

**METHOD, SYSTEM, AND COMPUTER PROGRAM PRODUCT FOR
GENERATING ROBUST GRAPH NEURAL NETWORKS USING UNIVERSAL
ADVERSARIAL TRAINING**

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to U.S. Provisional Patent Application No. 63/311,620, filed February 18, 2022, the disclosure of which is incorporated by reference herein in its entirety.

BACKGROUND

1. Technical Field

[0002] This disclosure relates generally to adversarial training for neural networks and, in some non-limiting embodiments or aspects, to methods, systems, and computer program products for generating robust graph neural networks that are resilient to perturbations by using universal adversarial training.

2. Technical Considerations

[0003] Graph neural networks (GNNs) provide a class of powerful architectures that are extremely effective for graph-based collaborative filtering. Nevertheless, GNNs are vulnerable to perturbations (*e.g.*, additions, modifications, deletions, missing edges, erroneously included edges, etc.) on nodes' features or graph structures (*e.g.*, edgewise connections to other nodes). Perturbations can catastrophically reduce the performance (*e.g.*, accuracy, efficiency, etc.) of GNNs. Underperforming GNNs, caused by perturbations, may require additional layers of training, require combination with other neural networks to provide ensemble analysis, or require additional post-generation testing in order to provide the level of performance required in a given system implementation. Each of the above remedies increase computer system requirements (*e.g.*, processing capacity, required memory storage, processing time, bandwidth, etc.) and could be avoided if the underlying GNN was generated to be robust and resilient to perturbations.

[0004] Accordingly, there is a need in the art for a technical solution that provides for generating robust GNNs that are resilient to perturbations, such that computer resources required with producing high-performing GNNs can be minimized.

SUMMARY

[0005] According to some non-limiting embodiments or aspects, provided are methods, systems, and computer program products for generating robust GNNs using universal adversarial training that overcome some or all of the deficiencies identified above.

[0006] According to some non-limiting embodiments or aspects, provided is a computer-implemented method for generating robust GNNs using universal adversarial training. The method includes receiving, with at least one processor, input including a GNN model and a bipartite graph including an adjacency matrix determined from an interaction matrix. The method also includes initializing, with at least one processor, model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss. The method further includes initializing, with at least one processor, perturbation parameters randomly. The method further includes determining, with at least one processor, a complementary graph based on the bipartite graph. The method further includes sampling, with at least one processor, a subgraph of the complementary graph. The method further includes repeating, with at least one processor, until convergence of the model parameters: drawing a random variable from a uniform distribution; generating a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determining BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; updating the perturbation parameters based on stochastic gradient ascent; and updating the model parameters based on stochastic gradient descent. The method further includes, in response to convergence of the model parameters, outputting, with at least one processor, the model parameters.

[0007] In some non-limiting embodiments or aspects, the method may further include, in response to convergence of the model parameters, generating, with at least one processor, the at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0008] In some non-limiting embodiments or aspects, the input may further include a perturbations budget, wherein the perturbations budget specifies a number of edges of the subgraph.

[0009] In some non-limiting embodiments or aspects, the input may further include a first regularized parameter, wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

[0010] In some non-limiting embodiments or aspects, the input may further include a second regularized parameter, wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

[0011] In some non-limiting embodiments or aspects, the input may further include a third regularized parameter, wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0012] In some non-limiting embodiments or aspects, the interaction matrix may be a user-item interaction matrix, wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0013] In some non-limiting embodiments or aspects, the generation of the universal perturbation matrix may be further based on a sigmoid function of the perturbation parameters.

[0014] In some non-limiting embodiments or aspects, the update of the model parameters based on stochastic gradient descent may be further based at least partly on BPR loss.

[0015] According to some non-limiting embodiments or aspects, provided is a system for generating robust GNNs using universal adversarial training. The system includes at least one processor that is programmed or configured to receive input including a GNN model and a bipartite graph including an adjacency matrix determined from an interaction matrix. The at least one processor is also programmed or configured to initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss. The at least one processor is further programmed or configured to initialize perturbation parameters randomly. The at least one processor is further programmed or configured to determine a complementary graph based on the bipartite graph. The at least one processor is further programmed or configured to sample a subgraph of the complementary graph. The at least one processor is further programmed or configured to repeat until convergence of the model parameters: draw a random variable from a uniform distribution; generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; update the perturbation parameters

based on stochastic gradient ascent; and update the model parameters based on stochastic gradient descent. The at least one processor is further programmed or configured to, in response to convergence of the model parameters, output the model parameters.

[0016] In some non-limiting embodiments or aspects, the at least one processor is further programmed or configured to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0017] In some non-limiting embodiments or aspects, the input may further include a perturbations budget, and the perturbations budget may specify a number of edges of the subgraph.

[0018] In some non-limiting embodiments or aspects, the input may further include a first regularized parameter, and the BPR loss may be based on the first regularized parameter to prevent overfitting.

[0019] In some non-limiting embodiments or aspects, the input may further include a second regularized parameter, and an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss may be limited by the second regularized parameter.

[0020] In some non-limiting embodiments or aspects, the input may further include a third regularized parameter, and the universal perturbation matrix may be generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0021] In some non-limiting embodiments or aspects, the interaction matrix may include a user-item interaction matrix, and the GNN model may be trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0022] In some non-limiting embodiments or aspects, the generation of the universal perturbation matrix may be further based on a sigmoid function of the perturbation parameters.

[0023] In some non-limiting embodiments or aspects, the update of the model parameters based on stochastic gradient descent may be further based at least partly on BPR loss.

[0024] According to some non-limiting embodiments or aspects, provided is a computer program product for generating robust GNNs using universal adversarial

training. The computer program product includes at least one non-transitory computer-readable medium including one or more instructions that, when executed by at least one processor, cause the at least one processor to receive input including a GNN model and a bipartite graph including an adjacency matrix determined from an interaction matrix. The one or more instructions also cause the at least one processor to initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss. The one or more instructions further cause the at least one processor to initialize perturbation parameters randomly. The one or more instructions further cause the at least one processor to determine a complementary graph based on the bipartite graph. The one or more instructions further cause the at least one processor to sample a subgraph of the complementary graph. The one or more instructions further cause the at least one processor to repeat until convergence of the model parameters: draw a random variable from a uniform distribution; generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; update the perturbation parameters based on stochastic gradient ascent; and update the model parameters based on stochastic gradient descent. The one or more instructions further cause the at least one processor to, in response to convergence of the model parameters, output the model parameters.

[0025] In some non-limiting embodiments or aspects, the one or more instructions may cause the at least one processor to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0026] In some non-limiting embodiments or aspects, the input may further include a perturbations budget, and the perturbations budget may specify a number of edges of the subgraph.

[0027] In some non-limiting embodiments or aspects, the input may further include a first regularized parameter, and the BPR loss may be based on the first regularized parameter to prevent overfitting.

[0028] In some non-limiting embodiments or aspects, the input may further include a second regularized parameter, and an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss may be limited by the second regularized parameter.

[0029] In some non-limiting embodiments or aspects, the input may further include a third regularized parameter, and the universal perturbation matrix may be generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0030] In some non-limiting embodiments or aspects, the interaction matrix may include a user-item interaction matrix, and the GNN model may be trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0031] In some non-limiting embodiments or aspects, the generation of the universal perturbation matrix may be further based on a sigmoid function of the perturbation parameters.

[0032] In some non-limiting embodiments or aspects, the update of the model parameters based on stochastic gradient descent may be further based at least partly on BPR loss.

[0033] Other non-limiting embodiments or aspects will be set forth in the following numbered clauses:

[0034] Clause 1: A computer-implemented method comprising: receiving, with at least one processor, input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix; initializing, with at least one processor, model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss; initializing, with at least one processor, perturbation parameters randomly; determining, with at least one processor, a complementary graph based on the bipartite graph; sampling, with at least one processor, a subgraph of the complementary graph; repeating, with at least one processor, until convergence of the model parameters: drawing a random variable from a uniform distribution; generating a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determining BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; updating the perturbation parameters based on stochastic gradient ascent; and updating the model parameters based on stochastic gradient descent; and, in response to convergence of the model parameters, outputting, with at least one processor, the model parameters.

[0035] Clause 2: The computer-implemented method of clause 1, further comprising, in response to convergence of the model parameters, generating, with at

least one processor, at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0036] Clause 3: The computer-implemented method of clause 1 or clause 2, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.

[0037] Clause 4: The computer-implemented method of any of clauses 1-3, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

[0038] Clause 5: The computer-implemented method of any of clauses 1-4, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

[0039] Clause 6: The computer-implemented method of any of clauses 1-5, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0040] Clause 7: The computer-implemented method of any of clauses 1-6, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0041] Clause 8: The computer-implemented method of any of clauses 1-7, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.

[0042] Clause 9: The computer-implemented method of any of clauses 1-8, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.

[0043] Clause 10: A system comprising at least one processor, the at least one processor being programmed or configured to: receive input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix; initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss; initialize perturbation parameters randomly; determine a complementary graph based on the bipartite graph; sample a subgraph of the complementary graph; repeat until convergence of the model parameters: draw a random variable from a uniform distribution; generate a

universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; update the perturbation parameters based on stochastic gradient ascent; and update the model parameters based on stochastic gradient descent; and, in response to convergence of the model parameters, output the model parameters.

[0044] Clause 11: The system of clause 10, wherein the at least one processor is further programmed or configured to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0045] Clause 12: The system of clause 10 or clause 11, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.

[0046] Clause 13: The system of any of clauses 10-12, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

[0047] Clause 14: The system of any of clauses 10-13, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

[0048] Clause 15: The system of any of clauses 10-14, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0049] Clause 16: The system of any of clauses 10-15, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0050] Clause 17: The system of any of clauses 10-16, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.

[0051] Clause 18: The system of any of clauses 10-17, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.

[0052] Clause 19: A computer program product comprising at least one non-transitory computer-readable medium including one or more instructions that, when executed by at least one processor, cause the at least one processor to: receive input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix; initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss; initialize perturbation parameters randomly; determine a complementary graph based on the bipartite graph; sample a subgraph of the complementary graph; repeat until convergence of the model parameters: draw a random variable from a uniform distribution; generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters; determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model; update the perturbation parameters based on stochastic gradient ascent; and update the model parameters based on stochastic gradient descent; and, in response to convergence of the model parameters, output the model parameters.

[0053] Clause 20: The computer program product of clause 19, wherein the one or more instructions cause the at least one processor to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0054] Clause 21: The computer program product of clause 19 or clause 20, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.

[0055] Clause 22: The computer program product of any of clauses 19-21, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

[0056] Clause 23: The computer program product of any of clauses 19-22, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

[0057] Clause 24: The computer program product of any of clauses 19-23, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

[0058] Clause 25: The computer program product of any of clauses 19-24, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0059] Clause 26: The computer program product of any of clauses 19-25, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.

[0060] Clause 27: The computer program product of any of clauses 19-26, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.

[0061] These and other features and characteristics of the present disclosure, as well as the methods of operation and functions of the related elements of structures and the combination of parts and economies of manufacture, will become more apparent upon consideration of the following description and the appended claims with reference to the accompanying drawings, all of which form a part of this specification, wherein like reference numerals designate corresponding parts in the various figures. It is to be expressly understood, however, that the drawings are for the purpose of illustration and description only and are not intended as a definition of the limits of the present disclosure. As used in the specification and the claims, the singular form of "a," "an," and "the" include plural referents unless the context clearly dictates otherwise.

BRIEF DESCRIPTION OF THE DRAWINGS

[0062] Additional advantages and details of the disclosure are explained in greater detail below with reference to the exemplary embodiments that are illustrated in the accompanying figures, in which:

[0063] FIG. 1 is a schematic diagram of a system for generating robust GNNs using universal adversarial training, according to some non-limiting embodiments or aspects;

[0064] FIG. 2 is a diagram of one or more components, devices, and/or systems, according to some non-limiting embodiments or aspects;

[0065] FIG. 3 is a flow diagram of a method for generating robust GNNs using universal adversarial training, according to some non-limiting embodiments or aspects; and

[0066] FIG. 4 is a flow diagram of a method for generating robust GNNs using universal adversarial training, according to some non-limiting embodiments or aspects.

[0067] It should be appreciated by those skilled in the art that any block diagrams herein represent conceptual views of illustrative systems embodying the principles of the present subject matter. Similarly, it may be appreciated that any flow charts, flow diagrams, state transition diagrams, pseudo code, and the like represent various processes which may be substantially represented in computer readable medium and executed by a computer or processor, whether or not such computer or processor is explicitly shown.

DETAILED DESCRIPTION

[0068] For purposes of the description hereinafter, the terms “upper”, “lower”, “right”, “left”, “vertical”, “horizontal”, “top”, “bottom”, “lateral”, “longitudinal,” and derivatives thereof shall relate to non-limiting embodiments or aspects as they are oriented in the drawing figures. However, it is to be understood that non-limiting embodiments or aspects may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings, and described in the following specification, are simply exemplary embodiments or aspects. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0069] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly described as such. Also, as used herein, the articles “a” and “an” are intended to include one or more items and may be used interchangeably with “one or more” and “at least one.” Furthermore, as used herein, the term “set” is intended to include one or more items (*e.g.*, related items, unrelated items, a combination of related and unrelated items, etc.) and may be used interchangeably with “one or more” or “at least one.” Where only one item is intended, the term “one” or similar language is used. Also, as used herein, the terms “has,” “have,” “having,” or the like are intended to be open-ended terms. Further, the phrase “based on” is intended to mean “based at least partially on” unless explicitly stated otherwise.

[0070] Some non-limiting embodiments or aspects are described herein in connection with thresholds. As used herein, satisfying a threshold may refer to a value being greater than the threshold, more than the threshold, higher than the threshold, greater than or equal to the threshold, less than the threshold, fewer than the threshold, lower than the threshold, less than or equal to the threshold, equal to the threshold, and/or the like.

[0071] As used herein, the term “acquirer institution” may refer to an entity licensed and/or approved by a transaction service provider to originate transactions (*e.g.*, payment transactions) using a payment device associated with the transaction service provider. The transactions the acquirer institution may originate may include payment transactions (*e.g.*, purchases, original credit transactions (OCTs), account funding transactions (AFTs), and/or the like). In some non-limiting embodiments or aspects, an acquirer institution may be a financial institution, such as a bank. As used herein, the term “acquirer system” may refer to one or more computing devices operated by or on behalf of an acquirer institution, such as a server computer executing one or more software applications.

[0072] As used herein, the term “account identifier” may include one or more primary account numbers (PANs), tokens, or other identifiers associated with a customer account. The term “token” may refer to an identifier that is used as a substitute or replacement identifier for an original account identifier, such as a PAN. Account identifiers may be alphanumeric or any combination of characters and/or symbols. Tokens may be associated with a PAN or other original account identifier in one or more data structures (*e.g.*, one or more databases, and/or the like) such that they may be used to conduct a transaction without directly using the original account identifier. In some examples, an original account identifier, such as a PAN, may be associated with a plurality of tokens for different individuals or purposes.

[0073] As used herein, the term “communication” may refer to the reception, receipt, transmission, transfer, provision, and/or the like, of data (*e.g.*, information, signals, messages, instructions, commands, and/or the like). For one unit (*e.g.*, a device, a system, a component of a device or system, combinations thereof, and/or the like) to be in communication with another unit means that the one unit is able to directly or indirectly receive information from and/or transmit information to the other unit. This may refer to a direct or indirect connection (*e.g.*, a direct communication connection, an indirect communication connection, and/or the like) that is wired and/or

wireless in nature. Additionally, two units may be in communication with each other even though the information transmitted may be modified, processed, relayed, and/or routed between the first and second unit. For example, a first unit may be in communication with a second unit even though the first unit passively receives information and does not actively transmit information to the second unit. As another example, a first unit may be in communication with a second unit if at least one intermediary unit processes information received from the first unit and communicates the processed information to the second unit.

[0074] As used herein, the term “computing device” may refer to one or more electronic devices configured to process data. A computing device may, in some examples, include the necessary components to receive, process, and output data, such as a processor, a display, a memory, an input device, a network interface, and/or the like. A computing device may be a mobile device. As an example, a mobile device may include a cellular phone (*e.g.*, a smartphone or standard cellular phone), a portable computer, a wearable device (*e.g.*, watches, glasses, lenses, clothing, and/or the like), a personal digital assistant (PDA), and/or other like devices. A computing device may also be a desktop computer or other form of non-mobile computer. An “application” or “application program interface” (API) may refer to computer code or other data stored on a computer-readable medium that may be executed by a processor to facilitate the interaction between software components, such as a client-side front-end and/or server-side back-end for receiving data from the client. An “interface” may refer to a generated display, such as one or more graphical user interfaces (GUIs) with which a user may interact, either directly or indirectly (*e.g.*, through a keyboard, mouse, etc.).

[0075] As used herein, the terms “electronic wallet” and “electronic wallet application” refer to one or more electronic devices and/or software applications configured to initiate and/or conduct payment transactions. For example, an electronic wallet may include a mobile device executing an electronic wallet application, and may further include server-side software and/or databases for maintaining and providing transaction data to the mobile device. An “electronic wallet provider” may include an entity that provides and/or maintains an electronic wallet for a customer, such as Google Pay®, Android Pay®, Apple Pay®, Samsung Pay®, and/or other like electronic payment systems. In some non-limiting examples, an issuer bank may be an electronic wallet provider.

[0076] As used herein, the term “issuer institution” may refer to one or more entities, such as a bank, that provide accounts to customers for conducting transactions (*e.g.*, payment transactions), such as initiating credit and/or debit payments. For example, an issuer institution may provide an account identifier, such as a PAN, to a customer that uniquely identifies one or more accounts associated with that customer. The account identifier may be embodied on a portable financial device, such as a physical financial instrument, *e.g.*, a payment card, and/or may be electronic and used for electronic payments. The term “issuer system” refers to one or more computer devices operated by or on behalf of an issuer institution, such as a server computer executing one or more software applications. For example, an issuer system may include one or more authorization servers for authorizing a transaction.

[0077] As used herein, the term “merchant” may refer to an individual or entity that provides goods and/or services, or access to goods and/or services, to customers based on a transaction, such as a payment transaction. The term “merchant” or “merchant system” may also refer to one or more computer systems operated by or on behalf of a merchant, such as a server computer executing one or more software applications. A “point-of-sale (POS) system,” as used herein, may refer to one or more computers and/or peripheral devices used by a merchant to engage in payment transactions with customers, including one or more card readers, scanning devices (*e.g.*, code scanners), Bluetooth® communication receivers, near-field communication (NFC) receivers, radio frequency identification (RFID) receivers, and/or other contactless transceivers or receivers, contact-based receivers, payment terminals, computers, servers, input devices, and/or other like devices that can be used to initiate a payment transaction.

[0078] As used herein, the term “payment device” may refer to a payment card (*e.g.*, a credit or debit card), a gift card, a smartcard, smart media, a payroll card, a healthcare card, a wristband, a machine-readable medium containing account information, a keychain device or fob, an RFID transponder, a retailer discount or loyalty card, a cellular phone, an electronic wallet mobile application, a PDA, a pager, a security card, a computing device, an access card, a wireless terminal, a transponder, and/or the like. In some non-limiting embodiments or aspects, the payment device may include volatile or non-volatile memory to store information (*e.g.*, an account identifier, a name of the account holder, and/or the like).

[0079] As used herein, the term “payment gateway” may refer to an entity and/or a payment processing system operated by or on behalf of such an entity (e.g., a merchant service provider, a payment service provider, a payment facilitator, a payment facilitator that contracts with an acquirer, a payment aggregator, and/or the like), which provides payment services (e.g., transaction service provider payment services, payment processing services, and/or the like) to one or more merchants. The payment services may be associated with the use of portable financial devices managed by a transaction service provider. As used herein, the term “payment gateway system” may refer to one or more computer systems, computer devices, servers, groups of servers, and/or the like, operated by or on behalf of a payment gateway.

[0080] As used herein, the term “server” may refer to or include one or more computing devices that are operated by or facilitate communication and processing for multiple parties in a network environment, such as the Internet, although it will be appreciated that communication may be facilitated over one or more public or private network environments and that various other arrangements are possible. Further, multiple computing devices (e.g., servers, POS devices, mobile devices, etc.) directly or indirectly communicating in the network environment may constitute a “system.” Reference to “a server” or “a processor,” as used herein, may refer to a previously-recited server and/or processor that is recited as performing a previous step or function, a different server and/or processor, and/or a combination of servers and/or processors. For example, as used in the specification and the claims, a first server and/or a first processor that is recited as performing a first step or function may refer to the same or different server and/or a processor recited as performing a second step or function.

[0081] As used herein, the term “transaction service provider” may refer to an entity that receives transaction authorization requests from merchants or other entities and provides guarantees of payment, in some cases through an agreement between the transaction service provider and an issuer institution. For example, a transaction service provider may include a payment network such as Visa® or any other entity that processes transactions. The term “transaction processing system” may refer to one or more computer systems operated by or on behalf of a transaction service provider, such as a transaction processing server executing one or more software applications. A transaction processing server may include one or more processors

and, in some non-limiting embodiments or aspects, may be operated by or on behalf of a transaction service provider.

[0082] Non-limiting embodiments or aspects of the disclosed subject matter are directed to methods, systems, and computer program products for generating robust GNNs including, but not limited to, using universal adversarial training. For example, non-limiting embodiments or aspects of the disclosed subject matter provide receiving input including a GNN model and a bipartite graph including an adjacency matrix determined from an interaction matrix, initializing model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss, initializing perturbation parameters randomly, determining a complementary graph based on the bipartite graph, sampling a subgraph of the complementary graph, and, until convergence of the model parameters, repeatedly drawing a random variable from a uniform distribution, generating a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters, determining BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model, updating the perturbation parameters based on stochastic gradient ascent, and updating the model parameters based on stochastic gradient descent. Such embodiments or aspects provide numerous technical advantages and improvements in systems for generating robust GNNs using universal adversarial training. For example, such techniques provide for improving the performance (*e.g.*, accuracy, efficiency, etc.) of GNN model training, testing, and execution. Additionally, such techniques reduce computer system requirements (*e.g.*, processing capacity, required memory storage, processing time, bandwidth, etc.) that are required to produce the same quality of performance of GNNs as those of prior art systems. Moreover, by employing adversarial training using the method described herein (*e.g.*, perturbations based on the perturbation matrix and/or perturbation parameters, as iteratively updated), the GNN model may be made to be resilient to perturbations without the need to employ additional computer resources to account for perturbations after GNN model generation and/or training. Furthermore, since perturbations (*e.g.*, additions, modifications, deletions, etc. of graph edges) may be caused by data collection errors, data storage errors, data processing errors, malicious attacks, and/or the like, the techniques described herein improve the overall performance of computer-implemented GNN systems by producing a robust GNN model that is resilient to such technical defects. GNN models produced according to non-limiting embodiments and

aspects of the present disclosure will outperform GNN models produced without the described training techniques, when in the presence of the same kinds of perturbations.

[0083] GNNs, such as graph convolutional networks (GCNs), may be powerful for collaborative filtering. For example, GCNs may be used to explore neighborhood aggregation mechanisms to extract high-level representations of users and items. Real-world user-item graphs may be incomplete and noisy, which may lead to suboptimal performance and misleading neighborhood information. In some non-limiting embodiments or aspects, the systems and methods described herein may use adversarial training to properly regularize GCNs and overcome such problems with incomplete and noisy user-item graphs.

[0084] GNNs, without such adversarial training, may be vulnerable to adversarial perturbations, in which small, unnoticeable perturbations on nodes' features or graph structures may significantly reduce the performance of the GNN. In some non-limiting embodiments or aspects, the systems and methods described herein may model adversarial graph perturbations during the training stage of GNNs to improve model performance. Such systems and methods may use min-max robust optimization, in which a universal graph perturbation matrix is obtained through an inner maximization problem, and an outer optimization aims to compute the model parameters of a GNN model. Furthermore, such systems and methods address the challenges of optimizing the inner maximization problem (due to the discrete nature of graph perturbations) by constructing a continuous relaxation of the discrete optimization, and further using an unbiased gradient estimator to compute the gradient of discrete variables. These techniques enhance the generalization performance of GNN-based recommender systems. Moreover, the described systems and methods are compatible with GNN models without changing their architectures.

[0085] Referring now to FIG. 1, a system 100 for generating robust GNNs using universal adversarial training is shown, according to some non-limiting embodiments or aspects. Specifically, the system 100 may include modeling system 102, user device 106, memory 110, and a communication network 108.

[0086] Modeling system 102 may include one or more computing devices configured to communicate with user device 106 and/or memory 110, including at least partly over the communication network 108. In some non-limiting embodiments or aspects, modeling system 102 may be associated with or included in a transaction

processing system. Modeling system 102 may further be associated with or include memory 110. Modeling system 102 may further be associated with or include user device 106. Modeling system 102 may include one or more servers. Modeling system 102 may include at least one processor and at least one non-transitory computer readable storage medium including program instructions that, when executed, cause the at least one processor to perform the methods of the present disclosure.

[0087] User device 106 may include one or more computing devices configured to communicate with modeling system 102 and/or memory 110, including at least partly over the communication network 108. User device 106 may include at least one processor and at least one non-transitory computer readable storage medium including program instructions that, when executed, cause the at least one processor to transmit/select input and/or receive/view output of the methods of the present disclosure.

[0088] Memory 110 may include one or more computing devices configured to communicate with modeling system 102 and/or user device 106, including at least partly over the communication network 108. Memory 110 may include at least one processor and at least one non-transitory computer readable storage medium including program instructions that, when executed, cause the at least one processor to store and/or communicate GNN model data, interaction matrices, GNN model recommendations / categorizations, GNN model parameters, and/or the like.

[0089] Communication network 108 may include one or more wired and/or wireless networks. For example, communication network 108 may include a cellular network (*e.g.*, a long-term evolution (LTE®) network, a third generation (3G) network, a fourth generation (4G) network, a fifth generation (5G) network, a code division multiple access (CDMA) network, and/or the like), a public land mobile network (PLMN), a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), a telephone network (*e.g.*, the public switched telephone network (PSTN)), a private network, an ad hoc network, a mesh network, a beacon network, an intranet, the Internet, a fiber optic-based network, a cloud computing network, and/or the like, and/or a combination of these or other types of networks.

[0090] In some non-limiting embodiments or aspects, modeling system 102 may perform one or more steps of a method for generating robust GNNs using universal adversarial training. For example, modeling system 102 may receive input including a GNN model and a bipartite graph including an adjacency matrix determined from an

interaction matrix. The input may also include a perturbations budget that specifies a number of edges of the subgraph. The input may further include a first regularized parameter, a second regularized parameter, and/or a third regularized parameter. Modeling system 102 may also initialize model parameters of the GNN model based on a BPR loss. Modeling system 102 may further initialize perturbation parameters randomly and determine a complementary graph based on the bipartite graph. Modeling system 102 may further sample a subgraph of the complementary graph.

[0091] In some non-limiting embodiments or aspects, modeling system 102 may repeat a process until convergence of the model parameters. In the repeated process, modeling system 102 may draw a random variable from a uniform distribution, generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters, determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model, update the perturbation parameters based on stochastic gradient ascent, and update the model parameters based on stochastic gradient descent, which may be further based at least partly on BPR loss. In response to convergence of the model parameters, modeling system 102 may output the model parameters. In response to convergence of the model parameters, modeling system 102 may generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0092] In some non-limiting embodiments or aspects, modeling system 102 may determine the BPR loss based on the first regularized parameter to prevent overfitting. When modeling system 102 generates the universal perturbation matrix, an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss may be limited by the second regularized parameter. Modeling system 102 may generate the universal perturbation matrix as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter. Modeling system 102 may further generate the universal perturbation matrix based on a sigmoid function of the perturbation parameters.

[0093] In some non-limiting embodiments or aspects, the interaction matrix may include a user-item interaction matrix. The GNN model may be trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0094] In some non-limiting embodiments or aspects, modeling system 102 may identify an interaction matrix, *e.g.*, a user-item interaction matrix. A user-item interaction matrix may be represented by:

Formula 1

$$\mathbf{R} \in \mathbb{R}^{|U| \times |I|}$$

where $|U|$ represents the number of users, $|I|$ represents the number of items, and R represents a relationship matrix, where $R_{ui}=1$ if user u has interacted with item i before, and $R_{ui}=0$ if user u has not interacted with item i before. Modeling system 102 may denote a general interaction matrix by substituting a first set of nodes M for “ U ” and a second set of nodes N for “ I ”, such that relationships between the first set of nodes M and the second set of nodes N may be modeled. For example, a relationship between node m (representing a first entity) and node n (representing a second entity) may be modeled as R_{mn} , where $R_{mn}=1$ if a relationship exists between node m and node n , and $R_{mn}=0$ if a relationship does not exist between node m and node n .

[0095] Modeling system 102 may determine a corresponding adjacency matrix of the interaction matrix as follows:

Formula 2

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^\top & \mathbf{0} \end{bmatrix}$$

where A is the adjacency matrix, and $R^\top$ is the transposition of the relationship matrix R . Modeling system 102 may use the adjacency matrix in a GNN with the goal of producing a recommendation, *e.g.*, a ranked list of items for which a user had no prior interactions, such as by performing link prediction on the bipartite graph.

[0096] In some non-limiting embodiments or aspects, modeling system 102 may use a GNN model to iteratively update the representation of each node by aggregating messages over each node’s neighbors. Modeling system 102 may use a message-passing schema represented as:

Formula 3

$$\mathbf{H}^{(l)} = f_{\text{agg}}(\mathbf{H}^{(l-1)}, \mathbf{A})$$

where $H^{(l)}$ represents the nodes' embeddings at the l -th layer, in which $H^{(0)}$ may be initialized with identifier (ID) embeddings via lookup tables, and where $f_{agg}(\cdot)$ is a differentiable aggregation function.

[0097] After L layers, modeling system 102 may use a readout function to generate the final embeddings for each node v , which may be annotated as:

Formula 4

$$\mathbf{h}_v = f_{readout}(\{\mathbf{h}_v^{(l)}, 0 \leq l \leq L\})$$

where h_v represents the final embedding of node v . For example, $f_{readout}(\cdot)$ may be a concatenation function, a weighted sum function, and/or the like. In some non-limiting embodiments or aspects, modeling system 102 may use an inner product to predict how likely it is that user u would interact with item i , which may be represented as:

Formula 5

$$\hat{y}_{ui} = \mathbf{h}_u^\top \mathbf{h}_i$$

[0098] To optimize model parameters, modeling system 102 may employ pairwise Bayesian Personalized Ranking (BPR) loss that encourages the prediction score of an observed entry to be higher than its unobserved counterparts. Modeling system 102 may represent BPR loss as:

Formula 6

$$\mathcal{L}_{BPR}(\Theta) = \sum_{(u,i,j) \in \mathcal{D}} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \alpha \|\Theta\|_F^2,$$

where $\mathcal{D} = \{(u, i, j) | R_{ui} = 1, R_{uj} = 0\}$ is the set of pairwise training instances, $\sigma(\cdot)$ is the sigmoid function, Θ denotes the model parameters, and α is the regularized parameter to prevent overfitting.

[0099] In some non-limiting embodiments or aspects, modeling system 102 may inject perturbations to the original graph when training a GNN. As such, Formula 3 may become:

Formula 7

$$\mathbf{H}^{(l)} = f_{agg}(\mathbf{H}^{(l-1)}, \mathbf{A} + \Delta),$$

where $\Delta = \{\Delta_{ij} \in \{0, 1\} \wedge \Delta_{ij} \notin A\}$ denotes the universal graph perturbations. Modeling system 102 may use the universal perturbations to improve both robustness and

generalization performance in image classification. Modeling system 102 may use adversarial perturbations with the goal of causing the largest influence (e.g., worst-case perturbations). Therefore, modeling system 102 may identify the optimal adversarial perturbations Δ_{adv} by maximizing the BPR loss as follows:

Formula 8

$$\Delta_{adv} = \arg \max_{\Delta} \mathcal{L}_{BPR}(\hat{\Theta}, A + \Delta), \quad \text{s.t.} \quad \|\Delta\|_0 \leq \delta,$$

where $\hat{\Theta}$ is a constant denoting the current model parameters. Because adversarial perturbations may be expected to be small, unnoticeable, and non-corrupting of the majority of graph structures, modeling system 102 may thus penalize the number of non-zero entries of Δ via L_0 norm with an upper bound $\delta \geq 0$.

[0100] In some non-limiting embodiments or aspects, modeling system 102 may use an objective function that is suitable for personalized ranking and is robust to adversarial graph perturbations. To that end, modeling system may minimize the adversarial BP loss by combining Formula 6 and Formula 8, as follows:

Formula 9

$$\mathcal{L}(\Theta) = \mathcal{L}_{BPR}(\Theta) + \beta \mathcal{L}_{BPR}(\Theta, A + \Delta_{adv}),$$

$$\text{where } \Delta_{adv} = \arg \max_{\Delta, \|\Delta\|_0 \leq \delta} \mathcal{L}_{BPR}(\hat{\Theta}, A + \Delta),$$

where β controls the impact of the adversarial perturbations on the BPR loss. In the extreme case, e.g., $\beta = 0$, Formula 9 may reduce to Formula 6. The objective of Formula 9 may consist of two steps: 1) an inner maximization that computes the universal perturbations Δ^* by fixing the model parameters Θ , and 2) an outer minimization that computes the model parameters Θ^* by fixing the perturbations Δ . The outer minimization may be represented as a GNN optimization problem, and modeling system 102 may learn Θ^* via Stochastic Gradient Descent. However, because the inner problem is intractable due to the discrete nature of Δ , modeling system 102 may employ continuous relaxation of Δ via probabilistic reparameterization.

[0101] In some non-limiting embodiments or aspects, the Δ in Formula 8 may contain $2^{|A|}$ possible candidates of edge perturbations. Modeling system 102 may use mask techniques to generate sub-graph perturbations that can be scalable to large

graphs. For example, given bipartite graph A , its complementary graph $\bar{A}$ contains all unobserved edges. Modeling system 102 may sample a sub-graph $T \sim (\bar{A})$ with a fixed budget of edges B . Then, modeling system 102 may generate the universal perturbations as: $\Delta = T \odot M$, where $\odot$ denotes the element-wise product and $M \in \{0, 1\}$ is a learnable binary mask (e.g., 1 is kept and 0 is dropped). As such, modeling system 102 may approximate the inner optimization of Formula 8 as:

Formula 10

$$\max_{\mathbf{M}} \mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbf{M}) - \lambda \|\mathbf{M}\|_0,$$

where the sparsity of Δ (e.g., $\|\Delta\|_0 \leq \delta$) can be achieved by adding L_0 regularizer on M with hyperparameter $\lambda \geq 0$. Formula 6 may still be intractable since the discrete variable M and the L_0 norm are non-differentiable. Modeling system 102 may compute the gradient via the REINFORCE algorithm (e.g., a Monte Carlo Policy gradient), and may reduce the high variance thereof by using an Antithetic REINFORCE-based gradient estimator to solve the inner optimization.

[0102] Modeling system 102 may consider each M_{ij} to be drawn from a Bernoulli distribution parameterized by $\Pi_{ij} \in [0, 1]$ such that $M_{ij} \sim \text{Bern}(M_{ij}; \Pi_{ij})$. Modeling system 102 may then rewrite Formula 10 by its expectation:

Formula 11

$$\max_{\Pi} \mathbb{E}_{\mathbf{M} \sim \text{Bernoulli}(\mathbf{M}; \Pi)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbf{M})] - \lambda \sum_{i,j} \Pi_{ij}.$$

To efficiently compute gradients of discrete variable Π , modeling system 102 may use probabilistic reparameterization to reparameterize $\Pi_{ij} \in [0, 1]$ with a deterministic function $g(\cdot)$, with parameter $\Phi_{ij} : \Pi_{ij} = g(\Phi_{ij})$. Since $g(\cdot)$ should be bounded within $[0, 1]$, modeling system 102 may choose the sigmoid function $\sigma(\cdot)$ as the deterministic function. This arrangement provides a number of probability properties. As a first property, given a Bernoulli random variable $z \sim \text{Bernoulli}(z; \sigma(\phi))$, variable z can be represented by comparing two augmented exponential random variables:

Formula 12

$$z = \mathbb{1}[xe^{-\phi} < y], \quad x, y \stackrel{i.i.d}{\sim} \text{Exp}(1),$$

where $\mathbb{1}[\cdot]$ is the indicator function having the value 1 if the condition is true and 0 otherwise. As a second property, for a random variable $z = xe^{-\phi}$, $x \sim \text{Exp}(1)$, variable z is equal in distribution to $z \sim (\text{Exp}(e^\phi))$. As a third property, for two variables:

Formula 13

$$x, y \stackrel{i.i.d}{\sim} \text{Exp}(1)$$

they are the same in distribution as $x = \epsilon u$, $y = \epsilon(1-u)$, where $u \sim \text{Uniform}(0, 1)$, $\epsilon \sim \text{Gamma}(2, 1)$.

[0103] With the first property and second property (described above), Formula 11 may be equivalent to:

Formula 14

$$\begin{aligned} f(\Phi) &= \mathbb{E}_{\mathbf{M} \sim \text{Bernoulli}(\mathbf{M}; \sigma(\Phi))} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbf{M})] - \lambda \sum_{i,j} \sigma(\Phi_{ij}) \\ &= \mathbb{E}_{\mathbf{X}, \mathbf{Y} \stackrel{i.i.d}{\sim} \text{Exp}(1)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbb{1}[\mathbf{X} \odot e^{-\Phi} < \mathbf{Y}])] - \lambda \sum_{i,j} \sigma(\Phi_{ij}) \\ &= \mathbb{E}_{\mathbf{Z} \sim \text{Exp}(e^\Phi), \mathbf{Y} \sim \text{Exp}(1)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbb{1}[\mathbf{Z} < \mathbf{Y}])] - \lambda \sum_{i,j} \sigma(\Phi_{ij}) \end{aligned}$$

[0104] Modeling system 102 may compute the gradients with respect to ϕ . The second term, therefore, is differentiable and its gradient is:

Formula 15

$$c = \lambda \sum_{i,j} \nabla \Phi_{ij} \sigma(\Phi_{ij})$$

Applying the REINFORCE algorithm to the first term in Formula 14, Formula 14 becomes:

Formula 16

$$\begin{aligned} &\nabla_{\Phi} f(\Phi) \\ &= \mathbb{E}_{\mathbf{Z} \sim \text{Exp}(e^\Phi), \mathbf{Y} \sim \text{Exp}(1)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbb{1}[\mathbf{Z} < \mathbf{Y}]) \nabla_{\Phi} \log \text{Exp}(\mathbf{Z}; e^\Phi)] - c \\ &= \mathbb{E}_{\mathbf{Z} \sim \text{Exp}(e^\Phi), \mathbf{Y} \sim \text{Exp}(1)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbb{1}[\mathbf{Z} < \mathbf{Y}]) (1 - \mathbf{Z} \odot e^\Phi)] - c \\ &= \mathbb{E}_{\mathbf{X}, \mathbf{Y} \stackrel{i.i.d}{\sim} \text{Exp}(1)} [\mathcal{L}_{\text{BPR}}(\hat{\Theta}, \mathbf{A} + \mathbf{T} \odot \mathbb{1}[\mathbf{X} \odot e^{-\Phi} < \mathbf{Y}]) (1 - \mathbf{X})] - c \end{aligned}$$

[0105] With the third property (described above), modeling system 102 may express

Formula 17

$$X, Y \stackrel{i.i.d}{\sim} \text{Exp}(1)$$

as:

Formula 18

$$X = E \odot U, \quad Y = E \odot (1 - U),$$

where U and E can be drawn from Uniform(0, 1) and Gamma (2, 1), respectively.

Accordingly, the inequality in Formula 16 becomes:

Formula 19

$$X \odot e^{-\Phi} < Y \Leftrightarrow U < \sigma(\Phi)$$

and Formula 16 may be expressed as:

Formula 20

$$\nabla_{\Phi} f(\Phi) = \mathbb{E}_{\substack{U \sim \text{Uniform}(0,1), \\ E \sim \text{Gamma}(2,1)}} \left[\mathcal{L}_{\text{BPR}}(\hat{\Theta}, A + T \odot \mathbb{1}[U < \sigma(\Phi)]) (1 - E \odot U) \right] - c$$

Using Rao-Blackwellization, modeling system 102 may further obtain gradients as:

Formula 21

$$\nabla_{\Phi} f(\Phi) = \mathbb{E}_{U \sim \text{Uniform}(0,1)} \left[\mathcal{L}_{\text{BPR}}(\hat{\Theta}, A + T \odot \mathbb{1}[U < \sigma(\Phi)]) (1 - 2U) \right] - c$$

[0106] In view of the above, modeling system 102 may efficiently solve the discrete optimization of Formula 10 via probabilistic reparameterization. The gradient estimator shown above provides a number of technical benefits, including: (i) applying antithetic sampling over augmented variables yields unbiased and low-variance gradients; and (ii) the gradient estimator has low computational complexity, in that sampling from a Bernoulli distribution is replaced by a non-parametric Uniform distribution, and the gradients in Formula 21 only involve one forward-pass of networks.

[0107] In non-limiting embodiments or aspects, modeling system 102 may iteratively update Δ and Θ . First, modeling system 102 may receive an input of GNN f_{gnn} , bipartite graph A , perturbations budget B , regularized parameters α and β for Formula 9, and regularized parameter λ for Formula 11. Modeling system 102 may

then initialize Θ by solving the original BPR loss of Formula 6. Next, modeling system 102 may initialize ϕ randomly. Modeling system 102 may then sample a sup-graph $T \sim (\bar{A})$ with budget B . Next, modeling system 102 may repeated perform a process until convergence, wherein the process includes: drawing $U \sim \text{Uniform}(0, 1)$; computing universal perturbations $\Delta = T \odot 1 [U < \sigma(\phi)]$; feeding (A, Δ) to the GNN f_{gnn} and computing the loss in Formula 9; updating ϕ with stochastic gradient ascent as shown in Formula 21, such that $\phi \leftarrow \phi + \eta_1 \cdot \nabla_{\phi} f(\phi)$; and updating Θ with stochastic gradient descent as shown in Formula 9, such that $\Theta \leftarrow \Theta - \eta_2 \cdot \nabla_{\Theta} \mathcal{L}(\Theta)$. Modeling system 102 may repeat this process may be repeated until convergence of the model parameters Θ . After convergence, modeling system 102 may return the model parameters Θ .

[0108] The number and arrangement of devices and networks shown in FIG. 1 are provided as an example. There may be additional devices and/or networks, fewer devices and/or networks, different devices and/or networks, or differently arranged devices and/or networks than those shown in FIG. 1. Furthermore, two or more devices shown in FIG. 1 may be implemented within a single device, or a single device shown in FIG. 1 may be implemented as multiple, distributed devices. Additionally or alternatively, a set of devices (e.g., one or more devices) of environment 100 may perform one or more functions described as being performed by another set of devices of environment 100.

[0109] Referring now to FIG. 2, shown is a diagram of example components of a device 200, according to some non-limiting embodiments or aspects. Device 200 may correspond to one or more devices of modeling system 102, user device 106, memory 110, and/or communication network 108, as shown in FIG. 1. In some non-limiting embodiments or aspects, such systems or devices may include at least one device 200 and/or at least one component of device 200. The number and arrangement of components shown in FIG. 2 are provided as an example. In some non-limiting embodiments or aspects, device 200 may include additional components, fewer components, different components, or differently arranged components than those shown in FIG. 2. Additionally, or alternatively, a set of components (e.g., one or more components) of device 200 may perform one or more functions described as being performed by another set of components of device 200.

[0110] As shown in FIG. 2, device 200 may include a bus 202, a processor 204, memory 206, a storage component 208, an input component 210, an output component 212, and a communication interface 214. Bus 202 may include a component that permits communication among the components of device 200. In some non-limiting embodiments or aspects, processor 204 may be implemented in hardware, firmware, or a combination of hardware and software. For example, processor 204 may include a processor (*e.g.*, a central processing unit (CPU), a graphics processing unit (GPU), an accelerated processing unit (APU), etc.), a microprocessor, a digital signal processor (DSP), and/or any processing component (*e.g.*, a field-programmable gate array (FPGA), an application-specific integrated circuit (ASIC), etc.) that can be programmed to perform a function. Memory 206 may include random access memory (RAM), read only memory (ROM), and/or another type of dynamic or static storage device (*e.g.*, flash memory, magnetic memory, optical memory, etc.) that stores information and/or instructions for use by processor 204.

[0111] With continued reference to FIG. 2, storage component 208 may store information and/or software related to the operation and use of device 200. For example, storage component 208 may include a hard disk (*e.g.*, a magnetic disk, an optical disk, a magneto-optic disk, a solid state disk, etc.) and/or another type of computer-readable medium. Input component 210 may include a component that permits device 200 to receive information, such as via user input (*e.g.*, a touch screen display, a keyboard, a keypad, a mouse, a button, a switch, a microphone, etc.). Additionally, or alternatively, input component 210 may include a sensor for sensing information (*e.g.*, a global positioning system (GPS) component, an accelerometer, a gyroscope, an actuator, etc.). Output component 212 may include a component that provides output information from device 200 (*e.g.*, a display, a speaker, one or more light-emitting diodes (LEDs), etc.). Communication interface 214 may include a transceiver-like component (*e.g.*, a transceiver, a separate receiver and transmitter, etc.) that enables device 200 to communicate with other devices, such as via a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication interface 214 may permit device 200 to receive information from another device and/or provide information to another device. For example, communication interface 214 may include an Ethernet interface, an optical interface, a coaxial interface, an infrared interface, a radio frequency (RF) interface, a

universal serial bus (USB) interface, a Wi-Fi® interface, a cellular network interface, and/or the like.

[0112] Device 200 may perform one or more processes described herein. Device 200 may perform these processes based on processor 204 executing software instructions stored by a computer-readable medium, such as memory 206 and/or storage component 208. A computer-readable medium may include any non-transitory memory device. A memory device includes memory space located inside of a single physical storage device or memory space spread across multiple physical storage devices. Software instructions may be read into memory 206 and/or storage component 208 from another computer-readable medium or from another device via communication interface 214. When executed, software instructions stored in memory 206 and/or storage component 208 may cause processor 204 to perform one or more processes described herein. Additionally, or alternatively, hardwired circuitry may be used in place of or in combination with software instructions to perform one or more processes described herein. Thus, embodiments or aspects described herein are not limited to any specific combination of hardware circuitry and software. The term “programmed or configured,” as used herein, refers to an arrangement of software, hardware circuitry, or any combination thereof on one or more devices.

[0113] Referring now to FIG. 3, a flow diagram of a method 300 for generating robust GNNs using universal adversarial training is shown, according to some non-limiting embodiments or aspects of the present disclosure. The method 300 may be performed by one or more processors of modeling system 102, user device 106, memory 110, and/or another computing device. One or more steps performed by a first processor may be performed by a same or different processor. The steps shown in FIG. 3 are for example purposes only. It will be appreciated that additional, fewer, different, and/or a different order of steps may be used in non-limiting embodiments or aspects.

[0114] As shown in FIG. 3, method 300 may include, in step 302, receiving input of a GNN model and a bipartite graph. For example, modeling system 102 may receive input including a GNN model and a bipartite graph including an adjacency matrix determined from an interaction matrix.

[0115] In some non-limiting embodiments or aspects, the interaction matrix may include a user-item interaction matrix generated from historic interactions between users and items (e.g., products, services, transactions, internet resources, etc.).

[0116] In some non-limiting embodiments or aspects, the GNN model may be trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

[0117] In some non-limiting embodiments or aspects, the input may further include a perturbations budget, a first regularized parameter, a second regularized parameter, a third regularized parameter, or any combination thereof.

[0118] As shown in FIG. 3, method 300 may include, in step 304, initializing model parameters. For example, modeling system 102 may initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss. BPR loss may be based on the first regularized parameter, *e.g.*, to prevent overfitting.

[0119] As shown in FIG. 3, method 300 may include, in step 306, initializing perturbation parameters. For example, modeling system 102 may initialize perturbation parameters randomly.

[0120] As shown in FIG. 3, method 300 may include, in step 308, determining a complementary graph based on the bipartite graph. For example, modeling system 102 may determine the complementary graph based on the bipartite graph. In some non-limiting embodiments or aspects, the complementary graph may be generated based on the adjacency matrix of the bipartite graph (*e.g.*, the complementary graph may include all unobserved edges from the bipartite graph).

[0121] As shown in FIG. 3, method 300 may include, in step 310, sampling a subgraph of the complementary graph. For example, modeling system 102 may sample a subgraph of the complementary graph.

[0122] In some non-limiting embodiments or aspects, the perturbations budget may specify a number of edges of the subgraph.

[0123] As shown in FIG. 3, method 300 may include, in step 312, repeating method 400 (shown in FIG. 4) until convergence of model parameters. For example, these repeated steps may include the steps shown in method 400 of FIG. 4.

[0124] As shown in FIG. 3, method 300 may include, in step 314, outputting the model parameters. For example, in response to convergence of the model parameters, modeling system 102 may output the model parameters.

[0125] In some non-limiting embodiments or aspects, modeling system 102 may generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

[0126] Referring now to FIG. 4, a flow diagram of a method 400 for generating robust GNNs using universal adversarial training is shown, according to some non-limiting embodiments or aspects of the present disclosure. The method 400 may be performed by one or more processors of modeling system 102, user device 106, memory 110, and/or another computing device. One or more steps performed by a first processor may be performed by a same or different processor. The steps shown in FIG. 4 are for example purposes only. It will be appreciated that additional, fewer, different, and/or a different order of steps may be used in non-limiting embodiments or aspects.

[0127] In some non-limiting embodiments or aspects, steps 402-410 may be repeated until convergence of the model parameters. Additionally or alternatively, step 402 may be executed after execution of step 310 of the method 300 detailed in FIG. 3.

[0128] As shown in FIG. 4, method 400 may include, in step 402, drawing a random variable from a uniform distribution. For example, modeling system 102 may draw a random variable from a uniform distribution.

[0129] As shown in FIG. 4, method 400 may include, in step 404, generating a universal perturbation matrix. For example, modeling system 102 may generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters.

[0130] In some non-limiting embodiments or aspects, the universal perturbation matrix may also be generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter. Additionally or alternatively, the universal perturbation matrix may be further generated based on a sigmoid function of the perturbation parameters.

[0131] As shown in FIG. 4, method 400 may include, in step 406, determining BPR loss. For example, modeling system 102 may determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model.

[0132] In some non-limiting embodiments or aspects, the effect of adversarial perturbations of the universal perturbation matrix on the BPR loss may be limited by the second regularized parameter.

[0133] As shown in FIG. 4, method 400 may include, in step 408, updating the perturbation parameters. For example, modeling system 102 may update the perturbation parameters based on stochastic gradient ascent.

[0134] As shown in FIG. 4, method 400 may include, in step 410, updating the model parameters. For example, modeling system 102 may update the model parameters based on stochastic gradient descent.

[0135] In some non-limiting embodiments or aspects, the stochastic gradient descent may be further based at least partly on BPR loss.

[0136] As shown in FIG. 4, method 400 may include, in step 412, evaluating the model parameters for convergence. For example, modeling system 102 may evaluate the model parameters for convergence (e.g., a local extrema, a change between iterations that is less than a threshold, and/or the like).

[0137] In response to determining that the model parameters have not yet converged, modeling system 102 may proceed to re-executing steps 402-410. In response to determining that the model parameters have converged, modeling system 102 may exit the loop and proceed to step 314 of FIG. 3.

[0138] With further reference to the foregoing figures, “electronic payment processing network” may refer to a plurality computing devices that transmit and/or receive communications between one or more entities for processing the transfer of monetary funds to one or more transactions. The electronic payment processing network may include a merchant system, an acquirer system, a transaction service provider system, and an issuer system.

[0139] Although the disclosure has been described in detail for the purpose of illustration based on what is currently considered to be the most practical and preferred embodiments or aspects, it is to be understood that such detail is solely for that purpose and that the disclosure is not limited to the disclosed embodiments or aspects, but, on the contrary, is intended to cover modifications and equivalent arrangements that are within the spirit and scope of the appended claims. For example, it is to be understood that the present disclosure contemplates that, to the extent possible, one or more features of any embodiment or aspect can be combined with one or more features of any other embodiment or aspect, and one or more steps may be taken in a different order than presented in the present disclosure.

WHAT IS CLAIMED IS:

1. A computer-implemented method comprising:
 - receiving, with at least one processor, input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix;
 - initializing, with at least one processor, model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss;
 - initializing, with at least one processor, perturbation parameters randomly;
 - determining, with at least one processor, a complementary graph based on the bipartite graph;
 - sampling, with at least one processor, a subgraph of the complementary graph;
 - repeating, with at least one processor, until convergence of the model parameters:
 - drawing a random variable from a uniform distribution;
 - generating a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters;
 - determining BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model;
 - updating the perturbation parameters based on stochastic gradient ascent; and
 - updating the model parameters based on stochastic gradient descent; and
 - in response to convergence of the model parameters, outputting, with at least one processor, the model parameters.
2. The computer-implemented method of claim 1, further comprising, in response to convergence of the model parameters, generating, with at least one processor, at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

3. The computer-implemented method of claim 1, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.
4. The computer-implemented method of claim 1, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.
5. The computer-implemented method of claim 1, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.
6. The computer-implemented method of claim 1, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.
7. The computer-implemented method of claim 1, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.
8. The computer-implemented method of claim 1, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.
9. The computer-implemented method of claim 1, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.
10. A system comprising at least one processor, the at least one processor being programmed or configured to:

receive input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix;
initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss;
initialize perturbation parameters randomly;
determine a complementary graph based on the bipartite graph;
sample a subgraph of the complementary graph;
repeat until convergence of the model parameters:
 draw a random variable from a uniform distribution;
 generate a universal perturbation matrix based on the subgraph, the random variable, and the perturbation parameters;
 determine BPR loss by inputting the bipartite graph and the universal perturbation matrix to the GNN model;
 update the perturbation parameters based on stochastic gradient ascent; and
 update the model parameters based on stochastic gradient descent; and
in response to convergence of the model parameters, output the model parameters.

11. The system of claim 10, wherein the at least one processor is further programmed or configured to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

12. The system of claim 10, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.

13. The system of claim 10, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

14. The system of claim 10, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

15. The system of claim 10, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

16. The system of claim 10, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

17. The system of claim 10, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.

18. The system of claim 10, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.

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

receive input comprising a graph neural network (GNN) model and a bipartite graph comprising an adjacency matrix determined from an interaction matrix;

initialize model parameters of the GNN model based on a Bayesian Personalized Ranking (BPR) loss;

initialize perturbation parameters randomly;

determine a complementary graph based on the bipartite graph;

sample a subgraph of the complementary graph;

repeat until convergence of the model parameters:

draw a random variable from a uniform distribution;
generate a universal perturbation matrix based on the subgraph,
the random variable, and the perturbation parameters;
determine BPR loss by inputting the bipartite graph and the
universal perturbation matrix to the GNN model;
update the perturbation parameters based on stochastic gradient
ascent; and
update the model parameters based on stochastic gradient
descent; and
in response to convergence of the model parameters, output the model
parameters.

20. The computer program product of claim 19, wherein the one or more instructions cause the at least one processor to, in response to convergence of the model parameters, generate at least one recommendation based on the interaction matrix by executing the GNN model using the model parameters.

21. The computer program product of claim 19, wherein the input further comprises a perturbations budget, and wherein the perturbations budget specifies a number of edges of the subgraph.

22. The computer program product of claim 19, wherein the input further comprises a first regularized parameter, and wherein the BPR loss is based on the first regularized parameter to prevent overfitting.

23. The computer program product of claim 19, wherein the input further comprises a second regularized parameter, and wherein an effect of adversarial perturbations of the universal perturbation matrix on the BPR loss is limited by the second regularized parameter.

24. The computer program product of claim 19, wherein the input further comprises a third regularized parameter, and wherein the universal perturbation matrix is generated as a sparse graph by using a learnable binary mask that is weighted with the third regularized parameter.

25. The computer program product of claim 19, wherein the interaction matrix comprises a user-item interaction matrix, and wherein the GNN model is trained to generate at least one item recommendation for at least one user based on the user-item interaction matrix.

26. The computer program product of claim 19, wherein the generation of the universal perturbation matrix is further based on a sigmoid function of the perturbation parameters.

27. The computer program product of claim 19, wherein the update of the model parameters based on stochastic gradient descent is further based at least partly on BPR loss.