, phone numbers, etc.). A machine learning computer can accurately generate predictions, such as “authorized” or “not authorized” from access request data captured from other devices an access request network. The access request network can allow a user device to request access to one or more resources from resource provider computers. During the access request, access request data comprising
10 quantitative data and categorical data can be provided to the machine learning computer to determine the prediction for the access request.

[0056] FIG. 1 shows a system 100 according to embodiments of the disclosure. The system 100 comprises a machine learning computer 102, a data storage 104, a client device 106, a user device 108, an access device 110, a resource provider computer 112, a transport
15 computer 114, a network processing computer 116, an authorizing entity computer 118, and a resource 120. The system 100 can be an access request network.

[0057] The machine learning computer 102 can be in operative communication with the data storage 104 and the client device 106. The data storage 104 can be in operative communication with the network processing computer 116. The user device 108 can be in
20 operative communication with the access device 110 and the resource provider computer 112. The resource provider computer 112 can be in operative communication with the access device 110 and the transport computer 114. The network processing computer 116 can be in operative communication with the transport computer 114 and the authorizing entity computer 118. Access to a resource 120 (e.g., an electronic resource such as a computer or
25 account or a physical location such as a building) can be controlled by resource provider computer 112 or access device 110.

[0058] For simplicity of illustration, a certain number of components are shown in FIG. 1. It is understood, however, that embodiments of the invention may include more than one of each component. In addition, some embodiments of the invention may include fewer
30 than or greater than all of the components shown in FIG. 1.

[0059] Messages between the devices illustrated in FIG. 1 can be transmitted using a secure communications protocols such as, but not limited to, File Transfer Protocol (FTP); HyperText Transfer Protocol (HTTP); Secure Hypertext Transfer Protocol (HTTPS), SSL, ISO (e.g., ISO 8583) and/or the like. The communications network may include any one
5 and/or the combination of the following: a direct interconnection; the Internet; a Local Area Network (LAN); a Metropolitan Area Network (MAN); an Operating Missions as Nodes on the Internet (OMNI); a secured custom connection; a Wide Area Network (WAN); a wireless network (e.g., employing protocols such as, but not limited to a Wireless Application Protocol (WAP), I-mode, and/or the like); and/or the like. The communications network can
10 use any suitable communications protocol to generate one or more secure communication channels. A communications channel may, in some instances, comprise a secure communication channel, which may be established in any known manner, such as through the use of mutual authentication and a session key, and establishment of a Secure Socket Layer (SSL) session.

15 [0060] The machine learning computer 102 can include a computer that can train, update, and utilize machine learning models. The machine learning computer 102 can obtain data from the data storage 104 for use in training one or more machine learning models. After training a machine learning model, as described in further detail herein, the machine learning computer 102 can provide predictions generated by the machine learning model or the
20 machine learning model itself to the client device 106.

[0061] The data storage 104 can store quantitative data and categorical data. The data storage 104 can store, for example, transaction data that includes both quantitative data and categorical data. The data storage 104 can include any suitable database. The data storage 104 may be a conventional, fault tolerant, relational, scalable, secure database such as those
25 commercially available from Oracle™ or Sybase™.

[0062] The client device 106 can include a computer. The client device 106 can obtain predictions from the machine learning computer 102. In some embodiments, the client device 106 can generate a prediction request message comprising a data identifier. The data identifier can be a transaction identifier that identifies a particular transaction associated with
30 quantitative data and categorical data.

- [0063] The client device 106 can provide the prediction request message to the machine learning computer 102. The machine learning computer 102 can obtain input data comprising the quantitative data and the categorical data as indicated by the data identifier from the data storage 104.
- 5 [0064] The machine learning computer 102 can utilize the trained machine learning model and the input data to determine a prediction (e.g., a predicted classification of authorized or not authorized). The machine learning computer 102 can generate a prediction response message comprising the prediction. The machine learning computer 102 can provide the prediction response message to the client device 106 in response to the prediction request
10 message.
- [0065] The user device 108 can include one or more computers, portable computers, laptop computers, tablet computers, mobile devices, cellular phones, wearable devices (e.g., watches, glasses, lenses, clothing, etc.), personal digital assistants (PDAs), Internet of Things (IoT) devices, and/or the like. The user device 108 can initiate access requests (e.g.,
15 transactions, secure location access requests, cybersecurity requests, etc.) with resource provider computers and/or access devices to request access to resources such as the resource 120. For example, the user device 108 can select one or more resources for the access request at a resource provider location (e.g., a grocery store, a transit terminal, a secure location access terminal). In some embodiments, the user device 108 can communicate with the
20 resource provider computer 112 directly. In other embodiments, the user device 108 can communicate with the resource provider computer 112 via the access device 110.
- [0066] The resource 120 can include an asset that may be used or consumed. The user device 108 can request access to the resource 120 from the resource provider computer 112. The resource 120 may be an electronic resource (e.g., stored data, received data, a computer
25 account, a network-based account, an email inbox, etc.), a physical resource (e.g., a tangible object, a building, a safe, a physical location, etc.), or other electronic communications between computers (e.g., a communication signal corresponding to an account for performing a transaction, etc.).
- [0067] The access device 110 can include a device operated by a resource provider.
30 The access device 110, for example, can include a mobile device, a POS terminal, a laptop,

etc. The access device 110 can communicate with another device (e.g., a user device 108) to perform an access request. During the access request, the access device 110 can receive credentials from the user device and can provide interaction data to the resource provider computer 112 for authorization of the interaction. In some embodiments, the access device
5 110 can generate an authorization request message comprising at least the access request data. The access device 110 can provide the authorization request message to the resource provider computer 112.

[0068] The resource provider computer 112 can include any suitable computational apparatus operated by a resource provider. In some embodiments, the resource provider
10 computer 112 may include one or more server computers that may host one or more websites associated with the resource provider (e.g., a merchant). In some embodiments, the resource provider computer 112 may be configured to send data to the network processing computer 116 via the transport computer 114 as part of a payment verification and/or authentication process for a transaction between the user (e.g., a consumer) and the resource provider. The
15 resource provider computer 112 may also be configured to generate authorization request messages for transactions between a resource provider and a user, and route the authorization request messages to the authorizing entity computer 118 for processing.

[0069] The transport computer 114 can include a server computer. In some embodiments, the transport computer 114 may be associated with an acquirer, which may be
20 an entity that has a business relationship with a particular resource provider or other entity. Some computers can perform both issuer and acquirer functions. Some embodiments may encompass such single entity issuer-acquirer computers.

[0070] The network processing computer 116 can include a server computer. The network processing computer 116 may be disposed between the transport computer 114 and
25 the authorizing entity computer 118. The network processing computer 116 may include data processing subsystems, networks, and operations used to support and deliver authorization services, exception file services, and clearing and settlement services. For example, the network processing computer 116 may comprise a server coupled to a network interface (e.g., by an external communication interface), and databases of information. The network
30 processing computer 116 may be representative of a transaction processing network. An exemplary transaction processing network may include VisaNet™. Transaction processing

networks such as VisaNet™ are able to process credit card transactions, debit card transactions, and other types of commercial transactions. VisaNet™, in particular, includes a VIP system (Visa Integrated Payments system) which processes authorization requests and a Base II system which performs clearing and settlement services. The network processing
5 computer 116 may use any suitable wired or wireless network, including the Internet.

[0071] The authorizing entity computer 118 can include a server computer operated by an authorizing entity. The authorizing entity computer 118 may be associated with an authorizing entity, which may be an entity that authorizes a request. An example of an authorizing entity may be an issuer. The authorizing entity computer 118 can maintain an
10 account for a user. An issuer may also issue and manage an account associated with the user device 108.

[0072] As an illustrative example, a user of the user device 108 can conduct an access request. For example, the user device 108 can conduct a transaction with the resource provider computer 112. The access request may be a payment transaction (e.g., for the
15 purchase of a good or service), an access transaction (e.g., for access to a transit system), or any other suitable transaction. The user device 108 can interact with the access device 110 at a resource provider location and is associated with the resource provider computer 112. For example, the user may tap the user device 108 against an NFC reader in the access device 110. Alternately, the user device 108 may indicate payment account information to the
20 resource provider computer 112 electronically, such as in an online transaction. The user device 108 may transmit access request data to the resource provider computer 112.

[0073] In order to authorize an access request, an authorization request message may be generated by the access device 110 or by the resource provider computer 112 and then forwarded to the transport computer 114. After receiving the authorization request message,
25 the authorization request message is then sent to the network processing computer 116.

[0074] The network processing computer 116 can store data related to the access request in the data storage 104. The network processing computer 116 can store access request data comprising a user device identifier, a resource provider identifier, a transport computer identifier, a network processing computer identifier, an authorizing entity computer

identifier, a primary account number (PAN), a transaction amount, a resource identifier, a date, a time, a geographic location, and/or any other data related to the access request.

[0075] In some embodiments, prior the network processing computer 116 providing the authorization request message to the authorizing entity computer 118, the machine learning computer 102 can obtain the access request data from the data storage 104. The machine learning computer 102 can use quantitative data and categorical data in the access request data to generate a prediction related to the access request. For example, the machine learning computer 102 can generate a prediction of “authorized” or “not authorized.” As another example, the machine learning computer 102 can generate a prediction of “fraudulent” or “not fraudulent.” As other examples, the prediction can indicate whether a user is likely to cancel a service or a transaction based on historical access requests, e.g., historical transactions.

[0076] The machine learning computer 102 can provide the prediction to the network processing computer 116. The network processing computer 116 can utilize the prediction to determine whether or not to deny the authorization request. If the network processing computer 116 determines to deny the authorization request based on the prediction, then the network processing computer 116 can generate an authorization response message comprising an indication that indicates that the access request is not authorized. If the network processing computer 116 determines to approve or otherwise proceed with the access request, then the network processing computer can proceed with the process below.

[0077] The network processing computer 116 can then forward the authorization request message to the corresponding authorizing entity computer 118 associated with an authorizing entity associated with the user. For example, the authorizing entity computer 118 can maintain a user account on behalf of the user of the user device 108.

[0078] After receiving the authorization request message, the authorizing entity computer 118 can determine whether or not to authorize the access request. The authorizing entity computer 118 can generate an indication of whether or not the access request is authorized. The authorizing entity computer 118 can generate an authorization response message comprising the indication of whether or not the access request is authorized.

[0079] The authorizing entity computer 118 can provide the authorization response message to the network processing computer 116 to indicate whether or not the current access request is authorized. In some embodiments, the network processing computer 116 can store information related to the authorization response message into the data storage 104 along
5 with the previously stored information. For example, the network processing computer 116 can store the indication of whether or not the current access request is authorized.

[0080] In some embodiments, the machine learning computer 102 can process the access request data to generate a prediction after the network processing computer 116 receives the authorization response message. In some cases, the network processing computer
10 116 can deny the access request based on the prediction even though the access request is authorized by the authorizing entity computer 118.

[0081] The network processing computer 116 can provide the authorization response message back to the transport computer 114. In some embodiments, network processing computer 116 may decline the access request even if the authorizing entity computer 116 has
15 authorized the access request, for example depending on a value of the fraud risk score or other criteria.

[0082] After receiving the authorization response message, the transport computer 114 can provide the authorization response message to the resource provider computer 112.

[0083] After the resource provider computer 112 receives the authorization response message, the resource provider computer 112 can provide the authorization response message
20 to the user of the user device 108. The authorization response message may be displayed by the access device 110 to the user, or may be printed out on a physical receipt. Alternately, if the access request is an online access request, the resource provider computer 112 may provide a web page or other indication of the authorization response message as a virtual
25 receipt to the user device 108.

[0084] If the authorization response message indicates that the access request is authorized, then the user device 108 and/or the user operating the user device 108 can access the resource 120. In some embodiments, the resource provider computer 112 can provide the resource 120 to the user device 108.

[0085] Over time, more access request data is stored into the data storage 104. The machine learning computer 102 can obtain the access request data from the data storage 104 to train the machine learning model used to generate the prediction. The machine learning computer 102 can develop the machine learning model over time with additional access
5 request data.

II. COMBINING QUANTITATIVE AND CATEGORICAL EMBEDDINGS

[0086] This section describes generating predictions based on quantitative data and categorical data from an access request. The machine learning computer 102 can process access request data comprising quantitative data and categorical data to generate predictions.
10 Predictions can be generated based on the access request data. The machine learning computer 102 can process the quantitative data, the categorical data, and interactions thereof to improve prediction accuracy.

A. *Processing quantitative data and categorical data*

[0087] The machine learning computer 102 can obtain quantitative data and categorical data. The machine learning computer 102 can generate quantitative embeddings from the quantitative data and can generate categorical embeddings from the categorical data. The machine learning computer 102 can then evaluate interactions between the quantitative embeddings and the categorical embeddings. The resulting embeddings can be used to generate a prediction relating to the access request.

20 [0088] FIG. 2 shows a diagram illustrating an overall framework according to embodiments. FIG. 2 illustrates a machine learning model using both quantitative data and categorical data as inputs. The machine learning computer 102 can train the machine learning model over a plurality of iterations. The machine learning computer 102 can maintain and utilize the machine learning model.

25 [0089] FIG. 2 includes four different layers that can be performed in the machine learning model, however, it is understood that more layers may exist. FIG. 2 includes an input layer 210, a bottom layer 220, an interaction layer 230, and a top layer 240.

[0090] The input layer 210 includes quantitative data 212 and categorical data 214. The machine learning computer 102 can obtain the quantitative data 212 and the categorical

data 214 from the data storage 104. The quantitative data 212 and the categorical data 214 can be stored in association with one another. The quantitative data 212 and the categorical data 214 can include data that relates to an instance of interaction data (e.g., transaction data).

[0091] The machine learning computer 102 can utilize both the quantitative data 212
5 and the categorical data 214 to train the machine learning model. The quantitative data 212 can include, for example, a transaction amount and an accumulated transaction volume. The categorical data 214 can include, for example, a card type, a device, an email domain, an IP address, a shipping address, and a resource provider identifier. The categorical data 214 can correspond to a plurality of categories. As an example, the categorical data 214 can include
10 four categories: C1, C2, Ci, and CM. The categorical data 214 can include any number of categories. For example, the categorical data 214 can include at least 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000 categories, etc.

[0092] In some embodiments, the input layer 210 can also include time data 216 (e.g., a current time, a time of data capture, etc.). Inclusion of time data into the input layer 210 will
15 be further described below.

[0093] The input data can be data that corresponds to a current transaction that is being processed by the system to determine a prediction (e.g., an output classification). The prediction can indicate a classification of the current transaction. For example, the prediction can indicate that the current transaction is classified as “fraud” or “not fraud.” As another
20 example, the predication can indicate that the current transaction is classified as “accept” or “do not accept.”

[0094] In some embodiments, the machine learning computer 102 can perform feature engineering to select, manipulate, and transform raw data into features that can be used in the machine learning process in the input layer 210. For example, the machine
25 learning computer 102 can perform a subjective workload assessment technique (SWAT), reduce sparsity in IP networks (as described in reference to FIG. 4), create embeddings for multi-valued sparse features, compress high dimension sparse features, evaluate a modified Cox-box feature transformation for outliers (e.g., utilize the Cox-box method with $\lambda = 0$ and add a learned parameter to scale the variable to reduce the impact of strong outliers),

perform quantitative feature embedding, utilize pre-trained graph embedding, and/or any other suitable feature engineering process.

[0095] The bottom layer 220 includes an embedding layer that can embed the quantitative data 212 and the categorical data 214 from the input layer 210. The bottom layer
5 220 can perform embedding on the quantitative data 212 or a combination of the quantitative data 212 and the time data 216 in a time indexed quantitative embedding process as described in further detail in reference to FIG. 7.

[0096] The quantitative data 212 can be embedded using an embedding machine learning model 222 to obtain a quantitative embedding 226. The embedding machine learning
10 model 222 can take quantitative data as input and can output the quantitative embedding 226. The quantitative embedding 226 can represent the quantitative data 212. The embedding machine learning model 222 can include a multilayer perception, a recurrent neural network, a convolutional neural network, or a transformer. An example transformer is described in further detail in reference to FIG. 9. An example convolutional neural network is described in
15 further detail in reference to FIG. 10.

[0097] In some embodiments, the quantitative embedding 226 can include one or more quantitative embeddings that relate to a particular quantitative data type in the quantitative data 212. For example, there may be 100 elements of a quantitative data type of access request volume over time, which can yield one compressed quantitative embedding or
20 100 different quantitative embeddings, as described in further detail in reference to FIG. 7.

[0098] The categorical data 214 can be embedded using a plurality of embedding layers 224 to obtain a plurality of categorical embeddings 228. Each embedding layer of the plurality of embedding layers 224 can embed a different category of the categorical data 214. The plurality of embedding layers 224 includes a C1 embedding layer, a C2 embedding layer,
25 a Ci embedding layer, and a CM embedding layer. Each embedding layer can correspond to a particular category of the categorical data 214. The C1 embedding layer can embed data from the category C1 of the categorical data 214. The C2 embedding layer can embed data from the category C2 of the categorical data 214. The Ci embedding layer can embed data from the category Ci of the categorical data 214. The CM embedding layer can embed data from the
30 category CM of the categorical data 214.

[0099] An embedding layer can include a machine learning model layer. An embedding layer can be a hidden layer in a neural network. An embedding layer can map input data from a first dimensional space (e.g., a high-dimensional space) to a second dimensional space (e.g., a lower-dimensional space). An embedding layer can allow machine learning computer 102 to learn more about the relationship between inputs.

[0100] In some embodiments, the embedding layers can be text embedding layers, image embedding layers, graph embedding layers, or other types of embedding layers depending on the category of the categorical data.

[0101] In some cases, the categorical data 214 can be represented as integer values, where the integer indicates a particular option of the category. For example, a category of card type can be represented by integer values, where a value of 1 indicates a credit card and a value of 2 indicates a debit card.

[0102] During training, the embedding layer can learn which inputs (e.g., categorical data from a particular category) are similar and can adjust the embedding vectors accordingly. The learning process can occur through backpropagation. For example, when the machine learning model makes a prediction that is not 100% accurate, as compared to a known value, the machine learning model can calculate how wrong the prediction was and can adjust the weights within the embedding layer. Adjustment of the weights can be performed using an optimization algorithm, such as gradient descent, which modifies the weight values in the embedding layer to minimize the error, or the loss function.

[0103] Each categorical embedding layer can output a categorical embedding that represents the input categorical data. The plurality of categorical embeddings 228 can include four categorical embeddings that are output from the four embedding layers of the plurality of embedding layers 224. For example, the C1 embedding layer can output the C1 categorical embedding, the C2 embedding layer can output the C2 categorical embedding, the Ci categorical embedding layer can output the Ci categorical embedding, and the CM embedding layer can output the CM categorical embedding.

[0104] The categorical embeddings can be of any suitable dimension. For example, the categorical embeddings can have a dimension of 128 floating-point numbers. Each categorical embedding can be the same length.

[0105] As an example, using the embedding layer, the machine learning computer 102 can convert the integer indexes of the categorical data 214 into dense vector representations of each of the integer numbers. The machine learning computer 102 can embed an integer of 1 that represents a credit card for a category of card type into a value of length 64 dimensions, 128 dimensions, 256 dimensions, 512 dimensions, etc.

[0106] The machine learning computer 102 can create a data structure that includes a plurality of embeddings 229. The plurality of embeddings 229 can include all of the categorical embeddings of the plurality of categorical embeddings 228 and the quantitative embedding 226.

10 [0107] The interaction layer 230 is a layer that includes processing that can allow the machine learning computer 102 to determine interactions between different embeddings from different input data from the input layer 210. The machine learning computer 102 can determine the interaction between quantitative embeddings and categorical embeddings. The machine learning computer 102 can utilize one or more methods to determine the interaction.
15 These methods can include one or more of a pair-wise dot product process 231, a concatenation process 236, a transformer process 238, and a convolutional neural network process (not depicted).

[0108] The pair-wise dot product process 231 can include a process of creating an interaction matrix of dot products between the quantitative embeddings and categorical
20 embeddings included in the plurality of embeddings 229 to determine how each pair of embeddings interact with one another. The machine learning computer 102 can determine a pair-wise dot product 232 from the plurality of embeddings 229, such that each dot product is created for each pair of embeddings in the plurality of embeddings 229.

[0109] As an illustrative example, the machine learning computer 102 can create the
25 interaction matrix from the plurality of embeddings 229 that includes four categorical embeddings (C1, C2, Ci, and CM) and one quantitative embedding (N). The machine learning computer 102 can determine a dot product between each pair of embeddings. For example, for the first categorical embedding C1, the machine learning computer 102 can determine a dot product between C1 and C2, between C1 and Ci, between C1 and CM, and

between C1 and N. The machine learning computer 102 can create the interaction matrix from the plurality of embeddings 229 as follows:

$$\begin{bmatrix} - & C1 \cdot C2 & C1 \cdot Ci & C1 \cdot CM & C1 \cdot N \\ C2 \cdot C1 & - & C2 \cdot Ci & C2 \cdot CM & C2 \cdot N \\ Ci \cdot C1 & Ci \cdot C2 & - & Ci \cdot CM & Ci \cdot N \\ CM \cdot C1 & CM \cdot C2 & CM \cdot Ci & - & CM \cdot N \\ N \cdot C1 & N \cdot C2 & N \cdot Ci & N \cdot CM & - \end{bmatrix}$$

- [0110] After generating the interaction matrix, the machine learning computer 102 can create an interacted embedding 233 by flattening the interaction matrix to a single embedding vector referred to as an interacted embedding 233. Since the interaction matrix is symmetrical, the machine learning computer 102 can keep half of the interaction matrix while not losing any information. The machine learning computer 102 can keep either the upper triangle or the lower triangle of entries of the interaction matrix. For example, the machine learning computer 102 can remove the lower triangle of entries of the interaction matrix as follows:

$$\begin{bmatrix} - & C1 \cdot C2 & C1 \cdot Ci & C1 \cdot CM & C1 \cdot N \\ - & - & C2 \cdot Ci & C2 \cdot CM & C2 \cdot N \\ - & - & - & Ci \cdot CM & Ci \cdot N \\ - & - & - & - & CM \cdot N \\ - & - & - & - & - \end{bmatrix}$$

- [0111] The machine learning computer 102 can flatten the interaction matrix into the interacted embedding 233. The machine learning computer 102 can separate each row out of the interaction matrix and append each row to one another to form the interacted embedding 233. The machine learning computer 102 can remove any empty entries from the rows. For example, the machine learning computer 102 can append the second row of the interaction matrix to the end of the first row. The machine learning computer 102 can create the following interacted embedding 233 from the interaction matrix:

$$\begin{aligned} & (C1 \cdot C2) + (C1 \cdot Ci) + (C1 \cdot CM) + (C1 \cdot N) + (C2 \cdot Ci) + (C2 \cdot CM) + (C2 \cdot N) \\ & + (Ci \cdot CM) + (Ci \cdot N) + (CM \cdot N) \end{aligned}$$

- [0112] Each element in the interacted embedding 233 can be appended to one another, as indicated by the symbol +.

[0113] The machine learning computer 102 can then determine a combined embedding 234 based on the interacted embedding 233 and the quantitative embedding 226. The machine learning computer 102 can determine the combined embedding 234 by combining the interacted embedding 233 and the quantitative embedding 226 in any suitable manner. For example, the machine learning computer 102 can concatenate the interacted embedding 233 with the quantitative embedding 234 to form the combined embedding 234.

[0114] For example, the machine learning computer 102 can generate the combined embedding 234 as follows:

$$(N) + (C1 \cdot C2) + (C1 \cdot Ci) + (C1 \cdot CM) + (C1 \cdot N) + (C2 \cdot Ci) + (C2 \cdot CM) + (C2 \cdot N) + (Ci \cdot CM) + (Ci \cdot N) + (CM \cdot N)$$

[0115] The machine learning computer 102 can efficiently compress the data using the pair-wise dot product process 231. For example, 10 embedded features at 100 dimensions each (1,000 total float numbers), results in 45 dot-product values, a reduction of approximately 20 times of feature space for more efficient training and scoring.

[0116] The machine learning computer 102 can perform a concatenation process 236 to combine the plurality of embeddings 229. The concatenation process 236 can include concatenating the plurality of embeddings 229. The machine learning computer 102 can concatenate each embedding in the plurality of embeddings 229. For example, the machine learning computer 102 can concatenate the embeddings as follows, where the symbol + indicates a concatenation operation:

$$(N) + (C1) + (C2) + (Ci) + (CM)$$

[0117] The machine learning computer 102 can perform a transformer process 238 to combine the plurality of embeddings 229. The transformer process 238 can include the machine learning computer 102 using a transformer. The transformer can take the plurality of embeddings 228 and the quantitative embedding 226 as input to determine an overall embedding as output. The transformer in the transformer 238 can be structured similar to the transformer model 222. Transformers are described in further detail in reference to FIG. 9.

[0118] In some embodiments, the machine learning computer 102 can utilize a recurrent neural network or a convolutional neural network to combine the plurality of

embeddings 229. For example, the machine learning computer 102 can generate an overall embedding from the plurality of embeddings 229 using a convolutional neural network. Convolutional neural networks described in further detail in reference to FIG. 10.

5 [0119] After generating an embedding that represents a combination of the plurality of embeddings 229, the machine learning computer 102 can proceed to processing the top layer 240. The top layer 240 includes a machine learning model 242. The machine learning model 242 can be a multilayer perceptron, recurrent neural network, a convolutional neural network, or a transformer.

10 [0120] The top layer 240 can accept the embeddings from the interaction layer 230 as input and can determine a prediction 244 as output. The prediction 244 can be a classification or value (e.g., a probability value). The machine learning computer 102 can generate a prediction based on the combined embedding for the input data using the machine learning model 242. For example, the machine learning computer 102 can input the combined embedding that represents the one or more quantitative embeddings and the plurality of
15 categorical embeddings into the machine learning model 242 to generate the prediction 244.

[0121] The prediction 244 can be a predicted classification that relates to the input data. For example, if the input data is access request data, then the prediction 244 can be “fraud” or “not fraud.” As such, the machine learning computer 102 can determine whether or not to classify the access request as fraudulent or as not fraudulent.

20 *B. Machine learning computer*

[0122] The machine learning computer 102 can perform data preprocessing on the quantitative data and the categorical data. The machine learning computer 102 can also generate embeddings for the quantitative data and the categorical data, respectively resulting in quantitative embeddings and categorical embeddings. The machine learning computer 102
25 can also evaluate interactions between the quantitative embeddings and the categorical embeddings.

[0123] FIG. 3 shows a block diagram of the machine learning computer 102 according to embodiments. The exemplary machine learning computer 102 may comprise a processor 304. The processor 304 may be coupled to a memory 302, a network interface 306,

and a computer readable medium 308. The computer readable medium 308 can include a quantitative embedding module 308A, a categorical embedding module 308B, and an interaction module 308C.

[0124] The memory 302 can be used to store data and code. For example, the memory
5 302 can store data including quantitative features and categorical features, models, model weights, and/or any other data related to training, maintaining, and utilizing a machine learning model. The memory 302 may be coupled to the processor 304 internally or externally (e.g., cloud based data storage), and may comprise any combination of volatile and/or non-volatile memory, such as RAM, DRAM, ROM, flash, or any other suitable
10 memory device.

[0125] The computer readable medium 308 may comprise code, executable by the processor 304, for performing methods described herein. One such method can include obtaining input data comprising quantitative data and categorical data. The categorical data can correspond to a plurality of categories. One or more quantitative embeddings can be
15 determined for the quantitative data. A plurality of categorical embeddings can be determined for the plurality of categories using a plurality of embedding layers. A combined embedding can be determined based on the one or more quantitative embeddings and the plurality of categorical embeddings. A prediction can be generated based on the combined embedding for the input data.

20 [0126] The quantitative embedding module 308A can comprise code or software, executable by the processor 304, for embedding quantitative data. The quantitative embedding module 308A, in conjunction with the processor 304, can generate quantitative embeddings from quantitative data. The quantitative embedding module 308A, in conjunction with the processor 304, can include a machine learning model that can be trained to convert
25 quantitative data into quantitative embeddings. The machine learning model can include a multilayer perceptron, a recurrent neural network, a convolutional neural network, or a transformer. The quantitative embedding module 308A, in conjunction with the processor 304, can generate quantitative embeddings of a fixed size (e.g., 128 dimensions, 256 dimensions, 512 dimensions, 1024 dimensions, etc.).

[0127] The categorical embedding module 308B can comprise code or software, executable by the processor 304, for embedding categorical data. The categorical embedding module 308B, in conjunction with the processor 304, can generate categorical embeddings from categorical data. The categorical embedding module 308B, in conjunction with the processor 304, can include a plurality of embedding layers, where each embedding layer corresponds to a different category of the categorical data. Each embedding layer can include a machine learning model that can be trained to convert categorical data into categorical embeddings. The categorical embedding module 308B, in conjunction with the processor 304, can generate categorical embeddings of a fixed size, which can be the same size as the quantitative embeddings.

[0128] The interaction module 308A can comprise code or software, executable by the processor 304, for interacting embeddings with one another. The interaction module 308A, in conjunction with the processor 304, can combine quantitative embeddings and categorical embeddings.

[0129] In some embodiments, the interaction module 308A, in conjunction with the processor 304, can combine the quantitative embeddings and categorical embeddings using a pair-wise dot product process. The interaction module 308A, in conjunction with the processor 304, can determine a dot product value for each pair of embeddings. The interaction module 308A, in conjunction with the processor 304, can generate an interacted embedding based on all of the dot products.

[0130] In other embodiments, the interaction module 308A, in conjunction with the processor 304, can combine the quantitative embeddings and categorical embeddings using a concatenation process. The interaction module 308A, in conjunction with the processor 304, can concatenate each quantitative embedding and each categorical embedding with one another.

[0131] In other embodiments, the interaction module 308A, in conjunction with the processor 304, can combine the quantitative embeddings and categorical embeddings using a transformer process. The interaction module 308A, in conjunction with the processor 304, can utilize a transformer to generate an embedding that represents a combination of the quantitative embeddings and the categorical embeddings.

[0132] The network interface 306 may include an interface that can allow the machine learning computer 102 to communicate with external computers. The network interface 306 may enable the machine learning computer 102 to communicate data to and from another device (e.g., the data storage 104, the client device 106, etc.). Some examples of the network interface 306 may include a modem, a physical network interface (such as an Ethernet card or other Network Interface Card (NIC)), a virtual network interface, a communications port, a Personal Computer Memory Card International Association (PCMCIA) slot and card, or the like. The wireless protocols enabled by the network interface 306 may include Wi-Fi™. Data transferred via the network interface 306 may be in the form of signals which may be electrical, electromagnetic, optical, or any other signal capable of being received by the external communications interface (collectively referred to as “electronic signals” or “electronic messages”). These electronic messages that may comprise data or instructions may be provided between the network interface 306 and other devices via a communications path or channel. As noted above, any suitable communication path or channel may be used such as, for instance, a wire or cable, fiber optics, a telephone line, a cellular link, a radio frequency (RF) link, a WAN or LAN network, the Internet, or any other suitable medium.

C. *Prediction generation using quantitative data, categorical data, and interactions thereof*

[0133] The machine learning computer 102 can obtain quantitative data and categorical data for an access request. The machine learning computer 102 can evaluate the quantitative data, the categorical data, and the interactions thereof to determine a prediction relating to the access request.

[0134] FIG. 4 shows a flow diagram illustrating a method of generating a prediction using mixed data types according to embodiments. The method illustrated in FIG. 4 can be performed by a computer system such as the machine learning computer 102.

[0135] At step 402, the machine learning computer 102 can obtain input data from the data storage 104. The input data can comprise quantitative data and categorical data. The categorical data corresponds to a plurality of categories.

[0136] As an illustrative example, the machine learning computer 102 can obtain input data that corresponds to an access request. The access request can be a request to access a resource. For example, a user of a user device can request access to a resource from a resource provider of a resource provider computer. The input data can include quantitative data and categorical data from the access request. The quantitative data can include an amount (e.g., a transaction amount). The categorical data can include an account number (e.g., a primary account number), an IP address, and a card type.

[0137] At step 404, after obtaining the input data, the machine learning computer 102 can determine one or more quantitative embeddings for the quantitative data. The machine learning computer 102 can determine the one or more quantitative embeddings using a quantitative embedding learning machine learning model (e.g., the embedding learning machine learning model 222 of FIG. 2). The quantitative embedding learning machine learning model can be trained to convert quantitative data into quantitative embeddings. For example, the quantitative embedding learning machine learning model can be a multilayer perceptron.

[0138] As an illustrative example, the machine learning computer 102 can determine a quantitative embedding for the quantitative data of the amount. For example, the amount can be a value of \$99.99. The machine learning computer 102 can generate an amount quantitative embedding that represents the amount value of \$99.99. For example, the amount quantitative embedding can be a vector with 128 dimensions.

[0139] At step 406, the machine learning computer 102 can determine a plurality of categorical embeddings for the plurality of categories using a plurality of embedding layers. The machine learning computer 102 can determine the plurality of categorical embeddings based on the categorical data using the plurality of embedding layers. Each embedding layer can correspond to a different category of the categorical data.

[0140] As an illustrative example, the categorical data can include data in the categories of account number, IP address, and card type. The machine learning computer 102 can obtain three embedding layers for the categories. For example, the machine learning computer 102 can obtain an account number embedding layer, an IP address embedding

layer, and a card type embedding layer. The machine learning computer 102 can utilize each layer to respectively generate categorical embeddings for the data in each category.

[0141] The machine learning computer 102 can generate an account number categorical embedding from the account number using the account number embedding layer.
5 The account number can include a value of 1234567890123456. The machine learning computer 102 can generate an account number categorical embedding that is a vector with 128 dimensions.

[0142] The machine learning computer 102 can generate an IP address categorical embedding from the IP address using the IP address embedding layer. The IP address can
10 include a value of 192.158.1.38. The machine learning computer 102 can generate an IP address categorical embedding that is a vector with 128 dimensions.

[0143] The machine learning computer 102 can generate a card type categorical embedding from the card type using the card type embedding layer. The card type can include a value of 1 that indicates a credit card. The machine learning computer 102 can
15 generate a card type categorical embedding that is a vector with 128 dimensions.

[0144] At step 408, after determining the one or more quantitative embeddings and the plurality of categorical embeddings, the machine learning computer 102 can determine a combined embedding based on the one or more quantitative embeddings and the plurality of categorical embeddings. A plurality of embeddings can include the one or more quantitative
20 embeddings and the plurality of categorical embeddings.

[0145] The machine learning computer 102 can determine the combined embedding using a pair-wise dot product process, a concatenation process, a transformer process, or a convolutional neural network process.

[0146] In some embodiments, the machine learning computer 102 can combine the
25 plurality of embeddings using the pair-wise dot product process. For example, the machine learning computer 102 can determine a dot product from each pair of embeddings of the plurality of embeddings. For example, for the following equation, the amount quantitative embedding can be represented as (amount), the account number categorical embedding can be represented as (account), the IP address categorical embedding can be represented as (IP),

and the card type categorical embedding can be represented as (card). The machine learning computer 102 can determine the pair-wise dot products and can generate the upper half of a matrix using the dot products as follows:

$$\begin{bmatrix} - & amount \cdot account & amount \cdot IP & amount \cdot card \\ - & - & account \cdot IP & account \cdot card \\ - & - & - & IP \cdot card \\ - & - & - & - \\ - & - & - & - \end{bmatrix}$$

- 5 [0147] The machine learning computer 102 can flatten the aforementioned matrix into a vector and can combine the vector with the quantitative embedding (e.g., the amount quantitative embedding) to determine the combined embedding, as described in further detail in reference to FIG. 2.

- 10 [0148] In some embodiments, the machine learning computer 102 can combine the plurality of embeddings using the concatenation process. For example, the machine learning computer 102 can concatenate the amount quantitative embedding, the account number categorical embedding, the IP address categorical embedding, and the card type categorical embedding into a vector.

- 15 [0149] In some embodiments, the machine learning computer 102 can combine the plurality of embeddings using the transformer process. For example, the machine learning computer 102 can input the plurality of the embeddings into the transformer to generate a single vector that represents the plurality of embeddings. The vector output from the transformer can be a combined embedding.

- 20 [0150] In some embodiments, the machine learning computer 102 can combine the plurality of embeddings using the convolutional neural network process. For example, the machine learning computer 102 can input the plurality of the embeddings into the convolutional neural network to generate a single vector that represents the plurality of embeddings. The vector output from the convolutional neural network can be a combined embedding.

- 25 [0151] At step 410, after determining the combined embedding, the machine learning computer 102 can generate a prediction based on the combined embedding for the input data. The machine learning computer 102 can generate the prediction using a machine learning

model. The machine learning model can be a multilayer perceptron that is trained to determine predictions as output based on an input combined embedding. The output can be a classification or a probability value that indicates a probability of the input matching a particular classification.

- 5 [0152] As an illustrative example, the machine learning computer 102 can input the combined embedding that represents the amount quantitative embedding, the account number categorical embedding, the IP address categorical embedding, and the card type categorical embedding into the multilayer perceptron to generate a prediction as output. The prediction can predict an aspect of the access request. For example, the prediction can be a classification of “authorized” or “not authorized” for the access request. As another example, the prediction can be a classification of “fraudulent” or “not fraudulent” for the access request.
- 10

III. HIERARCHICAL CATEGORICAL DATA

- [0153] When processing categorical data, the machine learning computer 102 can enrich the categorical data using hierarchical categorical data. The machine learning
- 15 computer 102 can evaluate the categorical data to determine if any of the categories in the categorical data can be utilized to generate one or more new categories to include into the categorical data.

- [0154] FIG. 5 shows a diagram illustrating hierarchical data enrichment according to embodiments. In some cases, categorical data can be sparse and unevenly distributed. The
- 20 machine learning computer 102 can enrich the categorical data by evaluating the categorical data at different hierarchical levels. The machine learning computer 102 can determine a plurality of hierarchical categories for each category in the categorical data. The plurality of hierarchical categories can relate to the category. The categorical data may be duplicated through multiple hierarchical categories but may be structured differently and may relate to
- 25 other categorical data differently in the hierarchical categories.

[0155] The machine learning computer 102 can identify categorical data for a category that includes a sparse graph of data. Categorical data can be represented by a graph of nodes and edges. Categorical data for a particular category may be a sparse graph if there are few edges in the graph compared to the maximal number of edges in the graph.

- [0156] For example, one category of the categorical data that may be a sparse graph can be IP address. The category of IP address can be a sparse graph since it can be assumed that not too many devices share IP addresses with one another at the same time and that certain types of IP addresses can be very popular (e.g., proxy, small ISP, internet cafe, etc.).
- 5 As such, a graph linking users and IP addresses will have many small isolated graphs within the graph. From graph computing perspective, it is advantageous to have a larger graph that can include denser topological information than to have a graph with many small isolated graphs.
- [0157] The machine learning computer 102 can enrich the categorical data for the IP
10 address category using classless inter-domain routing (CIDR) and autonomous system number (ASN) categories. CIDR is an IP address allocation method that improves data routing efficiency on the internet. Machines, servers, and end-user devices that connect to the internet are associated with an IP address. Devices find and communicate with one another by using these IP addresses. CIDR can be used to allocate IP addresses flexibly and
15 efficiently in a networks. ASN is a globally unique identifier that defines a group of one or more IP prefixes run by one or more network operators that maintain a single, clearly-defined routing policy. The categories of IP address, CIDR, and ASN relate to one another, but can be at different hierarchical levels.
- [0158] The machine learning computer 102 can generate one or more additional
20 graphs based on data derived from the category. The data derived from the category can be subcategories or can be alternate ways of organizing the category. For example, for the IP address category, data derived from the category can include CIDR and ASN. The machine learning computer 102 can generate two additional graphs for the IP address category, where the two additional graphs include a CIDR graph and an ASN graph.
- 25 [0159] FIG. 5 illustrates generating, based on a first graph 502, a second graph 504 and a third graph 506. The first graph 502 can be a graph linking users (C) via IP addresses (IP). The second graph 504 can be a CIDR graph that links users (C) to (CIDR)s. The third graph 506 can be a ASN graph that links users (C) to (ASN)s.
- [0160] Graphs linked by CIDR and ASN provide for denser graphs than the original
30 IP address graph. The graphs linked by CIDR and ASN allow information to propagate

farther through the graph, through larger connections of nodes and edges. Using CIDR and ASN provides for additional graphs that include more edges than the IP address graph. Utilizing each of these graphs as different categorical features improves prediction performance in the machine learning model.

- 5 **[0161]** Furthermore, the hierarchical categorical data determination technique further reduces the space requirement for embeddings. For example, an IP address graph can have hundreds of millions of dimensions, while a CIDR graph has hundreds of thousands, and a ASN graph has tens of thousands. The dimensionality of a categorical variable depends on how many possible values exist. For IP address, there can be millions different IP addresses.
- 10 For CIDR and ASN, there are typically only hundreds of values respectively for each.
- [0162]** The machine learning computer 102 can group the IP address by their prefix and can gradually proceed from sparse groupings to denser groupings. The CIDR graph can be evaluated similar to a routing network. A node representing a particular CIDR can include a plurality of IP addresses sharing the same common prefixes. These common prefixes can be
- 15 the addresses for the router or routing device on the internet. A node representing an ASN can include a group of CIDRs. Typically, the ASN can correspond to a network service provider, which are each assigned a number of a network range to which they can provide service. As such, the IP address, the CIDR, and the ASN have a hierarchical relationship with one another.
- 20 **[0163]** Using the IP address, the CIDR and ASN, embodiments can explicitly introduce these three different hierarchies of the category of IP address into new categories for the embedding layer. As such, for one IP address the machine learning computer 102 can generate three embeddings at different hierarchical levels: 1) IP address, 2) CIDR, and 3) ASN as ways to capture different hierarchical categorical information.
- 25 **[0164]** After generating the one or more additional graphs based on data derived from the category, the machine learning computer 102 can include the one or more additional graphs into the categorical data. The data derived from the category can include one or more new categories in the plurality of categories. The new categories can include, for example, CIDR and ASN.

[0165] The use of CIDR and ASN are examples of ways to analyze the categorical in a hierarchical manner, however, it is understood that other hierarchical classifications of the categorical data can be utilized. For example, a physical address can have different hierarchical categories. A physical address (e.g., a shipping address, a billing address, etc.) can include hierarchical categories of a street number, a city, a zip code, a state, and a country. A phone number is another example of a category of categorical data that can be evaluated and utilized to generate additional graphs. A phone number can include an area code and then a first three numbers or a first four numbers and a set of remaining numbers at the end. Another example is a category of email. An email can be used to determine a category of email domain. Email domains are based on internet servers, therefore each email domain has an ending component to the email domain such as .com, .net, personal domain names, etc. The email domains can be used to form a hierarchical structure.

[0166] As an example, prior to determining the plurality of categorical embeddings, the machine learning computer 102 can identify categorical data for a category that includes a sparse graph of data (e.g., identify the category of IP address and the corresponding graph data). Data from a sparse graph can be enriched by determining hierarchical categorical data. The machine learning computer 102 can generate one or more additional graphs based on data derived from the category (e.g., generate the CIDR and ASN graphs). The machine learning computer 102 can then include the one or more additional graphs into the categorical data. The data derived from the category is one or more new categories in the plurality of categories (e.g., the CIDR and ASN are included in the categorical data as new categories along with the original IP address category).

[0167] FIG. 6 shows a flow diagram illustrating a method of additional categorical data in a hierarchical manner according to embodiments. The method illustrated in FIG. 6 can be performed by a computer system such as the machine learning computer 102. The method illustrated in FIG. 6 can occur prior to determining the plurality of categorical embeddings as described in FIG. 2.

[0168] At step 602, the machine learning computer 102 can identifying categorical data for a category based on a predetermined criteria. The predetermined criteria can be that the categorical data for a category includes a sparse graph of data. For example, the

categories of IP address, physical address, and phone number can include sparse graphs of data.

[0169] In some embodiments, the predetermined criteria can indicate that a particular category is to be used to generate additional categories in a hierarchical manner. For example,
5 the IP address can be indicated to always be used to generate additional categories in a hierarchical manner.

[0170] At step 604, after identifying the categorical data for the category, the machine learning computer 102 can generate one or more additional data graphs based on data derived from the categorical data for the category. The data derived from the categorical data can
10 indicate additional categories. The data derived from the category can be one or more new categories to be included in the plurality of categories.

[0171] As an example, the predetermined criteria can be that the categorical data for the category includes a sparse graph of data. The category that includes the sparse graph of data can be IP address. The data derived from the category can include a first category of
15 classless inter-domain routing and a second category of autonomous system number. For example, for the category of IP address, data derived from the categorical data for the IP address category can include classless inter-domain routing (CIDR) and autonomous system number (ASN). The CIDR and the ASN can be utilized to generate additional data graphs from the categorical data from the category of IP address.

20 [0172] The machine learning computer 102 can generate two additional data graphs, one for each new category. A first new data graph can correspond to the first category of CIDR. The first new data graph can include nodes representing CIDR numbers that are connected to nodes representing users. A second new data graph can correspond to the second category of ASN. The second new data graph can include nodes representing ASN
25 values that are connected to nodes representing users.

[0173] As another example, the predetermined criteria can be that the categorical data for the category includes a sparse graph of data. The category that includes the sparse graph of data can be telephone number. The data derived from the category can include a first
30 category of country code, a second category of area code, a third category of prefix, and a fourth category of line number. For example, a first number(s) of a phone number is the

country code, the next three numbers are the area code, the next three numbers are the prefix (which can correspond to smaller areas within the area code's region), and the last four numbers are the line number that indicate the unique phone number in the aforementioned regions.

5 **[0174]** The machine learning computer 102 can generate four additional data graphs, one for each new category. A first new data graph can correspond to the first category of country code. The first new data graph can include nodes representing country codes that are connected to nodes representing users. A second new data graph can correspond to the second category of area code. The second new data graph can include nodes representing area
10 codes that are connected to nodes representing users. A third new data graph can correspond to the third category of prefix. The third new data graph can include nodes representing prefix values that represent sub-areas of the area indicated by the area code that are connected to nodes representing users. A fourth new data graph can correspond to the fourth category of line number. The fourth new data graph can include nodes representing line numbers that are
15 connected to nodes representing users.

[0175] At step 606, after generating the one or more additional data graphs, the machine learning computer 102 can include the one or more additional data graphs into the categorical data. The one or more additional data graphs can correspond to new categories.

IV. TIME INDEXED QUANTITATIVE EMBEDDINGS

20 **[0176]** Quantitative data can change over time and can be represented as time series data. The machine learning model 102 can obtain time data to combine with the quantitative data prior to generating the quantitative embeddings. By combining time data with the quantitative data the machine learning model can capture changes in the quantitative data over time. The machine learning computer 102 can then generate time indexed quantitative
25 embeddings using the combination of the quantitative data and the time data.

[0177] FIG. 7 shows a diagram illustrating a time indexed quantitative embeddings creation method according to embodiments. The machine learning computer 102 can combine quantitative data and time data during data preprocessing in the input layer 210. By combining the quantitative data with the time data, the model can gain a more explicit
30 expression of the time dimension in general.

[0178] The inclusion of the time data allows for trends in the quantitative data to be more easily captured by the machine learning model, since the resulting embedding not only depends on the original quantitative data, but also on a timestamp at which the quantitative data is associated. For example, quantitative data can include velocity data that indicates a number of access requests that occurred within a particular timespan (e.g., 1 month). The number of access requests can be evaluated over time using the time data.

[0179] FIG. 7 illustrates a method of generating a learned time indexed quantitative embedding. The method illustrated in FIG. 7 can be performed by the machine learning computer 102 to combine the quantitative data with time data in the input layer 210 of FIG. 2. The machine learning computer 102 can generate a learned time indexed quantitative embedding.

[0180] The quantitative data can include a plurality of data entries for a particular feature. For example, a feature can be transaction amount, transaction volume, etc. Each feature can have the plurality of data entries that correspond to the feature at specific points in time. For example, a quantitative data feature of transaction volume can have 100 entries of transaction volume and each entry can correspond to a timestamp in the time data. The feature when analyzed along the time dimension is a time-series. A quantitative data feature vector of n-dimensions can be transformed into an n+1 dimensioned time-series when combined with the time data.

[0181] At step 702, the machine learning computer 102 can obtain quantitative data and time data. The machine learning computer 102 can obtain the quantitative data and the time data from the data storage 104. The data storage 104 can store the quantitative data in association with the time data. For example, the machine learning computer 102 can obtain 100 data entries of quantitative data for the feature of transaction volume and 100 timestamps from the time data.

[0182] At step 704, in some embodiments, after obtaining the quantitative data and the time data, the machine learning computer 102 can combine the quantitative data and the time data. The machine learning computer 102 can combine the quantitative data and the time data in any suitable manner.

[0183] In some embodiments, the machine learning computer 102 can combine the quantitative data with the time data using a machine learning model (e.g., a neural network) with a periodical activation function (e.g., such as a sine/cosine function). The machine learning model can be trained to combine quantitative data and time data into single values.

5 For example, the machine learning model can output 100 combination values from the 100 entries of quantitative data for the feature of transaction volume and 100 timestamps of the time data. The 100 combination values can be referred to as 100 machine learning generated combination values.

[0184] In other embodiments, the machine learning computer 102 can combine the quantitative data with the time data by concatenating the quantitative data with the time data.

10 For example, if there are 100 entries of quantitative data for the feature of transaction volume and 100 timestamps of the time data, then the machine learning computer 102 can generate 100 combinations by respectively concatenating the quantitative data with the time data. For example, the machine learning computer 102 can form the following combinations, wherein

15 the symbol + indicates concatenation:

$$(q_1 + t_1), (q_2 + t_2), \dots, (q_{100} + t_{100})$$

[0185] In some embodiments, if the quantitative data and the time data are not combined at step 702, then the machine learning computer 102 can have 100 entries of quantitative data for the feature of transaction volume and 100 timestamps of the time data.

20 [0186] At step 706, the machine learning computer 102 can generate time indexed quantitative embeddings. The machine learning computer 102 can generate the time indexed quantitative embeddings using a time indexed quantitative embedding machine learning model. The time indexed quantitative embedding machine learning model can include a multilayer perceptron or a transformer with an attention-based neural network architecture.

25 [0187] The time indexed quantitative embeddings can be embedding vectors generated from a combination of the quantitative data with the time data. The embedding vectors can be indexed to the time associated with the quantitative data. The embedding process can include any suitable embedding process to obtain an output embedding vector from an input vector.

[0188] Depending on the combinations generated in step 704, the machine learning computer 102 can generate time indexed quantitative embeddings based on 1) the 100 machine learning generated combination values, 2) the 100 concatenation generated combination values, or 3) the 100 entries of the quantitative data for the feature of transaction volume and the 100 timestamps of the time data. The machine learning computer 102 can generate 100 time indexed quantitative embeddings.

[0189] After obtaining the embedding vectors, the machine learning computer 102 can determine the format of the output embedding(s) created from the time indexed quantitative embeddings vectors. The machine learning computer 102 can perform step 708A or step 708B to determine the output embedding(s) that are then provided to the interaction layer at step 710.

[0190] At step 708A, the machine learning computer 102 compress the time indexed quantitative embedding vectors into a single compressed embedding. The machine learning computer 102 can feed all of the time indexed quantitative embedding vectors into a compression method such as a multilayer perceptron or a transformer with attention-based neural network architecture. The compression method can compress the input embedding vectors into a single output embedding vector that represents the features.

[0191] For example, the machine learning computer 102 can input the 100 time indexed quantitative embeddings into a machine learning model, such as a multilayer perceptron or a transformer to generate one compressed embedding that represents the 100 time indexed quantitative embeddings. The compressed embedding can have a predetermined dimensionality such that the compressed embedding can be later utilized in an interaction layer.

[0192] At step 708B, the machine learning computer 102 can stack the time indexed quantitative embedding vectors. For example, each time indexed quantitative embedding vector can be included into a larger tensor that includes all of the time indexed quantitative embedding vectors. As an example, if there are 100 quantitative features in the quantitative data, then the machine learning computer 102 can create 100 time indexed quantitative embedding vectors with, for example, 128 elements per vector. The machine learning

computer 102 can then provide the stacked 100 time indexed quantitative embedding vectors to the interaction layer in step 710.

[0193] At step 710, the machine learning computer 102 can provide the compressed time indexed quantitative embedding or the stack of time indexed quantitative embeddings to the interaction layer (e.g., the interaction layer 230 of FIG. 2).

[0194] FIG. 8 shows a flow diagram illustrating a method of generating time indexed quantitative embeddings according to embodiments. The method illustrated in FIG. 8 can be performed by a computer system such as the machine learning computer 102.

[0195] At step 802, the machine learning computer 102 can obtain time data associated with quantitative data. The time data can be stored in association with the quantitative data in the data storage 104. The time data can include timestamps that correspond to each data entry in the quantitative data.

[0196] At step 804, after obtaining the time data and the quantitative data, the machine learning computer 102 can combine the quantitative data with the time data. The machine learning computer 102 can combine the quantitative data with the time data in any suitable manner.

[0197] In some embodiments, the machine learning computer 102 can combine the quantitative data and the time data using a machine learning model (e.g., a neural network) that is trained to combine quantitative data and time data entries into single values.

[0198] In other embodiments, the machine learning computer 102 can combine the quantitative data and the time data by concatenating each entry in the quantitative data to the corresponding entry in the time data.

[0199] At step 806, after combining the quantitative data and the time data, the machine learning computer 102 can determine one or more time indexed quantitative embeddings using the combination of the quantitative data with the time data. The machine learning computer 102 can determine the one or more time indexed quantitative embeddings using a machine learning model. The machine learning model can be a multilayer perceptron, a recurrent neural network, a transformer, a convolutional neural network, or other machine learning model trained to generate embeddings based on input data.

[0200] At step 808, after determining the one or more time indexed quantitative embeddings, if there are more than one time indexed quantitative embeddings, the machine learning computer 102 can compress the one or more time indexed quantitative embeddings into a compressed quantitative embedding.

5 [0201] The machine learning computer 102 can compress the one or more time indexed quantitative embeddings into a single compressed quantitative embedding using a machine learning model. The machine learning model can be a multilayer perceptron, a recurrent neural network, a transformer, a convolutional neural network, or other machine
10 embeddings.

V. MODEL PROCESSING COMPONENTS

[0202] This section describes model processing components that aid in processing the model and/or processing data related to the model. The machine learning computer 102 can utilize model processing components such as transformers and convolutional neural
15 networks.

A. Transformers

[0203] A transformer can be a type of neural network architecture that can process an input data sequence by utilizing a self-attention mechanism to understand relationships and contexts between different elements within the input sequence, allowing the transformer to
20 effectively learn long-range dependencies. Transformers can be utilized to process data as described herein.

[0204] FIG. 9 shows a diagram illustrating a transformer 900 according to embodiments. The transformer illustrated in FIG. 9 is described in reference to the transformer 900 being utilized to determine a quantitative embedding from quantitative data.
25 However, it is understood that the transformer 900 can be utilized to process other data. For example, the transformer 900 can be utilized to determine a combined embedding based on the one or more quantitative embeddings and the plurality of categorical embeddings. The transformer 900 can be a machine learning model that can encode data.

- [0205] The transformer 900 can accept data as input 902 and can determine an output 922. For example, the input 902 can include quantitative data from the input layer 210 of FIG. 2. The output 922 can include a quantitative embedding that is output to the interaction layer 230 of FIG. 2.
- 5 [0206] The transformer 900 can include an embedding layer 904, a plurality of transformer blocks 906, and a multilayer perceptron (MLP) layer 920. The plurality of transformer blocks 906 can include a first transformer block 908 a second transformer block 912, a third transformer block 916, and a fourth transformer block 918. The first transformer block 908, as an example, can include a multi-head attention mechanism 910. The second
10 transformer block 912, as an example, can include a self-attention mechanism 914.
- [0207] The embedding layer 904 can generate initial embeddings for the input data. The embedding layer 904 can generate an embedding in any suitable manner. For example, the embedding layer 904 can generate initial embeddings for input quantitative data.
- [0208] The embedding layer 904 can convert the input sequence into the
15 mathematical domain that the transformer 900 can process. For example, the input sequence can be split into a series of tokens. The embedding layer 904 can transforms the token sequence into a vector sequence. The vectors carry semantic and syntax information, represented as numbers, and their attributes can be learned during a training process.
- [0209] In some embodiments, the embedding layer 904 or a positional encoding layer
20 (not shown) after the embedding layer 904 can perform positional encoding. Positional encoding can add information to each token's embedding to indicate its position in the sequence. Positional encoding can be performed by using a set of functions that generate a unique positional signal that is added to the embedding of each token. With positional encoding, the model can preserve the order of the tokens and understand the sequence
25 context.
- [0210] The plurality of transformer blocks 906 can include transformer blocks that manipulate data received from a previous block in the transformer 900. The plurality of transformer blocks 906 can include any number of transformer blocks (e.g., 3 transformer blocks, 6 transformer blocks, 10 transformer blocks, etc.).

[0211] Each transformer block in the transformer 900 can have multiple parallel attention heads to learn different sets of attention weights between every pair of features. Multiple attention blocks can be stacked together to learn sophisticated interactions between features. Further, each transformer block can include a self-attention process along with the multiple parallel attention heads. The first transformer block 908 will be discussed in conjunction with the multi-head attention mechanism 910. The second transformer block 912 will be discussed in conjunction with the self-attention mechanism 914. However, it is understood that each transformer block of the plurality of transformer blocks 906 can include a multi-head attention mechanism and a self-attention mechanism.

10 [0212] The first transformer block 908 can include the multi-head attention mechanism 910. The multi-head attention mechanism 910 can perform a multi-head attention process on the received data (e.g., an embedding). The multi-head attention mechanism 910 can include a process for that iterates through an attention mechanism a plurality of times in parallel. Independent attention outputs from the parallel heads are then concatenated and
15 linearly transformed into the expected dimension required by the next block in the transformer 900. Multiple attention heads allows for attending to parts of the sequence differently (e.g., longer-term dependencies versus shorter-term dependencies).

[0213] The second transformer block 912 can include a self-attention mechanism 914. Self-attention is a mechanism used in machine learning to capture dependencies and
20 relationships within input sequences. The input sequence can be the output of the previous block. Self-attention allows the transformer 900 to identify and weight the importance of different parts of the input sequence by attending the input sequence to itself. Self-attention operates by transforming the input sequence into three vectors: query, key, and value. These vectors are obtained through linear transformations of the input. The attention mechanism
25 914 can calculate a weighted sum of the values based on the similarity between the query and key vectors. The resulting weighted sum, along with the original input, can be passed through a feed-forward neural network to produce the final output self-attention mechanism 914. This self-attention process can allow the transformer 900 to focus on relevant information and capture long-range dependencies.

[0214] The third transformer block 916 and the fourth transformer block 918 can further process the data similarly to the first transformer block 908 and the second transformer block 912.

[0215] The multilayer perceptron 920 can be present after the plurality of transformer blocks 906. The multilayer perceptron 920 can be a type of feed forward neural network. The multilayer perceptron 920 can include three types of layers: an input layer, an output layer and a hidden layer. The multilayer perceptron 920 can be trained to output embeddings from the transformer based on inputs received from the plurality of transformer blocks 906. For example, the output of the fourth transformer block 918 can be input into the multilayer perceptron 920. The multilayer perceptron 920 can determine an output based on the input data.

B. Convolutional neural networks

[0216] A convolutional neural network can include a type of neural network that uses convolutional layers to identify and classify data. Convolutional neural networks can be utilized to process data as described herein.

[0217] FIG. 10 shows a diagram illustrating a convolutional neural network according to embodiments. The method illustrated in FIG. 8 can be performed by a computer system such as the machine learning computer 102. The machine learning computer 102 can utilize the convolutional neural network to explore channel interactions among features in input data.

[0218] A convolutional neural network (CNN) can include a regularized type of feed-forward neural network that learns features by via kernel optimization. A convolutional neural network can include an input layer, one or more hidden layers and an output layer. The one or more hidden layers can perform convolutions. The one or more hidden layers can include a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product can be the Frobenius inner product. The convolutional neural network can utilize an activation function, such as rectified linear unit activation function (ReLU). As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This can be

followed by other layers such as pooling layers, fully connected layers, and normalization layers.

[0219] Prior to step 1002, the machine learning computer 102 can obtain data to input into the convolutional neural network. In some embodiments, the machine learning computer
 5 102 can obtain quantitative data to input into the convolutional neural network. In other embodiments, the machine learning computer 102 can obtain categorical data to input into the convolutional neural network. In yet other embodiments, the machine learning computer 102 can obtain a plurality of embeddings (e.g., including quantitative embeddings and categorical embeddings) to input into the convolutional neural network.

10 [0220] At step 1002, the machine learning computer 102 can project the input data into a higher dimensional space. The machine learning computer 102 can project the input data into a higher dimensional space using a fully-connected layer (FCL) projection.

[0221] At step 1004, the machine learning computer 102 can divide the features of the input data into channels to form convolutional input data. For example, the input data can
 15 include categorical data. The categorical data can include data that corresponds to a plurality of categories. For example, the categorical data can include a category of user device identifier, a category of IP address, and a category of email address. The machine learning computer 102 can separate each category of the categorical data into a different channel of the convolutional neural network. As another example, in a convolutional neural network that
 20 evaluates images, there may be three channels: red, green, and blue.

[0222] In the convolutional neural network, the convolutional input data can be a tensor with the following shape:

$$(number\ of\ inputs) \times (input\ height) \times (input\ width) \times (input\ channels)$$

[0223] The number of inputs can be the number of entries in the categorical data. For
 25 example, the categorical data can include entries for five access requests. The input height can be a value of 1, while the input width can be the length of the entries in the categorical data (e.g., 128 dimensions, 512 dimensions, etc.). The number of input channels can be the number of categories included in the categorical data.

[0224] At step 1006, the machine learning computer 102 can apply a convolution to the convolutional input data. The machine learning computer 102 can pass the convolutional input data through a convolutional layer in the convolutional neural network. The convolution can allow the machine learning computer 102 to learn the interactions among
5 features. Convolutional layers convolve the input and pass the result to the next layer in the convolutional neural network. After passing through a convolutional layer, the data becomes abstracted to a feature map (e.g., a map) with a shape as follows:

$$(number\ of\ inputs) \times (map\ height) \times (map\ width) \times (map\ channels)$$

[0225] At step 1008, the machine learning computer 102 can utilize residual
10 connections from the channels utilized in step 1004. The residual connections can help the learning process in deeper networks. Residual connections can include additional links that connect some layers in a neural network to other layers that are not directly adjacent. For example, in the convolutional neural network, each layer can receive input from the previous layer and passes output to the next layer. However, with residual connections, some layers
15 can also receive input from or send output to layers that are several layers away. The residual connections create a parallel path for information flow that bypasses some intermediate layers.

[0226] Residual connections can work by adding the output of a previous layer to the input of a later layer. Utilizing residual connections can allow the later layer to not have to
20 learn the entire function that maps the input to the output, but only the residual or difference between them. For example, if the input is x and the output is y , then the layer with a residual connection only has to learn the function $f(x)$ such that $y = x + f(x)$. Doing so can make the learning process more stable, as the layer can simply learn an identity function $f(x) = 0$ if there is no difference between x and y .

[0227] Residual connections are beneficial for several reasons. For example, residual
25 connections help alleviate a problem of vanishing gradients or exploding gradients, which occurs when the gradients of the loss function become too small or too large as they propagate back through the network. This can cause the network to stop learning or diverge. Residual connections allow the gradients to flow more directly and smoothly through the
30 network, avoiding these extremes. Residual connections also aid in preventing overfitting,

which occurs when the network memorizes the training data and fails to generalize to new data. Residual connections introduce some regularization and diversity to the network, making it less prone to overfitting. Furthermore, residual connections help avoid degradation, which occurs when the network performance deteriorates as more layers are added. Residual
 5 connections enable the network to preserve or improve its performance by learning incremental features rather than redundant or irrelevant ones.

[0228] In some embodiments, the machine learning computer 102 can repeat the convolution multiple times.

[0229] At step 1010, the machine learning computer 102 can project the convoluted
 10 data to a lower dimension using a fully-connected layer. In the fully-connected layer, each node in the output layer connects directly to a node in the previous layer. The final fully-connected layer can generate an output of a predetermined length. For example, the final fully-connected layer can output an embedding generated based on the input data. In some
 15 embodiments, projecting the data into the lower dimension can be an opposite operation from projecting the data into the higher dimension in step 1002.

VI. ADVANTAGES

[0230] Embodiments, as described herein, were compared with XGBoost and MLP
 methods of generating predictions. Embodiments outperformed the other implementations significantly as show in Table 1, below. Table 1 illustrates a ratio between the true positive
 20 rate and the false positive rate for XGBoost, MLP, and embodiments when the models were tested with test data. The XGBoost model obtained a true positive rate and false positive rate value of 0.46. The MLP model obtained a true positive rate and false positive rate value of 0.47. The XGBoost model and the MLP model performed similarly according to this metric with the MLP model having slightly better results. The model according to embodiments
 25 obtained a true positive rate and false positive rate value of 0.50, which outperforms both the XGBoost model and the MLP model.

Model	XGBoost	MLP	Embodiments
TPR_FPR@5	0.46	0.47	0.50

Table 1: experimental results

- [0231] Embodiments provide for a number of advantages. For example, embodiments provide for systems and methods that extract information from quantitative features, categorical features, and their interactions to obtain improved prediction performance
5 compared to previous methods.
- [0232] Prediction performance (e.g., accuracy) can be improved due to 1) hierarchical categorical embedding, 2) time-indexed embedding, and 3) interaction between quantitative embeddings and categorical embeddings.
- [0233] Previous methods did not consistently deliver accurate performance when
10 evaluating access data. During the aforementioned experiment, systems and methods according to embodiments have improved accuracy compared to gradient boost tree models.
- [0234] Although the steps in the flowcharts and process flows described above are illustrated or described in a specific order, it is understood that embodiments of the invention may include methods that have the steps in different orders. In addition, steps may be omitted
15 or added and may still be within embodiments of the invention.
- [0235] Any of the computer systems mentioned herein may utilize any suitable number of subsystems. Examples of such subsystems are shown in FIG. 11 in computer system 1100. In some embodiments, a computer system includes a single computer apparatus, where the subsystems can be the components of the computer apparatus. In other
20 embodiments, a computer system can include multiple computer apparatuses, each being a subsystem, with internal components. A computer system can include desktop and laptop computers, tablets, mobile phones and other mobile devices.
- [0236] The subsystems shown in FIG. 11 are interconnected via a system bus 1124. Additional subsystems such as a printer 1108, keyboard 1116, storage device(s) 1118, monitor 1122 (e.g., a display screen, such as an LED), which is coupled to display adapter 1112, and others are shown. Peripherals and input/output (I/O) devices, which couple to I/O controller 1102, can be connected to the computer system by any number of means known in the art such as input/output (I/O) port 1114 (e.g., USB, FireWire®). For example, I/O port 1114 or external interface 1120 (e.g., Ethernet, Wi-Fi, etc.) can be used to connect computer

system 1100 to a wide area network such as the Internet, a mouse input device, or a scanner. The interconnection via system bus 1124 allows the central processor 1106 to communicate with each subsystem and to control the execution of a plurality of instructions from system memory 1104 or the storage device(s) 1118 (e.g., a fixed disk, such as a hard drive, or optical disk), as well as the exchange of information between subsystems. The system memory 1104 and/or the storage device(s) 1118 may embody a computer readable medium. Another subsystem is a data collection device 1110, such as a camera, microphone, accelerometer, and the like. Any of the data mentioned herein can be output from one component to another component and can be output to the user.

10 [0237] A computer system can include a plurality of the same components or subsystems, for example, connected together by external interface 1120, by an internal interface, or via removable storage devices that can be connected and removed from one component to another component. In some embodiments, computer systems, subsystem, or apparatuses can communicate over a network. In such instances, one computer can be
15 considered a client and another computer a server, where each can be part of a same computer system. A client and a server can each include multiple systems, subsystems, or components. In various embodiments, methods may involve various numbers of clients and/or servers, including at least 10, 20, 50, 100, 200, 500, 1,000, or 10,000 devices. Methods can include various numbers of communication messages between devices, including at least
20 100, 200, 500, 1,000, 10,000, 50,000, 100,000, 500,00, or one million communication messages. Such communications can involve at least 1 MB, 10 MB, 100 MB, 1 GB, 10 GB, or 100 GB of data.

[0238] Aspects of embodiments can be implemented in the form of control logic using hardware circuitry (e.g., an application specific integrated circuit or field programmable
25 gate array) and/or using computer software stored in a memory with a generally programmable processor in a modular or integrated manner, and thus a processor can include memory storing software instructions that configure hardware circuitry, as well as an FPGA with configuration instructions or an ASIC. As used herein, a processor can include a single-core processor, multi-core processor on a same integrated chip, or multiple processing units
30 on a single circuit board or networked, as well as dedicated hardware. Based on the disclosure and teachings provided herein, a person of ordinary skill in the art will know and

appreciate other ways and/or methods to implement embodiments of the present disclosure using hardware and a combination of hardware and software.

[0239] Any of the software components or functions described in this application may be implemented as software code to be executed by a processor using any suitable computer language such as, for example, Java, C, C++, C#, Objective-C, Swift, or scripting language such as Perl or Python using, for example, conventional or object-oriented techniques. The software code may be stored as a series of instructions or commands on a computer readable medium for storage and/or transmission. A suitable non-transitory computer readable medium can include random access memory (RAM), a read only memory (ROM), a magnetic medium such as a hard-drive or a floppy disk, or an optical medium such as a compact disk (CD) or DVD (digital versatile disk) or Blu-ray disk, flash memory, and the like. The computer readable medium may be any combination of such devices. In addition, the order of operations may be re-arranged. A process can be terminated when its operations are completed but could have additional steps not included in a figure. A process may correspond to a method, a function, a procedure, a subroutine, a subprogram, etc. When a process corresponds to a function, its termination may correspond to a return of the function to the calling function or the main function.

[0240] Such programs may also be encoded and transmitted using carrier signals adapted for transmission via wired, optical, and/or wireless networks conforming to a variety of protocols, including the Internet. As such, a computer readable medium may be created using a data signal encoded with such programs. Computer readable media encoded with the program code may be packaged with a compatible device (e.g., as firmware) or provided separately from other devices (e.g., via Internet download). Any such computer readable medium may reside on or within a single computer product (e.g., a hard drive, a CD, or an entire computer system), and may be present on or within different computer products within a system or network. A computer system may include a monitor, printer, or other suitable display for providing any of the results mentioned herein to a user.

[0241] Any of the methods described herein may be totally or partially performed with a computer system including one or more processors, which can be configured to perform the steps. Any operations performed with a processor may be performed in real-time. The term “real-time” may refer to computing operations or processes that are completed

within a certain time constraint. As examples, a time constraint may be 30 seconds, 1 minute, 10 minutes, 30 minutes, 1 hour, 4 hours, 1 day, or 7 days. Thus, embodiments can be directed to computer systems configured to perform the steps of any of the methods described herein, potentially with different components performing a respective step or a respective group of steps. Although presented as numbered steps, steps of methods herein can be performed at a same time or at different times or in a different order. Additionally, portions of these steps may be used with portions of other steps from other methods. Also, all or portions of a step may be optional. Additionally, any of the steps of any of the methods can be performed with modules, units, circuits, or other means of a system for performing these steps.

10 [0242] The specific details of particular embodiments may be combined in any suitable manner without departing from the spirit and scope of embodiments of the disclosure. However, other embodiments of the disclosure may be directed to specific embodiments relating to each individual aspect, or specific combinations of these individual aspects.

15 [0243] The above description of example embodiments of the present disclosure has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the disclosure to the precise form described, and many modifications and variations are possible in light of the teaching above.

[0244] A recitation of "a", "an" or "the" is intended to mean "one or more" unless specifically indicated to the contrary. The use of "or" is intended to mean an "inclusive or," and not an "exclusive or" unless specifically indicated to the contrary. Reference to a "first" component does not necessarily require that a second component be provided. Moreover, reference to a "first" or a "second" component does not limit the referenced component to a particular location unless expressly stated. The term "based on" is intended to mean "based at least in part on."

25 [0245] The claims may be drafted to exclude any element which may be optional. As such, this statement is intended to serve as antecedent basis for use of such exclusive terminology as "solely", "only", and the like in connection with the recitation of claim elements, or the use of a "negative" limitation.

[0246] All patents, patent applications, publications, and descriptions mentioned herein are incorporated by reference in their entirety for all purposes. None is admitted as prior art. Where a conflict exists between the instant application and a reference provided herein, the instant application shall dominate.

WHAT IS CLAIMED IS:

- 1 1. A method performed by a computer system, the method comprising:
2 obtaining input data comprising quantitative data and categorical data, wherein
3 the categorical data corresponds to a plurality of categories;
4 determining one or more quantitative embeddings for the quantitative data;
5 determining a plurality of categorical embeddings for the plurality of
6 categories using a plurality of embedding layers;
7 determining a combined embedding based on the one or more quantitative
8 embeddings and the plurality of categorical embeddings; and
9 generating a prediction based on the combined embedding for the input data.
- 1 2. The method of claim 1, further comprising:
2 prior to determining the plurality of categorical embeddings, identifying
3 categorical data for a category based on a predetermined criteria;
4 generating one or more additional data graphs based on data derived from the
5 categorical data for the category; and
6 including the one or more additional data graphs into the categorical data,
7 wherein the data derived from the category is one or more new categories in the plurality of
8 categories.
- 1 3. The method of claim 2, wherein the predetermined criteria is that the
2 categorical data for the category includes a sparse graph of data, wherein the category that
3 includes the sparse graph of data is IP address and wherein the data derived from the category
4 includes a first category of classless inter-domain routing and a second category of
5 autonomous system number.
- 1 4. The method of claim 2, wherein the predetermined criteria is that the
2 categorical data for the category includes a sparse graph of data, wherein the category that
3 includes the sparse graph of data is telephone number and wherein the data derived from the
4 category includes a first category of country code, a second category of area code, a third
5 category of prefix, and a fourth category of line number.

1 5. The method of claim 1, wherein the one or more quantitative
2 embeddings are one or more time indexed quantitative embeddings wherein determining the
3 one or more indexed quantitative embeddings comprises:
4 obtaining time data associated with the quantitative data from a data storage;
5 combining the quantitative data with time data; and
6 determining one or more time indexed quantitative embeddings using the
7 combination of the quantitative data with the time data using a machine learning model.

1 6. The method of claim 5 further comprising:
2 compressing the one or more time indexed quantitative embeddings into a
3 compressed quantitative embedding.

1 7. The method of claim 5, wherein the time data is stored in the data
2 storage in association with the quantitative data by a network processing computer.

1 8. The method of claim 5, wherein the machine learning model is a first
2 machine learning model, wherein combining the quantitative data with time data comprises:
3 combining the quantitative data with the time data using a second machine
4 learning model, wherein the second machine learning model is a neural network with a
5 periodical activation function.

1 9. The method of claim 5, wherein combining the quantitative data with
2 time data comprises:
3 concatenating the quantitative data with the time data.

1 10. The method of claim 1, wherein determining the combined embedding
2 further comprises:
3 determining a dot product for each pair of each categorical embedding of the
4 plurality of categorical embeddings and each quantitative embedding of the one or more
5 quantitative embeddings;
6 combining each dot product to form an interacted embedding; and
7 generating the combined embedding by concatenating the one or more
8 quantitative embeddings and the interacted embedding.

1 11. The method of claim 1, wherein determining the combined embedding
2 further comprises:
3 generating the combined embedding using a transformer, wherein the one or
4 more quantitative embeddings and the plurality of categorical embeddings are inputs to the
5 transformer.

1 12. The method of claim 1, wherein determining the combined embedding
2 further comprises:
3 concatenating each categorical embedding of the plurality of categorical
4 embeddings and each quantitative embedding of the one or more quantitative embeddings to
5 form the combined embedding.

1 13. The method of claim 1, wherein the quantitative data includes a
2 transaction amount and/or an accumulated transaction volume, and wherein the categorical
3 data includes a card type, a device, an email domain, an IP address, a shipping address,
4 and/or a resource provider identifier.

1 14. The method of claim 1, wherein the prediction is 1) authorized or not
2 authorized, 2) fraudulent or not fraudulent, or 3) accept or do not accept.

1 15. The method of claim 1, wherein the quantitative data and the
2 categorical data relate to an access request that includes a request for access to a resource
3 between a user of a user device and a resource provider of a resource provider computer.

1 16. The method of claim 1, wherein obtaining the input data comprises:
2 obtaining the input data from a data storage, wherein the input data is stored in
3 the data storage by a network processing computer in an access request network.

1 17. The method of claim 1, wherein the method further comprises:
2 prior to obtaining the input data, receiving a prediction request message from a
3 client device, wherein the prediction request message comprises a data identifier that
4 identifies the input data;
5 after generating the prediction, generating a prediction response message
6 comprising the prediction; and

7 providing the prediction response message to the client device.

1 18. The method of any preceding claim, wherein the one or more
2 quantitative embeddings for the quantitative data are determined using a transformer model.

1 19. A computer product comprising a computer readable medium storing a
2 plurality of instructions for controlling a computer system to perform operations of any of the
3 methods above.

1 20. A system comprising:
2 the computer product of claim 19; and
3 one or more processors for executing instructions stored on the computer
4 readable medium.