

TRANSACTION GENERATIVE PRE-TRAINED TRANSFORMER

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application No. 63/519,659, filed August 15, 2023, which is herein incorporated by reference in its
5 entirety for all purposes.

BACKGROUND

[0002] Currently, generative pre-trained transformers can pull information from a large dataset of text to learn about the language, the grammar, and the structure and the meaning of words and sentences. This enables generative pre-trained
10 transformers to understand the context and intent of user queries and generate appropriate responses.

[0003] Generative pre-trained transformer models are transformer neural networks. The transformer neural network architecture uses self-attention mechanisms to focus on different parts of the input text during each processing step.
15 A transformer model captures more context and improves performance on natural language processing (NLP) tasks. A transform can have two main modules an encoder and a decoder.

[0004] Generative pre-trained transformer models typically take different length sentences as input, where each word in each sentence can vary with no required
20 repetition between sentences. For example, a user can input text queries (e.g., "describe a story about space travel") that are input into the transformer. A next input text does necessarily repeat any words in the first input.

[0005] However, in some cases, inputs to the generative pre-trained transformer can be repeatable values, but will still have the same processing as if they
25 did not repeat, which can lead to longer processing times than needed. Embodiments of the disclosure address this problem and other problems individually and collectively.

SUMMARY

[0006] One embodiment is related to a method comprising: obtaining, by a computer, interaction data comprising a plurality of interaction data sets, each interaction data set comprising a plurality of tokens, wherein the plurality of tokens
5 include non-repeatable tokens and repeatable tokens; for each interaction data set: encoding, by the computer, the non-repeatable tokens of the plurality of tokens to form non-repeatable token embeddings, retrieving, by the computer, repeatable token embeddings from memory using the repeatable tokens, wherein the repeatable token embeddings and the non-repeatable token embeddings are in a plurality of first
10 embeddings, concatenating, by the computer, the plurality of first embeddings with positional encodings to form second embeddings, generating, by the computer, an interaction data set embedding based on the second embeddings; determining, by the computer, one or more predicted interaction data set embeddings using interaction data set embeddings comprising the interaction data set embedding for each
15 interaction data set; and decoding, by the computer, the one or more predicted interaction data set embeddings to predicted tokens.

[0007] Another embodiment is related to a computer comprising: a processor; and a computer-readable medium coupled to the processor, the computer-readable medium comprising code executable by the processor for implementing a method
20 comprising: obtaining, by a computer, interaction data comprising a plurality of interaction data sets, each interaction data set comprising a plurality of tokens; for each interaction data set: encoding, by the computer, the plurality of tokens to obtain a plurality of first embeddings, concatenating, by the computer, the plurality of first embeddings with positional encodings to form second embeddings, generating, by the
25 computer, an interaction data set embedding based on the second embeddings; determining, by the computer, one or more predicted interaction data set embeddings using interaction data set embeddings comprising the interaction data set embedding for each interaction data set; and decoding, by the computer, the one or more predicted interaction data set embeddings to predicted tokens.

30 **[0008]** Another embodiment is related to a system comprising: a database storing interaction data stored by a network processing computer; and an analysis computer comprising: a processor; and a computer-readable medium coupled to the

processor, the computer-readable medium comprising code executable by the processor for implementing a method comprising: obtaining, by a computer, interaction data comprising a plurality of interaction data sets, each interaction data set comprising a plurality of tokens; for each interaction data set: encoding, by the
5 computer, the plurality of tokens to obtain a plurality of first embeddings, concatenating, by the computer, the plurality of first embeddings with positional encodings to form second embeddings, generating, by the computer, an interaction data set embedding based on the second embeddings; determining, by the computer, one or more predicted interaction data set embeddings using interaction data set
10 embeddings comprising the interaction data set embedding for each interaction data set; and decoding, by the computer, the one or more predicted interaction data set embeddings to predicted tokens.

[0009] Further details regarding embodiments of the disclosure can be found in the Detailed Description and the Figures.

15 BRIEF DESCRIPTION OF THE DRAWINGS

[0010] FIG. 1 shows a block diagram of an analysis system according to embodiments.

[0011] FIG. 2 shows a block diagram of components of an analysis computer according to embodiments.

20 **[0012]** FIG. 3 shows a diagram of interaction data according to embodiments.

[0013] FIG. 4 shows a flow diagram illustrating an encoding method according to embodiments.

[0014] FIG. 5 shows a diagram of a prediction method according to embodiments.

25 **[0015]** FIG. 6 shows a diagram of a reconstruction method according to embodiments.

DETAILED DESCRIPTION

[0016] Prior to discussing embodiments of the disclosure, some terms can be described in further detail.

[0017] An “interaction” may include a reciprocal action or influence. An
5 interaction can include a communication, contact, or exchange between parties, devices, and/or entities. Example interactions include a transaction between two parties and a data exchange between two devices. In some embodiments, an interaction can include a user requesting access to secure data, a secure webpage, a secure location, and the like. In other embodiments, an interaction can include a
10 payment transaction in which two devices can interact to facilitate a payment.

[0018] “Interaction data” can include data related to and/or recorded during an interaction. In some embodiments, interaction data can be transaction data of the network data. Transaction data can comprise a plurality of data elements with data values.

[0019] A “token” can include a thing serving as a representation of something
15 else. A token can include a smallest meaningful unit of information in a sequence of data. A token can include a data element in a dataset. For example, a transaction data set can comprise a plurality of tokens including a timestamp, a primary account number, an issuer identifier, etc. A token can be a data element.

[0020] “Credentials” may comprise any evidence of authority, rights, or
20 entitlement to privileges. For example, access credentials may comprise permissions to access certain tangible or intangible assets, such as a building or a file. Examples of credentials may include passwords, passcodes, or secret messages. In another example, payment credentials may include any suitable information associated with
25 and/or identifying an account (e.g., a payment account and/or payment device associated with the account). Such information may be directly related to the account or may be derived from information related to the account. Examples of account information may include an “account identifier” such as a PAN (primary account number or “account number”), a token, a subtoken, a gift card number or code, a
30 prepaid card number or code, a user name, an expiration date, a CVV (card verification value), a dCVV (dynamic card verification value), a CVV2 (card verification value 2), a

CVC3 card verification value, etc. An example of a PAN is a 16-digit number, such as “4147 0900 0000 1234”. In some embodiments, credentials may be considered sensitive information.

[0021] A “machine learning model” may include an application of artificial intelligence that provides systems with the ability to automatically learn and improve from experience without explicitly being programmed. A machine learning model may include a set of software routines and parameters that can predict an output of a process (e.g., identification of an attacker of a computer network, authentication of a computer, a suitable recommendation based on a user search query, etc.) based on a “feature vector” or other input data. A structure of the software routines (e.g., number of subroutines and the relation between them) and/or the values of the parameters can be determined in a training process, which can use actual results of the process that is being modeled, e.g., the identification of different classes of input data. Examples of machine learning models include support vector machines (SVM), models that classify data by establishing a gap or boundary between inputs of different classifications, as well as neural networks, which are collections of artificial “neurons” that perform functions by activating in response to inputs. In some embodiments, a neural network can include a convolutional neural network, a recurrent neural network, etc.

[0022] 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.” An analysis 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.

[0023] 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 numerical 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)

[0024] A "large language model" (LLM) can include a probabilistic model of a natural language. A large language model can generate probabilities of a series of words, based on text corpora in one or multiple languages it was trained on. Large language models can include a combination of feedforward neural networks and transformers. A large language model can include artificial neural networks that contain from tens of millions and up to billions of weights and can be (pre-)trained using self-supervised learning and semi-supervised learning. A large language model accept an input text and can repeatedly predict a next token or word as an output.

[0025] A "transformer" can include a deep learning architecture that utilizes attention mechanisms. A transformers can include primary components of 1) tokenizers, which convert text into tokens, 2) embedding layers, which convert tokens into semantically meaningful representations, and 3) transformer layers, which carry out the reasoning capabilities, and can consist of Attention and MLP layers. Transformer layers can be one of two types, encoder and decoder. A transformer can include an encoder, a decoder, or both an encoder and a decoder. Bidirectional encoder representations from transformers (BERT) are an example of encoder-only

models, while generative pre-trained transformers (GPT) are an example of decoder-only models.

[0026] An “encoder” can include a type of neural network that converts data from a raw format into another form. An encoder can include a self-attention mechanism and a feed-forward neural network. The self-attention mechanism accepts input encodings from the previous encoder and weights the encodings’ relevance to each other to generate output encodings. The feed-forward neural network further processes each output encoding individually. These output encodings are then passed to the next encoder as its input in a looping manner to encode a full input text. The encoder can be bidirectional in terms of attention, for example, attention can be placed on tokens before and after the current token.

[0027] A “decoder” can include a type of neural network that generates output from encoded representations. A decoder can include attention mechanisms (e.g., a self-attention mechanism and an attention mechanism over the encodings) and a feed-forward neural network. A decoder can function in a similar fashion to an encoder, but an additional attention mechanism is inserted, which instead draws relevant information from the encodings generated by the encoders. This mechanism can also be called the encoder-decoder attention.

[0028] An “embedding” can include vector representation of data. An embedding can be a low-dimensional learned continuous vector representation of discrete variables. An embedding can be generated using a neural network that is trained to convert input data into output vector representations of the input data.

[0029] A “question” can include a sentence worded or expressed so as to elicit information. A question can include text strings in a plaintext format. A question can be an input message into a large language model.

[0030] 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).

5 **[0031]** 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.

10 **[0032]** 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
15 a variety of computing structures, arrangements, and compilations for servicing the requests from one or more client computers.

20 **[0033]** Embodiments of the disclosure allow for systems and methods to structure temporal based interaction data into two dimensional structures. Embodiments also provide for interaction predictions based on current interaction data using a transformer system that includes a vertical transformer encoder, a horizontal transformer, and a vertical non-recurrent transformer decoder. Embodiments can provide for systems and methods that generate predictions of a next interaction based on historical interaction data.

25 **[0034]** FIG. 1 shows a system 100 according to embodiments of the disclosure. The system 100 comprises an analysis computer 102, a database 104, a network processing computer 106, an authorizing entity computer 108, a transport computer 110, a resource provider computer 112, and an access device 114.

30 **[0035]** The analysis computer 102 can be in operative communication with the database 104. The network processing computer 106 can be in operative communication with the authorizing entity computer 108, the database 104, and the

transport computer 110. The resource provider computer 112 can be in operative communication with the transport computer 110 and the access device 114.

[0036] 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
5 more than one of each component. In addition, some embodiments of the invention may include fewer than or greater than all of the components shown in FIG. 1.

[0037] Messages between at least the devices included in the system 100 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
10 Hypertext Transfer Protocol (HTTPS), SSL, ISO (e.g., ISO 8583) and/or the like. The communications network may include any one 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
15 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 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
20 through the use of mutual authentication and a session key, and establishment of a Secure Socket Layer (SSL) session.

[0038] The analysis computer 102 can include a computer or a server computer that can be configured to analyze data. The analysis computer 102 can analyze interaction data that is stored in the database 104. The analysis computer 102 can
25 generate predicted tokens for predicted interactions based on the interaction data.

[0039] The database 104 can include any suitable database. The database 104 may be a conventional, fault tolerant, relational, scalable, secure database such as those commercially available from Oracle™ or Sybase™. The database 104 can store interaction data. The interaction data can include a plurality of interaction data sets,
30 where each interaction data set includes a plurality of tokens. A token can be a data element. For example, interaction data set can correspond to a particular interaction (e.g., transaction). The interaction data set can include a plurality of tokens including

a timestamp, a primary account number (PAN), a zipcode, and/or other data related to the interaction. The plurality of tokens can include non-repeatable tokens, repeatable tokens, and partially repeatable tokens.

[0040] The network processing computer 106 can include a computer or a server computer. The network processing computer 106 may route or switch messages between a number of transport computers including the transport computer 110, and a number of authorizing entity computers including the authorizing entity computer 108. The network processing computer 106 may be a payment processing computer in some embodiments. The network processing computer 106 may be configured to provide authorization services, and clearing and settlement services for payment transactions. A network processing computer 106 may include data processing subsystems, networks, and operations used to support and deliver authorization services, exception file services, and clearing and settlement services. An exemplary payment processing network may include VisaNet™. Payment 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 Visa Integrated Payments (VIP) system which processes authorization requests and a Base II system which performs clearing and settlement services. Furthermore, the payment processing network may include a server computer and may use any suitable wired or wireless telecommunications network, including the Internet. In some embodiments, the processing network computer may forward an authorization request received from a transport computer to the authorizing entity computer via a communication channel. The processing network computer may further forward an authorization response message received from the authorizing entity computer to the transport computer.

[0041] The authorizing entity computer 108 may be configured to authorize any suitable request, including access to data, access to a location, or approval for a payment. In some embodiments, the authorizing entity computer 108 may be operated by an account issuer. Typically, the issuer is an entity (e.g., a bank) that issues and maintains an account of a user. The account may be a credit, debit, prepaid, or any other type of account.

[0042] The transport computer 110 can include a computer or a server computer. The transport computer 110 be located between (in an operational sense) the access device 114 and the network computer 118. The transport computer 110 may be operated by an entity such as an acquirer. An acquirer can maintain an
5 account of any merchants with which users may interact.

[0043] The resource provider computer 112 can include any suitable computational apparatus operated by a resource provider (e.g., a merchant). In some embodiments, the resource provider computer 112 may include one or more server computers that may host one or more websites associated with the resource provider
10 (e.g., a merchant). In some embodiments, the resource provider computer 112 may be configured to send data to a network processing computer 106 via a transport computer 110 as part of a payment verification and/or authentication process for a interaction between the user (e.g., consumer) and the resource provider. In some embodiments, the resource provider computer 112 may be accessed via an resource
15 provider-operated website accessible to a user device. The website may be configured to be accessible from an application (e.g., a browser application, the resource provider application 116, etc.) operating on the user device.

[0044] The access device 114 can include any suitable device that provides access to a remote system. The access device 114 may also be used for
20 communicating with the resource provider computer 112, the transport computer 110, or any other suitable system. The access device 114 may be located at a resource provider location, such as a merchant location. The access device 114 may be in any suitable form. Some examples of access devices include point of sale (POS) devices, cellular phones, PDAs, personal computers (PCs), tablet PCs, hand-held specialized
25 readers, set-top boxes, electronic cash registers (ECRs), automated teller machines (ATMs), virtual cash registers (VCRs), kiosks, security systems, access systems, and the like. The access device 114 may use any suitable contact or contactless mode of operation to send or receive data from, or associated with, a mobile communication or payment device. In some embodiments, in which the access device 114 may comprise
30 a POS terminal, any suitable POS terminal may be used and may include a reader, a processor, and a computer-readable medium. A reader may include any suitable contact or contactless mode of operation. For example, exemplary card readers can

include radio frequency (RF) antennas, optical scanners, bar code readers, or magnetic stripe readers to interact with a payment device and/or mobile device. In some embodiments, a cellular phone, tablet, or other dedicated wireless device used as a POS terminal may be referred to as a mobile point of sale or an “mPOS” terminal.

5 **[0045]** FIG. 2 shows a block diagram of an analysis computer 102 according to embodiments. The exemplary analysis computer 102 may comprise a processor 204. The processor 204 may be coupled to a memory 202, a network interface 206, and a computer readable medium 208. The computer readable medium 208 can comprise a vertical transformer encoder module 208A, a horizontal transformer module 208B,
10 and a transformer decoder module 208C.

[0046] The memory 202 can be used to store data and code. For example, the memory 202 can store tokens, data sets, machine learning models, etc. The memory 202 may be coupled to the processor 204 internally or externally (e.g., cloud based data storage), and may comprise any combination of volatile and/or non-volatile
15 memory, such as RAM, DRAM, ROM, flash, or any other suitable memory device.

[0047] The computer readable medium 208 may comprise code, executable by the processor 204, for performing a method comprising: obtaining, by a computer, interaction data comprising a plurality of interaction data sets, each interaction data set comprising a plurality of tokens, wherein the plurality of tokens include non-
20 repeatable tokens and repeatable tokens; for each interaction data set: encoding, by the computer, the non-repeatable tokens of the plurality of tokens to form non-repeatable token embeddings, retrieving, by the computer, repeatable token embeddings from memory using the repeatable tokens, wherein the repeatable token embeddings and the non-repeatable token embeddings are in a plurality of first
25 embeddings, concatenating, by the computer, the plurality of first embeddings with positional encodings to form second embeddings, generating, by the computer, an interaction data set embedding based on the second embeddings; determining, by the computer, one or more predicted interaction data set embeddings using interaction data set embeddings comprising the interaction data set embedding for each
30 interaction data set; and decoding, by the computer, the one or more predicted interaction data set embeddings to predicted tokens.

[0048] The vertical transformer encoder module 208A may comprise code or software, executable by the processor 204, for transforming vertical columns of data entries in interaction data (e.g., interaction data sets) into an interaction data set embedding. Each interaction data set comprises a plurality of tokens, where the plurality of tokens include non-repeatable tokens and repeatable tokens.

[0049] The vertical transformer encoder module 208A, in conjunction with the processor 204, can process the interaction data set to form the interaction data set embedding. To generate the interaction data set embedding, the vertical transformer encoder module 208A, in conjunction with the processor 204, can perform the following processing steps.

[0050] The vertical transformer encoder module 208A, in conjunction with the processor 204, can generate a plurality of first embeddings, where each first embedding of the plurality of first embeddings corresponds to a token of the interaction data set. To generate the plurality of first embeddings, the vertical transformer encoder module 208A, in conjunction with the processor 204, can process each type of token (e.g., repeatable token, non-repeatable token, partially repeatable token) in a different manner.

[0051] The vertical transformer encoder module 208A, in conjunction with the processor 204, can encode the non-repeatable tokens of the plurality of tokens to form non-repeatable token embeddings. The vertical transformer encoder module 208A, in conjunction with the processor 204, can encode the non-repeatable tokens using a machine learning model (e.g., a multi-layer perceptron (MLP)) that is trained to generate non-repeatable token embeddings from non-repeatable tokens. The multi-layer perceptron is a type of artificial neural network that includes multiple layers of neurons, which can use nonlinear activation functions, allowing the network to learn complex patterns in data.

[0052] The vertical transformer encoder module 208A, in conjunction with the processor 204, can retrieve repeatable token embeddings from memory using the repeatable tokens. The analysis computer 102 can cache repeatable tokens in the memory 202 or other in data storage means (e.g., a repeatable token embedding database). The vertical transformer encoder module 208A, in conjunction with the processor 204, can obtain a repeatable token embedding from the memory 202 using

the repeatable token. The repeatable token embeddings can be previously computed in a similar manner to the non-repeatable tokens (e.g., using a multi-layer perceptron).

[0053] In some embodiments, the vertical transformer encoder module 208A, in conjunction with the processor 204, can also determine partially repeatable token embeddings for partially repeatable tokens. The vertical transformer encoder module 208A, in conjunction with the processor 204, can split the partially repeatable token into a repeatable token portion and a non-repeatable token portion. The vertical transformer encoder module 208A, in conjunction with the processor 204, can obtain a non-repeatable token embedding portion for the non-repeatable token portion from, for example, the memory 202. The vertical transformer encoder module 208A, in conjunction with the processor 204, can determine a repeatable token embedding portion for the repeatable token portion using the machine learning model (e.g., the multi-layer perceptron). The vertical transformer encoder module 208A, in conjunction with the processor 204, can then generate a partially repeatable token embedding based on the repeatable token embedding portion and the non-repeatable token embedding portion using a partially repeatable token embedding machine learning model. The partially repeatable token embedding machine learning model can be trained to generate partially repeatable token embeddings based on two inputs including the repeatable token embedding portion and the non-repeatable token embedding portion.

[0054] The repeatable token embeddings, the non-repeatable token embeddings, and the partially repeatable token embeddings are in a plurality of first embeddings.

[0055] After generating the plurality of first embedding, the vertical transformer encoder module 208A, in conjunction with the processor 204, can concatenate the plurality of first embeddings with positional encodings to form second embeddings. The positional encodings can indicate the position of the corresponding token in the interaction data set (e.g., position 1).

[0056] The vertical transformer encoder module 208A, in conjunction with the processor 204, can then generate a plurality of transformed second embeddings and an interaction data set embedding based on the second embeddings using a plurality of transformer blocks in the transformer. The interaction data set embedding can

summarize a plurality of transformed second embeddings, where each transformed second embedding corresponds to a token of the interaction data set.

[0057] The horizontal transformer module 208B can include may comprise code or software, executable by the processor 204, for transforming interaction data horizontally across interactions. The horizontal transformer module 208B, in conjunction with the processor 204, can accept a plurality of interaction data set embeddings (e.g., generated by the vertical transformer encoder module 208A) as input. The horizontal transformer module 208B, in conjunction with the processor 204, can evaluate a sequence in time of the interaction data set embeddings in the plurality of interaction data set embeddings. Each interaction data set embedding can be an embedding that represents an interaction. The horizontal transformer module 208B, in conjunction with the processor 204, can evaluate the changes in interactions over time using a cross attention process. The horizontal transformer module 208B, in conjunction with the processor 204, can determine one or more predicted interaction data set embeddings using interaction data set embeddings comprising the interaction data set embedding for each interaction data set. A predicted interaction data set embeddings can be generated as a next interaction data set embedding in the sequence of interaction data set embeddings similar to generating a next word in a sentence.

[0058] The transformer decoder module 208C can include may comprise code or software, executable by the processor 204, for decoding predicted interaction data set embeddings to predicted tokens. The transformer decoder module 208C, in conjunction with the processor 204, can decode the one or more predicted interaction data set embeddings (e.g., determined by the horizontal transformer module 208B) to predicted tokens of a predicted interaction. The transformer decoder module 208C, in conjunction with the processor 204, can decode a predicted interaction data set embedding by performing the following processing steps.

[0059] The transformer decoder module 208C, in conjunction with the processor 204, can generate an intermediate embedding for each predicted interaction data set embedding based on the predicted interaction data set embedding and corresponding positional encoding using a multi-layer perceptron. The transformer decoder module 208C, in conjunction with the processor 204, can determine the same

number of intermediate embeddings as there are tokens in the original interaction dataset. Each intermediate embedding can correspond to a token of the interaction dataset.

5 **[0060]** The transformer decoder module 208C, in conjunction with the processor 204, can then input the intermediate embeddings into a plurality of transformer blocks (e.g., transformer decoder blocks) to obtain output embeddings. Each output embedding can correspond to a token. The transformer blocks can carry out reasoning capabilities and can include attention and multi-layer perceptron layers. The transformer blocks can determine the interactions between tokens via an attention
10 mechanism. As such, using the transformer blocks, the computer can determine how much each input embedding relates to the other input embedding.

[0061] The transformer decoder module 208C, in conjunction with the processor 204, can then determine a predicted token from each intermediate embedding. The transformer decoder module 208C, in conjunction with the processor
15 204, can concatenate the intermediate embedding with a positional encoding that indicates the position of the token represented by the intermediate embedding in the interaction data set. The transformer decoder module 208C, in conjunction with the processor 204, can then utilize a multi-layer perceptron to determine the predicted token from the positionally encoded intermediate embedding.

20 **[0062]** In some embodiments, any of the transformers described herein can include one or more encoders and one or more decoders. Each encoder can include two layers: 1) a self-attention layer and 2) a feed forward neural network layer. The encoder's inputs first flow through the self-attention layer. The self-attention layer can help the encoder look at other tokens in the input sentence (e.g., interaction dataset)
25 as the encoder encodes a specific token. The decoder can have a self-attention layer, an encoder-decoder attention layer, and a feed forward neural network layer. The encoder-decoder attention layer can help the decoder focus on relevant parts of the input sentence.

[0063] The attention process can be performed in any suitable manner. As an
30 example, a first step in calculating self-attention is to create three vectors from each of the encoder's input vectors. So for each input, the encoder creates a query vector,

a key vector, and a value vector. These vectors are created by multiplying the embedding by three matrices that were trained during the training process.

[0064] A second step in calculating self-attention can be to calculate a score. For example, the computer can be calculating the self-attention for a first token (or an embedding thereof) of a primary account number of "1234567890123456." The self-attention layer can score each token of the input sentence against this token. The score can determine how much focus to place on other parts of the input sentence as the computer encodes a token at a certain position in the sentence. The score can be calculated by taking the dot product of the query vector with the key vector of the respective token that is being scored. So if the computer is processing the self-attention for the token in position #1, the first score would be the dot product of q1 and k1. The second score would be the dot product of q1 and k2.

[0065] A third and a fourth step are to divide the scores by a value (e.g., 8) to obtain more stable gradients overtime. The result can then be passed through a softmax operation. The softmax can normalize the scores so they're all positive and add up to 1. The softmax score can determine how much each token will be expressed at this position. A fifth step is to multiply each value vector by the softmax score (in preparation to sum them up). A sixth step is to sum up the weighted value vectors. This can produce the output of the self-attention layer at this position (for the first token). This process can be repeated for each token to determine the attention values.

[0066] The network interface 206 may include an interface that can allow the analysis computer 102 to communicate with external computers. The network interface 206 may enable the analysis computer 102 to communicate data to and from another device (e.g., the database 104, etc.). Some examples of the network interface 206 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 206 may include Wi-Fi™. Data transferred via the network interface 206 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 206 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
5 network, the Internet, or any other suitable medium.

[0067] FIG. 3 shows a diagram of interaction data 300 according to embodiments. The dataset illustrated in FIG. 3 illustrates a number of tokens that are included in interaction data from multiple interactions (e.g., transactions). A token can include a piece of data. The tokens include non-repeatable tokens, repeatable tokens,
10 and partially repeatable tokens. Non-repeatable tokens may not repeat between interactions. Repeatable tokens may repeat between interactions. Partially repeatable tokens may partially repeat between interactions.

[0068] The non-repeatable tokens can include a timestamp 302. Non-repeatable tokens can also include security values such as a nonce. A non-repeatable
15 token may not repeat at all between interactions or may rarely repeat between interactions.

[0069] The repeatable tokens can include a primary account number (PAN) 304, an issuer identifier 306, an acquirer identifier 308, a zip code 310, a resource provider category code (MCC) 312, and a resource provider identifier 314. A
20 repeatable token can repeat in every interaction or in some of the interactions. For example, a user can be associated with a primary account number 304. When evaluating interactions performed by that user, the primary account number 304 will repeat in each interaction. The resource provider category code 312 can repeat in some of the interactions for the user, but may not necessarily be repeated in every
25 interaction.

[0070] The partially repeatable tokens can include a combined data entry of a resource provider identifier and an amount of the interaction 316. A partially repeatable token can include data that may partially repeat between interactions (e.g., the first N digits of the partially repeatable token repeat). A partially repeatable token
30 can be a combination of a repeatable token and a non-repeatable token.

[0071] Over time, interactions involving interaction data can be performed and stored in a dataset in the database 104 by a data collection computer, such as the network processing computer 106. Interaction data associated with an interaction can be included in the dataset as a new column. FIG. 3 illustrates an interaction data set 318, which can be associated with a particular interaction. The interaction data set 318 can be utilized as a vertical sentence / multi-token word.

[0072] The interaction data 300 can include horizontal rows that include similar data entry types for each interaction. For example, the row 320 can include data for the issuer identifiers including the issuer identifier 306. As another example, the row 322 can include data for the resource provider identifiers including the resource provider category code 312. Each interaction can include the tokens in the same order such that a row in the overall dataset includes the same data type (e.g., zip code, MCC, etc.).

[0073] FIG. 4 shows a flow diagram illustrating an encoding method according to embodiments. FIG. 4 illustrates generating interaction data set embeddings. The method illustrated in FIG. 4 can be performed by a computer (e.g., an analysis computer). During the method of FIG. 4, the computer can generate an interaction data set embedding for each interaction data set of the interaction data.

[0074] Prior to step 402, the computer can obtain interaction data comprising a plurality of interaction data sets. Each interaction data set can include a plurality of tokens. The computer can obtain the interaction data from a database.

[0075] At step 402, the computer can parse the interaction data set for the interaction into a plurality of tokens (e.g., a plurality of data elements). The interaction data set can include non-repeatable tokens (NRTs), repeatable tokens (RTs), and partially repeatable tokens (PRTs).

[0076] At steps 404, 406, and 408, the computer can encode the plurality of tokens to obtain a plurality of first embeddings. In some embodiments, each type of token (e.g., NRT, RT, or PRT) can be embedded in a different manner for efficiency due to some tokens repeating. The plurality of first embeddings can include non-repeatable token embeddings, repeatable token embeddings, and partially repeatable token embeddings.

[0077] At step 404, the computer can encode the non-repeatable tokens using a machine learning model (e.g., a multi-layer perceptron). The machine learning model can be trained to generate embeddings based on input non-repeatable tokens. The computer can iteratively input each non-repeatable token from the interaction data
5 into the machine learning model to generate non-repeatable token embeddings.

[0078] At step 406, the computer can encode the repeatable tokens using an embed lookup process. Since the repeatable tokens can repeat throughout a plurality of interactions, the embedding for the token can be cached in a manner such that it can be efficiently obtained via lookup. The computer or other device can pre-generate
10 repeatable token embeddings for repeatable tokens using a machine learning model that is trained to generate embeddings based on repeatable tokens. The machine learning model that generates repeatable token embeddings can be the same as the machine learning model that generates non-repeatable token embeddings. The computer or other device can store the pre-generated repeatable token embeddings
15 into a database, memory, or other storage means. Each repeatable token can be stored in association with the corresponding repeatable token embedding.

[0079] During step 406, the computer can retrieve repeatable token embeddings from the database or the memory using the repeatable tokens.

[0080] At step 408, the computer can separate the partially repeatable tokens
20 into two portions that include repeatable token portions and non-repeatable token portions. The computer can encode the non-repeatable token portions using a machine learning model, similar to encoding the non-repeatable tokens, to obtain non-repeatable token portion embeddings. The computer can utilize an embed lookup process for the repeatable token portions, similar to obtaining embeddings for a
25 repeatable token, to obtain repeatable token portion embeddings.

[0081] After obtaining the non-repeatable token portion embedding and the repeatable token portion embedding for each partially repeatable token, the computer can generate a partially repeatable token embedding based on the non-repeatable token portion embedding and the repeatable token portion embedding. The computer
30 can input the non-repeatable token portion embedding and the repeatable token portion embedding for both the non-repeatable token portions and the repeatable token portions into a machine learning model to obtain an output embedding. The

machine learning model can be trained to combine two input embeddings and generate one output embedding.

[0082] In some embodiments, each embedding for each token can be in the same embedding space and can be of any suitable dimensionality.

5 **[0083]** At steps 410, 412, and 414, the computer can combine the plurality of first embeddings (e.g., the embeddings generated during steps 404, 406, and 408) with positional encodings to form second embeddings. The positional encodings can include a vector of ones and zeros. The positional encodings can include a 1 in a vector position element that indicates the position of the originating token from the
10 interaction data set. For example, the interaction data sets illustrated in FIG. 3 include 8 elements. The positional encoding can be a vector of length 8. The positional encoding relating to a timestamp can be [1, 0, 0, 0, 0, 0, 0, 0]. The positional encoding relating to a zip code can be [0, 0, 0, 0, 1, 0, 0, 0]. The positional encodings can be a one-hot encoding.

15 **[0084]** The plurality of first embeddings can be combined with the positional encodings via concatenation. The combination of the plurality of first embeddings with the positional encodings can form second embeddings.

[0085] For example, each non-repeatable token embedding can be concatenated with a positional encoding that identifies the position of the
20 corresponding non-repeatable token in the interaction data. Each repeatable token embedding can be concatenated with a positional encoding that identifies the position of the corresponding repeatable token in the interaction data. Each partially repeatable token embedding can be concatenated with a positional encoding that identifies the position of the corresponding partially repeatable token in the interaction
25 data.

[0086] At step 416, the computer can generate a plurality of transformed embeddings 418 and an interaction data set embedding 420 based on the second embeddings. The computer can input the second embeddings into one or more transformer blocks to output the interaction data set embedding 420 and the plurality
30 of transformed embeddings 418. The plurality of transformed embeddings 418 can include an embedding that corresponds to each token in the originating interaction

data. The interaction data set embedding 410 can be a summary embedding that represents the embeddings that correspond to tokens.

[0087] The one or more transformer blocks can include a deep learning architecture that can utilize parallel multi-head attention mechanisms. Attention mechanisms are described in further detail in A. Vaswani et. al., "Attention Is All You Need," arXiv:1706.03762 [cs], June 2017. The transformer block can include the following components 1) tokenizers, which convert text into tokens; 2) embedding layers, which convert tokens into semantically meaningful representations; and 3) transformer layers, which carry out the reasoning capabilities, and consists of Attention and MLP layers. The transformer layer can be of two types, encoder and decoder. The transformer layer can include both an encoder and decoder or only an encoder or only a decoder. BERT is an example of encoder-only model (J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv:1810.04805 [cs], May 2019). Generative pre-trained transformers (GPT) are examples of decoder-only models.

[0088] For example, the computer can utilize the transformer to model the interactions between the different tokens. For example, if a user consistently utilizes a credit card for payment interactions related to airline companies (e.g., used for purchasing plane tickets, but not for other types of resources). The transformer block can identify the relationship between the account number of the credit card and the resource provider category code of the interactions (e.g., airline). Such relationships can be taken into account when generating the plurality of transformed embeddings 418 and the interaction data set embedding 420.

[0089] FIG. 5 shows a diagram of a prediction method according to embodiments. The method illustrated in FIG. 5 can include a horizontal transformer 500 that processes data from the horizontal rows in the interaction data 300. The horizontal transformer 500 can include a plurality of transformer encoders 502 (or transformer encoder blocks) and a plurality of transformer decoder blocks 504.

[0090] After generating the interaction data set embedding 420 (e.g., generated in FIG. 4) and one or more other generated interaction data set embeddings, the computer can determine one or more predicted interaction data set embeddings using interaction data set embeddings. The computer can input the generated interaction

data set embedding 420, along with other interaction data set embeddings, into the horizontal transformer 500 that includes the plurality of transformer encoders 502 and the plurality of transformer decoder blocks 504 that utilize a cross attention process.

[0091] The computer can combine each interaction data set embedding with a positional encoding (PE) 506. The positional encoding 506 can be a timestamp of when an interaction occurred that corresponds to the interaction data set embedding. The positional encoding 506 can be a numerical scalar or vector. The computer can combine each interaction data set embedding with a positional encoding by concatenating the positional encoding to the end of the interaction data set embedding.

[0092] After combining each interaction data set embedding with a positional encoding, the computer can create an input sequence 508 of interaction data set embeddings (which are positionally encoded) that include historical interactions over time. The computer can input the input sequence 508 into a first transformer encoder block of the plurality of transformer encoders 502.

[0093] The first transformer encoder block of the plurality of transformer encoders 502 can encode the input sequence of interaction data set embeddings. The output of the first transformer encoder block can be provided to the next transformer encoder block. Each subsequent transformer encoder block can iteratively encode received data. The final transformer encoder block N can output a sequence encoding 510. The sequence encoding 510 can be an encoding of the input sequence.

[0094] The encoder portion of the transformer can be responsible for processing the input data and creating a fixed-dimensional representation (often called a "context" or a "thought vector") of the input sequence. The encoder can take an input sequence (e.g., a sentence or an image) and can transform the input sequence into a continuous representation. Encoders can extract relevant features from the input data, capturing the contextual information necessary for the task. At the end of encoding, the encoder can produce a vector that can summarize the input sequence's information. This vector can be used as an initial input for the decoder (or a cross attention process).

[0095] After generating the sequence encoding 510, the computer can process the sequence encoding 510 with the plurality of transformer decoder blocks 504. Each transformer decoder block can accept the sequence encoding 510 as processed during a cross attention process. For example, the output of the last transformer encoder block is transformed by the cross attention 512 into a set of attention vectors K and V (e.g., one set for each of the sequence encodings 510). These are to be used by each decoder in its “encoder-decoder attention” layer which helps the decoder focus on appropriate places in the input sequence. Each transformer decoder block of the plurality of transformer decoder blocks 504 (including transformer decoder blocks including blocks 1-N) accepts the set of attention vectors determined from the sequence encoding 510 using the cross attention 512.

[0096] The computer can process each transformer decoder block of the plurality of transformer decoder blocks 504 X times, where X can be the number of interaction data set embeddings included in the input sequence and a number predicted interaction data set embeddings. Each time the plurality of transformer decoder blocks 504 are executed, the computer can increment a decoding time step.

[0097] In embodiments, the first transformer decoder block can accept the set of attention vectors (K and V) associated with an interaction data set embedding / positional encoding combination 508A as input. The first transformer decoder block can process the set of attention vectors (K and V) associated with an interaction data set embedding / positional encoding combination 508A. The output of the first transformer decoder block is provided to the next transformer decoder block, which also receives the set of attention vectors from the sequence encoding 510. Each subsequent transformer decoder block can process the previous transformer decoder block’s output and the set of attention vectors. The last transformer decoder block (e.g., transformer decoder block N) can output a first translated interaction data set embedding (515A) / positional encoding combination 516A. Then, the next the set of attention vectors (K and V) associated with a second interaction data set embedding (515B)/ positional encoding combination 508B, and the first translated interaction data set embedding / positional encoding combination can be input to the transformer decoder blocks 1-N to obtain a second translated interaction data set embedding / positional encoding combination 516B. The next iteration would input the set of

attention vectors (K and V) associated with a third interaction data set embedding (515C)/ positional encoding combination 508C and the first and second translated interaction data set embedding / positional encoding combinations 516A, 516B into the transformer decoder blocks 1-N to obtain a third translated interaction data set embedding / positional encoding combination 516C. This process can repeat for each of the interaction data set embedding / positional encoding combinations.

[0098] Stated generally, after processing each transformer decoder block, the computer can increment a decoding time step and proceed to process the next decoder round where the previous round's output is used as input to the first transformer decoder block. Each round of processing by the plurality of transformer blocks can generate a next translated interaction data set embedding. For example, at seventh decoding time in FIG. 5, the plurality of transformer decoder blocks can output a predicted interaction data set embedding 514 after generating the previous six interaction data set embeddings (which, in some embodiments, can be the same as the original input interaction data set embeddings).

[0099] The computer can utilize the transformer decoder blocks to both generate future transactions and reconstruct input interactions. For example, the computer can generate predicted interactions that are different from the original interactions. The predicted interactions can share statistical properties with the historical interactions, but since they have predictive power due to the transformation process, the predicted interactions are the most likely ones in the future. The computer also determines the reconstructed input interactions, which can be statistically similar to the input data.

[0100] In some embodiments, the horizontal transformer 500 can include a standard transformer that does not utilize cross attention and using a decoder only architecture, as known to one of skill in the art. A decoder only architecture can be utilized when there is limited historical data for a particular payment device (e.g., if a particular credit card has only ever been utilized in three interactions). The decoder transformer may not include the cross-attention process as included in the method illustrated in FIG. 5, due to the lack of historical data.

[0101] FIG. 6 shows a diagram of a reconstruction method according to embodiments. A computer can decode one or more predicted interaction data set

embeddings to predicted tokens. The predicted tokens can include data relating to a predicted interaction (e.g., a predicted primary account number, a predicted amount, a predicted resource provider identifier, etc.).

5 **[0102]** In FIG. 6, a computer can determine intermediate embeddings 604 for a predicted interaction data set embedding. The computer can determine a number of intermediate embeddings 604 equal to the number of tokens in the original corresponding interaction data set (e.g., 8).

10 **[0103]** The computer can utilize a machine learning model to determine each intermediate embedding from the predicted interaction data set embedding. In particular, the computer can concatenate the predicted interaction data set embedding with a positional encoding vector to identify which token position of the interaction data set is to be determined by the machine learning model. The computer can generate the intermediate embedding that corresponds to the token position from the positionally encoded predicted interaction data set embedding. The computer can
15 iterate through a plurality of positional encodings, one per number of tokens in the interaction data set, to determine the intermediate embeddings 604.

20 **[0104]** The computer can input the intermediate embeddings 604 into one or more transformer blocks 606 (e.g., transformer decoder blocks) to obtain output embeddings 608. Each output embedding can correspond to a token. The one or more transformer blocks 606 can transform the input intermediate embeddings 604 into output embeddings 608. The computer can utilize the one or more transformer blocks 606 to convert the embeddings from a transformed embedding space to an embedding space, such that the predicted tokens can be determined from the embeddings in the embedding space.

25 **[0105]** After determining the output embeddings 608, the computer can combine each output embedding with a positional encoding (e.g., via concatenation) to form positionally encoded output embeddings. The computer can then determine a predicted token for each output embedding.

30 **[0106]** The computer can determine a predicted token from an output embedding using a machine learning model (e.g., a multi-layer perceptron). The computer can iteratively input each output embedding into the machine learning model

to output a predicted token. Different types of tokens (e.g., repeating tokens, non-repeating tokens, partially repeating tokens) can utilize a different final layer in the machine learning model. The final layer can be a final linear projection layer or a final SoftMax layer.

- 5 **[0107]** A use case for SoftMax in the output layer can include a classification problem, where the output is an array of probabilities for each class. A use case for a linear projection in the output layer can include a regression problem, where the output is an array of floating point numbers that are estimates for some measurement.

- 10 **[0108]** The computer can determine the non-repeating tokens using a machine learning model that includes a final linear projection layer. The computer can determine the repeating tokens using a machine learning model that includes a final SoftMax layer. The computer can determine the partially repeating tokens using a machine learning model that includes both a SoftMax layer and a linear projection layer to determine the different portions of the partially repeating tokens.

- 15 **[0109]** The computer can select the machine learning model for each output embedding from a first machine learning model and a second machine learning model based on the positional encoding, where the first machine learning model includes the final linear projection layer, and where the second machine learning model includes the final SoftMax layer.

- 20 **[0110]** The predicted token can be a predicted value for the predicted interaction. For example, predicted tokens include a predicted amount, a predicted resource provider identifier, a predicted primary account number, a predicted timestamp, and a predicted resource provider category code, etc.

- 25 **[0111]** Embodiments of the disclosure have a number of advantages. For example, embodiments reduce the amount of computation needed due to efficient caching of repeating data token embeddings that repeat between interaction data sets. Embeddings can be pre-cached and retrieved rather than be re-generated for every interaction. Previously, transformers did not account for repeating tokens since typically transforms are used for language processing and therefore do not typically
30 encounter situations in which tokens repeat across inputs. Embodiments provide for generating a two dimensional data structures of interaction data that accounts for the

repetition of data tokens for the interaction and the interactions over time that can be utilized in transformer architectures. The system can generate predicted interactions using the two dimensional data structure.

5 [0112] 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 or added and may still be within embodiments of the invention.

10 [0113] 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, suitable media 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 15 medium such as a compact disk (CD) or DVD (digital versatile disk), flash memory, and the like. The computer readable medium may be any combination of such storage or transmission devices.

20 [0114] 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 according to an embodiment of the present invention 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 or 25 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 30 mentioned herein to a user.

[0115] The above description is illustrative and is not restrictive. Many variations of the invention will become apparent to those skilled in the art upon review

of the disclosure. The scope of the invention should, therefore, be determined not with reference to the above description, but instead should be determined with reference to the pending claims along with their full scope or equivalents.

5 **[0116]** One or more features from any embodiment may be combined with one or more features of any other embodiment without departing from the scope of the invention.

[0117] As used herein, the use of "a," "an," or "the" is intended to mean "at least one," unless specifically indicated to the contrary.

WHAT IS CLAIMED IS:

- 1 1. A method comprising:
2 obtaining, by a computer, interaction data comprising a plurality of
3 interaction data sets, each interaction data set comprising a plurality of tokens,
4 wherein the plurality of tokens include non-repeatable tokens and repeatable tokens;
5 for each interaction data set:
6 encoding, by the computer, the non-repeatable tokens of the
7 plurality of tokens to form non-repeatable token embeddings,
8 retrieving, by the computer, repeatable token embeddings from
9 memory using the repeatable tokens, wherein the repeatable token
10 embeddings and the non-repeatable token embeddings are in a plurality of
11 first embeddings,
12 concatenating, by the computer, the plurality of first embeddings
13 with positional encodings to form second embeddings,
14 generating, by the computer, an interaction data set embedding
15 based on the second embeddings;
16 determining, by the computer, one or more predicted interaction data
17 set embeddings using interaction data set embeddings comprising the interaction
18 data set embedding for each interaction data set; and
19 decoding, by the computer, the one or more predicted interaction data
20 set embeddings to predicted tokens.
- 1 2. The method of claim 1, wherein the plurality of tokens include
2 partially repeatable tokens.
- 1 3. The method of claim 2, wherein the non-repeatable tokens
2 include a timestamp and a security value.
- 1 4. The method of claim 2, wherein the repeatable tokens include a
2 primary account number, an issuer identifier, an acquirer identifier, a zip code, a
3 resource provider category code, and a resource provider identifier.
- 1 5. The method of claim 2, wherein the partially repeatable tokens
2 include a combination of a resource provider identifier and an amount.

1 6. The method of claim 1, wherein determining one or more
2 predicted interaction data set embeddings comprises:
3 determining, by the computer, the one or more predicted interaction
4 data set embeddings using a transformer.

1 7. The method of claim 1, wherein encoding the non-repeatable
2 tokens comprises:
3 for each non-repeatable token of the non-repeatable tokens,
4 processing, by the computer, the non-repeatable token using a machine learning
5 model to determine a non-repeatable token embedding corresponding to the non-
6 repeatable token.

1 8. The method of claim 1, wherein the plurality of tokens include
2 partially repeatable tokens, wherein the method further comprises:
3 for each interaction data set:
4 separating, by the computer, each partially repeatable token of
5 the partially repeatable tokens into a repeatable token portion and a non-
6 repeatable token portion;
7 encoding, by the computer, the non-repeatable token portion for
8 each partially repeatable token to form a non-repeatable token portion
9 embedding for each partially repeatable token using a first machine learning
10 model;
11 retrieving, by the computer, a repeatable token portion
12 embedding for each repeatable token portion from the memory using the
13 repeatable token portion; and
14 generating, by the computer, partially repeatable token
15 embeddings for each partially repeatable token based on the non-repeatable
16 token portion embedding and the repeatable token portion embedding using a
17 second machine learning model, wherein the partially repeatable token
18 embeddings are in the plurality of first embeddings.

1 9. The method of claim 1, wherein generating the interaction data
2 set embedding comprises:

3 generating, by the computer, the interaction data set embedding based
4 on the second embeddings using one or more transformer blocks.

1 10. The method of claim 1, wherein decoding the one or more
2 predicted interaction data set embeddings comprises:
3 for each predicted interaction data set embedding of the one or more
4 predicted interaction data set embeddings, generating, by the computer,
5 intermediate embeddings;
6 for each predicted interaction data set embedding, determining, by the
7 computer, output embeddings based on the intermediate embeddings using one or
8 more transformer blocks;
9 for each predicted interaction data set embedding, for each output
10 embedding, concatenating, by the computer, the output embedding with a positional
11 encoding to form a positionally encoded output embedding; and
12 for each predicted interaction data set embedding, for each positionally
13 encoded output embedding, determining, by the computer, a predicted token based
14 on the positionally encoded output embedding using a machine learning model.

1 11. The method of claim 1, wherein the predicted tokens include a
2 predicted amount, a predicted resource provider identifier, and a predicted primary
3 account number.

1 12. A computer comprising:
2 a processor; and
3 a computer-readable medium coupled to the processor, the computer-
4 readable medium comprising code executable by the processor for implementing a
5 method comprising:
6 obtaining interaction data comprising a plurality of interaction
7 data sets, each interaction data set comprising a plurality of tokens;
8 for each interaction data set:
9 encoding the plurality of tokens to obtain a plurality of first
10 embeddings,
11 concatenating the plurality of first embeddings with
12 positional encodings to form second embeddings,

13 generating an interaction data set embedding based on
14 the second embeddings;
15 determining one or more predicted interaction data set
16 embeddings using interaction data set embeddings comprising the interaction
17 data set embedding for each interaction data set; and
18 decoding the one or more predicted interaction data set
19 embeddings to predicted tokens.

1 13. The computer of claim 12, wherein the plurality of tokens include
2 partially repeatable tokens, wherein the method further comprises:
3 for each interaction data set:
4 separating each partially repeatable token of the partially
5 repeatable tokens into a repeatable token portion and a non-repeatable token
6 portion;
7 encoding the non-repeatable token portion for each partially
8 repeatable token to form a non-repeatable token portion embedding for each
9 partially repeatable token using a first machine learning model;
10 retrieving a repeatable token portion embedding for each
11 repeatable token portion from memory using the repeatable token portion; and
12 generating partially repeatable token embeddings for each
13 partially repeatable token based on the non-repeatable token portion
14 embedding and the repeatable token portion embedding using a second
15 machine learning model, wherein the partially repeatable token embeddings
16 are in the plurality of first embeddings.

1 14. The computer of claim 12, wherein the predicted tokens include
2 a predicted amount, a predicted resource provider identifier, a predicted primary
3 account number, a predicted timestamp, and a predicted resource provider category
4 code.

1 15. The computer of claim 12, wherein decoding the one or more
2 predicted interaction data set embeddings comprises:
3 for each predicted interaction data set embedding of the one or more
4 predicted interaction data set embeddings, generating intermediate embeddings;

5 for each predicted interaction data set embedding, determining output
6 embeddings based on the intermediate embeddings using one or more transformer
7 blocks;

8 for each predicted interaction data set embedding, for each output
9 embedding, concatenating the output embedding with a positional encoding to form
10 a positionally encoded output embedding; and

11 for each predicted interaction data set embedding, for each positionally
12 encoded output embedding, determining a predicted token based on the positionally
13 encoded output embedding using a machine learning model.

1 16. The computer of claim 15, wherein the method further
2 comprises:

3 for each predicted interaction data set embedding, for each positionally
4 encoded output embedding, selecting the machine learning model from a first
5 machine learning model and a second machine learning model based on the
6 positional encoding.

1 17. The computer of claim 16, wherein the first machine learning
2 model includes a final linear projection layer, and wherein the second machine
3 learning model includes a final SoftMax layer.

1 18. A system comprising:

2 a database storing interaction data stored by a network processing
3 computer; and

4 an analysis computer comprising:

5 a processor; and

6 a computer-readable medium coupled to the processor, the
7 computer-readable medium comprising code executable by the processor for
8 implementing a method comprising:

9 obtaining, by a computer, interaction data comprising a
10 plurality of interaction data sets, each interaction data set comprising a
11 plurality of tokens;

12 for each interaction data set:

13 encoding, by the computer, the plurality of tokens
14 to obtain a plurality of first embeddings,
15 concatenating, by the computer, the plurality of
16 first embeddings with positional encodings to form second
17 embeddings,
18 generating, by the computer, an interaction data
19 set embedding based on the second embeddings;
20 determining, by the computer, one or more predicted
21 interaction data set embeddings using interaction data set embeddings
22 comprising the interaction data set embedding for each interaction data
23 set; and
24 decoding, by the computer, the one or more predicted
25 interaction data set embeddings to predicted tokens.

1 19. The system of claim 18, wherein the plurality of tokens include
2 non-repeatable tokens, repeatable tokens, and partially repeatable tokens, wherein
3 the non-repeatable tokens include a timestamp and a security value, wherein the
4 repeatable tokens include a primary account number, an issuer identifier, an acquirer
5 identifier, a zip code, a resource provider category code, and a resource provider
6 identifier, and wherein the partially repeatable tokens include a combination of the
7 resource provider identifier and an amount.

1 20. The system of claim 18, wherein the database storing the
2 interaction data comprises one hundred or more entities of interaction data sets.