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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
05 May 2022 (05.05.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/093979 A1

(51) International Patent Classification:
G16B 30/20 (2019.01) *G16B 35/10* (2019.01)
G16B 15/20 (2019.01) *G06N 3/08* (2006.01)
G16B 40/20 (2019.01)

(21) International Application Number:
PCT/US2021/056879

(22) International Filing Date:
27 October 2021 (27.10.2021)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/105,926 27 October 2020 (27.10.2020) US
17/510,882 26 October 2021 (26.10.2021) US

(71) Applicant: **NEC LABORATORIES AMERICA, INC.** [US/US]; 4 Independence Way, Suite 200, Princeton, NJ 08540 (US).

(72) Inventors: **MIN, Renqiang**; 7 Carlton Circle, Princeton, NJ 08540 (US). **DURDANOVIC, Igor**; 734 Putnam Avenue, Lawrenceville, NJ 08648 (US). **GRAF, Hans Peter**; 55 South Shore Drive, South Amboy, NJ 08879 (US).

(74) Agent: **BITETTO, James J.**; Tutunjian & Bitetto, P.C., 401 Broadhollow Road, Suite 402, Melville, NY 11747 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

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

(54) Title: PEPTIDE-BASED VACCINE GENERATION

(57) Abstract: Methods and systems for generating (208) a peptide sequence include transforming an input peptide sequence into disentangled representations, including a structural representation and an attribute representation, using an autoencoder model. One of the disentangled representations is modified (210). The disentangled representations, including the modified disentangled representation, are transformed (212) to generate a new peptide sequence using the autoencoder model.

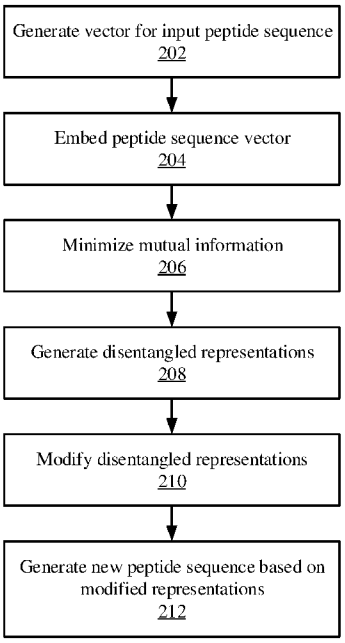

FIG. 2

WO 2022/093979 A1

[Continued on next page]

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

PEPTIDE-BASED VACCINE GENERATION

RELATED APPLICATION INFORMATION

[0001] This application claims priority to U.S. Patent Application Serial No. 17/510,882, filed on October 26, 2021, and U.S. Provisional Patent Application Serial No. 63/105,926, filed on October 27, 2020, both incorporated herein by reference in their entirety.

BACKGROUND

Technical Field

[0002] The present invention relates to peptide searching, and, more particularly, to identifying potential new binding peptides with new properties.

Description of the Related Art

Peptide-MHC (Major Histocompatibility Complex) protein interactions are involved in cell-mediated immunity, regulation of immune responses, and transplant rejection. While computational tools exist to predict a binding interaction score between an MHC protein and a given peptide, tools for generating new binding peptides with new specified properties from existing binding peptides are lacking.

SUMMARY

[0003] A method for generating a peptide sequence includes transforming an input peptide sequence into disentangled representations, including a structural representation and an attribute representation, using an autoencoder model. One of the disentangled representations is modified. The disentangled representations, including

the modified disentangled representation, are transformed to generate a new peptide sequence using the autoencoder model.

[0004] A method for generating a peptide sequence includes training a Wasserstein neural network model using a set of training peptide sequences by minimizing a mutual information between a structural representation and an attribute representation of the training peptide sequences. An input peptide sequence is transformed into disentangled structural and attribute representations, using an encoder of the Wasserstein autoencoder neural network model. One of the disentangled representations is modified to alter an attribute to improve vaccine efficacy against a predetermined pathogen, including changing coordinates of a vector representation of the disentangled representations within an embedding space. The disentangled representations, including the modified disentangled representation, are transformed to generate a new peptide sequence using a decoder of the Wasserstein autoencoder neural network model.

[0005] A system for generating a peptide sequence includes a hardware processor and a memory that stores a computer program product. When executed by the hardware processor, the computer program product causes the hardware processor to transform an input peptide sequence into disentangled representations, including a structural representation and an attribute representation, using an autoencoder model, to modify one of the disentangled representations, and to transform the disentangled representations, including the modified disentangled representation, to generate a new peptide sequence using the autoencoder model.

[0006] These and other features and advantages will become apparent from the following detailed description of illustrative embodiments thereof, which is to be read in connection with the accompanying drawings.

BRIEF DESCRIPTION OF DRAWINGS

[0007] The disclosure will provide details in the following description of preferred embodiments with reference to the following figures wherein:

[0008] FIG. 1 is a diagram illustrating a peptide and a major histocompatibility complex binding, in accordance with an embodiment of the present invention;

[0009] FIG. 2 is a block/flow diagram of a method of generating modified peptide sequences with useful attributes, in accordance with an embodiment of the present invention;

[0010] FIG. 3 is a block diagram of a Wasserstein autoencoder that generates disentangled representations of an input peptide sequence and modifies the disentangled representations to generate a new peptide sequence, in accordance with an embodiment of the present invention;

[0011] FIG. 4 is a block/flow diagram of a method for training a Wasserstein autoencoder to generate disentangled representations of an input peptide sequence, in accordance with an embodiment of the present invention;

[0012] FIG. 5 is a block diagram of a computing device that can perform peptide sequence generation, in accordance with an embodiment of the present invention;

[0013] FIG. 6 is a block diagram of a peptide sequence generation system that uses a Wasserstein autoencoder to modify attributes of an input peptide sequence and to generate a new peptide sequence, in accordance with an embodiment of the present invention;

[0014] FIG. 7 is a diagram illustrating a neural network architecture, in accordance with an embodiment of the present invention; and

[0015] FIG. 8 is a diagram illustrating a deep neural network architecture, in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0016] Strongly binding peptides can be generated given a set of existing positive binding peptide examples for a major histocompatibility complex (MHC) protein. For example, a regularized Wasserstein autoencoder may be used to generate disentangled representations of a peptide. These disentangled representations may include a first representation of structural information for the peptide, and a second representation for attribute information for the peptide. The disentangled representations may then be altered to change the properties of the peptide, and the autoencoder's decoder may then be used to convert the altered disentangled representations into a new peptide that has the desired attributes.

[0017] Prediction of binding peptides for MHC proteins is helpful in vaccine research and design. Once a binding peptide for an MHC protein has been identified, it can be used in the generation of a new peptide vaccine with new properties to target a pathogen, such as a virus. The existing binding peptide may be used as input to the encoder of a learned regularized Wasserstein autoencoder to get disentangled representations, and the decoder of the Wasserstein autoencoder may be used to generate new peptides from the altered representations with new properties. For example, the structural and sequence similarity information of the existing binding peptide may be maintained, but the antigen processing score and T-cell receptor interaction score of the given binding peptide may be increased. The newly generated peptides are similar to the given binding peptide, but may have much higher chances of triggering immune responses corresponding to the targeted T-cell receptors.

[0018] Disentangled representation learning maps different aspects of data into distinct and independent low-dimensional latent vector spaces, and can be used to make deep learning models more interpretable. To disentangle the attributes of peptides, two

different types of embeddings may be used, including an attribute embedding and a content embedding. The content embedding may encapsulate general structural or sequential constraints of a peptide, while the attribute embedding may represent attributes such as the binding, antigen processing, and T-cell receptor recognition properties of a peptide.

[0019] Referring now to FIG. 1, a diagram of a peptide-MHC protein bond is shown. A peptide 102 is shown as bonding with an MHC protein 104, with complementary two-dimensional interfaces of the figure suggesting complementary shapes of these three-dimensional structures. The MHC protein 104 may be attached to a cell surface 106.

[0020] An MHC is an area on a DNA strand that codes for cell surface proteins that are used by the immune system. MHC molecules are used by the immune system and contribute to the interactions of white blood cells with other cells. For example, MHC proteins impact organ compatibility when performing transplants and are also important to vaccine creation.

[0021] A peptide, meanwhile, may be a portion of a protein. When a pathogen presents peptides that are recognized by a MHC protein, the immune system triggers a response to destroy the pathogen. Thus, by finding peptide structures that bind with MHC proteins, an immune response may be intentionally triggered, without introducing the pathogen itself to a body. In particular, given an existing peptide that binds well with the MHC protein 104, a new peptide 102 may be automatically identified according to desired properties and attributes.

[0022] Referring now to FIG. 2, a method for generating new binding peptide sequences is shown. Block 202 accepts an input peptide sequence and generates a vector representation. For example, this vector may be a blocks substitution matrix

(BLOSUM) representation of the peptide sequence, which may be implemented as a matrix for sequence alignment of proteins.

[0023] Block 204 encodes the input vector using the encoder part of an autoencoder model. As will be described in greater detail below, the encoder part of the autoencoder model translates a vector representation into an embedding in a latent space, and the decoder part translates an embedding in the latent space back into a vector representation. This embedding may include minimization of mutual information between distinct disentangled representations, including a structure representation and an attribute representation, in block 206, thereby generating the disentangled representations in block 208.

[0024] Block 210 makes modifications to the disentangled representations. For example, the attributes of the peptide can be altered by moving an attribute representation vector within the latent space. Modifying only the attribute representation, while keeping the structure representation the same, will produce a peptide that is structurally and sequentially similar to the input peptide, but that has the new attributes indicated by the altered attribute representation. As the disentangled representations may be represented by vectors in a latent space, this modification may be performed by changing the coordinates of one or more such vectors.

[0025] For example, the attribute representation of the given binding peptide can be replaced with the corresponding attribute representation of another peptide that has high binding affinity, and/or high antigen processing score, and/or high T-cell receptor interaction score. In this way, the newly generated peptide from the altered disentangled representation will have high sequence similarity to the original given binding peptide and the desired attributes.

[0026] Block 212 translates the altered disentangled representations back to a peptide sequence vector representation, for example using the decoder part of the autoencoder model. This generates a new peptide sequence that can have high binding affinity, and/or high antigen processing score, and/or high T-cell receptor interaction score.

[0027] Referring now to FIG. 3, a peptide generation model is shown. An input peptide sequence 301 is provided, for example in the form of a BLOSUM vector. An encoder 302 embeds the input peptide sequence 301 into a latent space, particularly generating disentangled representations that may include a structure or sequence content representation 304 and an attribute representation.

[0028] Modifications 307 may be made to the attribute representation 306 in the latent space. When a decoder 308 transforms the structure representation 304 and the modified attribute representation 306/307 into a peptide sequence, the peptide sequence represents a new peptide that includes the new attributes indicated by the modification 307.

[0029] The autoencoder that is implemented by the encoder 302 and the decoder 308 may be a Wasserstein autoencoder, implemented as a generative model and trained in an end-to-end fashion. An input peptide x may be encoded into a structure or sequence content embedding s and an attribute embedding a. The attribute embedding a may be classified using a classifier $q(y|a)$ to predict an attribute label y, which may be include experimentally or computationally determined binding affinities. The structure or sequence content embedding s may be used to reconstruct the information of the input peptide x.

[0030] A network $p(a|s)$ helps to disentangle the attribute embedding and the structure embedding by minimizing mutual information, while a separate sample-based approximated mutual information term between a and s may also be minimized. The

generator $p(x|a,s)$ generates peptides based on the combination of attributes a and structure s, and may represent the decoder 304. Thus, the encoder 302 may be represented by classifier $q(a,s|x)$, which determines the disentangled representations a and s.

[0031] A prior distribution $p(a,s) = p(a)p(s)$ represents the product of two multivariate, isotropic unit-variance Gaussian functions, and may be used to regularize the posterior distribution $q(a,s|x)$ by a Wasserstein distance. The log-likelihood term for the peptide sequence reconstruction may be maximized.

[0032] The objective for the encoder may be expressed as:

$$L_{AE} = W\big(q(a,s), N(\mathbf{0},\mathbf{1})\big) - \mathbb{E}_{q(a,s|x)}[\log p(x|a,s)]$$

where $W(\cdot)$ is the 1-Wasserstein distance metric, which can be approximated by a discriminator as follows:

$$E_{p(x)}\big(D(z)\big) - E_{p(x)}\big(D(a,s)\big)$$

where z is sampled from the Gaussian distribution and (a,s) is sampled from the posterior distribution.

[0033] A regularization term may be expressed as:

$$L_{reg} = -\log q(y|a) - MI(a;s)$$

where $MI(\cdot)$ is the mutual information and may be expressed as:

$$MI(a;s) = KL(q(a,s)||q(a)q(s) = \mathrm{f}\big(\mathrm{q(s,c)}\big) - \mathrm{f}\big(\mathrm{q(s)}\big) - \mathrm{f}\big(\mathrm{q(c)}\big)$$

where $f(\cdot) = E_{q(a,s)}\log(\cdot)$ and $E_{q(a,s)}$ is the expectation with respect to $q(a,s)$. The term can be approximated using a mini-batch weighted sampling estimator, thus:

$$E_{q(a,s)}[\log q(z)] \approx \frac{1}{M}\sum_{i=1}^{M}\log\sum_{j=1}^{M} q\big(z(x_i)\big|x_j\big) + C$$

where C is a constant and M is the size of the mini-batch.

[0034] The final loss function is thus:

$$L = L_{AE} + \lambda L_{reg}$$

where λ is a regularization hyper-parameter.

[0035] After the regularized autoencoder is trained on a large-scale peptide dataset, which may include attribute and structural/content information, to learn different types of disentangled semantic factors, the disentangled factors (e.g., attribute representation 306, which may include binding/non-binding, high/low antigen processing score, high/low T-cell receptor recognition score, and structural/sequence content representation 304, which may include structural properties) can be replaced for conditional peptide generation. One type of factor may be fixed, such as a high-binding affinity or medium-binding affinity to an MHC protein, and content around the embedding can be sampled from the prior to generate new binding peptides that satisfy different properties.

[0036] In the loss function of $\mathcal{L}_{AE}$, a regularization term forces the aggregated latent attribute distribution and the aggregated latent structure/sequence content distribution to follow multivariate unit Gaussian distributions (prior distributions). To sample from the prior distribution of attribute or structure/sequence content vector, a sample can be drawn from a multivariate unit Gaussian distribution while fixing other disentangled latent representations.

[0037] The following pseudo-code may be optionally used to further disentangle the attribute representation *a* from the structural/sequence content representation *s* besides minimizing the KL-divergence based mutual information above:

[0038] Input: Data $\{x_j\}_{j=1}^{M}$, encoder $q(a, s|x)$, approximation network $p(a|s)$

[0039] for (each training iteration) do

[0040] Sample $\{a_k, s_j\}_{j=1}^{M}$ from $q(a, s|x)$

[0041] $\mathcal{L} = 1/M \sum_{j=1}^{M} \log p(a_j, s_j)$

[0042] Update $p(a|s)$ by maximizing $\mathcal{L}$

[0043] for $j = 1$ to M do

[0044] Sample k' uniformly from $\{1,2, \ldots, M\}$

[0045] $\hat{R}_j = \log p(a_j|s_j) - \log p(a_j|s_{k'})$

[0046] end

[0047] Update $q(a, s|x)$ by minimizing $\frac{1}{M}\sum_{j=1}^{M} \hat{R}_j + MI(s; c)$

[0048] End

[0049] Here M is a mini-batch size. This process may be performed during the training to further disentangle the attribute representation from the sequence content/structural representation.

[0050] Referring now to FIG. 4, a method of training the autoencoder is shown. Block 402 encodes peptide sequences as original vectors, drawn from a training dataset. Block 404 then uses the encoder 302 to convert the vectors into disentangled representations, including the structural representation 304 and the attribute representation 306.

[0051] Block 406 decodes the disentangled representations 304 and 306 to generate reconstructed vectors. Block 408 compares the reconstructed vectors to the original vectors, identifying any differences between the respective pairs. Block 410 updates weights in the encoder 302 and the decoder 308 to correct the differences between the reconstructed vectors and the original vectors. This process may be repeated for any appropriate number of training peptide sequences, for example until a predetermined number of training steps have been performed or until the differences between original vectors and reconstructed vectors drop below a threshold value.

[0052] Embodiments described herein may be entirely hardware, entirely software or including both hardware and software elements. In a preferred embodiment, the present invention is implemented in software, which includes but is not limited to firmware, resident software, microcode, etc.

[0053] Embodiments may include a computer program product accessible from a computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any instruction execution system. A computer-usable or computer readable medium may include any apparatus that stores, communicates, propagates, or transports the program for use by or in connection with the instruction execution system, apparatus, or device. The medium can be magnetic, optical, electronic, electromagnetic, infrared, or semiconductor system (or apparatus or device) or a propagation medium. The medium may include a computer-readable storage medium such as a semiconductor or solid state memory, magnetic tape, a removable computer diskette, a random access memory (RAM), a read-only memory (ROM), a rigid magnetic disk and an optical disk, etc.

[0054] Each computer program may be tangibly stored in a machine-readable storage media or device (e.g., program memory or magnetic disk) readable by a general or special purpose programmable computer, for configuring and controlling operation of a computer when the storage media or device is read by the computer to perform the procedures described herein. The inventive system may also be considered to be embodied in a computer-readable storage medium, configured with a computer program, where the storage medium so configured causes a computer to operate in a specific and predefined manner to perform the functions described herein.

[0055] A data processing system suitable for storing and/or executing program code may include at least one processor coupled directly or indirectly to memory elements

through a system bus. The memory elements can include local memory employed during actual execution of the program code, bulk storage, and cache memories which provide temporary storage of at least some program code to reduce the number of times code is retrieved from bulk storage during execution. Input/output or I/O devices (including but not limited to keyboards, displays, pointing devices, etc.) may be coupled to the system either directly or through intervening I/O controllers.

[0056] Network adapters may also be coupled to the system to enable the data processing system to become coupled to other data processing systems or remote printers or storage devices through intervening private or public networks. Modems, cable modem and Ethernet cards are just a few of the currently available types of network adapters.

[0057] As employed herein, the term "hardware processor subsystem" or "hardware processor" can refer to a processor, memory, software or combinations thereof that cooperate to perform one or more specific tasks. In useful embodiments, the hardware processor subsystem can include one or more data processing elements (e.g., logic circuits, processing circuits, instruction execution devices, etc.). The one or more data processing elements can be included in a central processing unit, a graphics processing unit, and/or a separate processor- or computing element-based controller (e.g., logic gates, etc.). The hardware processor subsystem can include one or more on-board memories (e.g., caches, dedicated memory arrays, read only memory, etc.). In some embodiments, the hardware processor subsystem can include one or more memories that can be on or off board or that can be dedicated for use by the hardware processor subsystem (e.g., ROM, RAM, basic input/output system (BIOS), etc.).

[0058] In some embodiments, the hardware processor subsystem can include and execute one or more software elements. The one or more software elements can include

an operating system and/or one or more applications and/or specific code to achieve a specified result.

[0059] In other embodiments, the hardware processor subsystem can include dedicated, specialized circuitry that performs one or more electronic processing functions to achieve a specified result. Such circuitry can include one or more application-specific integrated circuits (ASICs), field-programmable gate arrays (FPGAs), and/or programmable logic arrays (PLAs).

[0060] These and other variations of a hardware processor subsystem are also contemplated in accordance with embodiments of the present invention.

[0061] FIG. 5 is a block diagram showing an exemplary computing device 500, in accordance with an embodiment of the present invention. The computing device 500 is configured to identify a top-down parametric representation of an indoor scene and provide navigation through the scene.

[0062] The computing device 500 may be embodied as any type of computation or computer device capable of performing the functions described herein, including, without limitation, a computer, a server, a rack based server, a blade server, a workstation, a desktop computer, a laptop computer, a notebook computer, a tablet computer, a mobile computing device, a wearable computing device, a network appliance, a web appliance, a distributed computing system, a processor- based system, and/or a consumer electronic device. Additionally or alternatively, the computing device 500 may be embodied as a one or more compute sleds, memory sleds, or other racks, sleds, computing chassis, or other components of a physically disaggregated computing device.

[0063] As shown in FIG. 5, the computing device 500 illustratively includes the processor 510, an input/output subsystem 520, a memory 530, a data storage device

540, and a communication subsystem 550, and/or other components and devices commonly found in a server or similar computing device. The computing device 500 may include other or additional components, such as those commonly found in a server computer (e.g., various input/output devices), in other embodiments. Additionally, in some embodiments, one or more of the illustrative components may be incorporated in, or otherwise form a portion of, another component. For example, the memory 530, or portions thereof, may be incorporated in the processor 510 in some embodiments.

[0064] The processor 510 may be embodied as any type of processor capable of performing the functions described herein. The processor 510 may be embodied as a single processor, multiple processors, a Central Processing Unit(s) (CPU(s)), a Graphics Processing Unit(s) (GPU(s)), a single or multi-core processor(s), a digital signal processor(s), a microcontroller(s), or other processor(s) or processing/controlling circuit(s).

[0065] The memory 530 may be embodied as any type of volatile or non-volatile memory or data storage capable of performing the functions described herein. In operation, the memory 530 may store various data and software used during operation of the computing device 500, such as operating systems, applications, programs, libraries, and drivers. The memory 530 is communicatively coupled to the processor 510 via the I/O subsystem 520, which may be embodied as circuitry and/or components to facilitate input/output operations with the processor 510, the memory 530, and other components of the computing device 500. For example, the I/O subsystem 520 may be embodied as, or otherwise include, memory controller hubs, input/output control hubs, platform controller hubs, integrated control circuitry, firmware devices, communication links (e.g., point-to-point links, bus links, wires,

cables, light guides, printed circuit board traces, etc.), and/or other components and subsystems to facilitate the input/output operations. In some embodiments, the I/O subsystem 520 may form a portion of a system-on-a-chip (SOC) and be incorporated, along with the processor 510, the memory 530, and other components of the computing device 500, on a single integrated circuit chip.

[0066] The data storage device 540 may be embodied as any type of device or devices configured for short-term or long-term storage of data such as, for example, memory devices and circuits, memory cards, hard disk drives, solid state drives, or other data storage devices. The data storage device 540 can store program code 540A for generating peptide sequences. The communication subsystem 550 of the computing device 500 may be embodied as any network interface controller or other communication circuit, device, or collection thereof, capable of enabling communications between the computing device 500 and other remote devices over a network. The communication subsystem 550 may be configured to use any one or more communication technology (e.g., wired or wireless communications) and associated protocols (e.g., Ethernet, InfiniBand®, Bluetooth®, Wi-Fi®, WiMAX, etc.) to effect such communication.

[0067] As shown, the computing device 500 may also include one or more peripheral devices 560. The peripheral devices 560 may include any number of additional input/output devices, interface devices, and/or other peripheral devices. For example, in some embodiments, the peripheral devices 560 may include a display, touch screen, graphics circuitry, keyboard, mouse, speaker system, microphone, network interface, and/or other input/output devices, interface devices, and/or peripheral devices.

[0068] Of course, the computing device 500 may also include other elements (not shown), as readily contemplated by one of skill in the art, as well as omit certain elements. For example, various other sensors, input devices, and/or output devices can be included in computing device 500, depending upon the particular implementation of the same, as readily understood by one of ordinary skill in the art. For example, various types of wireless and/or wired input and/or output devices can be used. Moreover, additional processors, controllers, memories, and so forth, in various configurations can also be utilized. These and other variations of the processing system 500 are readily contemplated by one of ordinary skill in the art given the teachings of the present invention provided herein.

[0069] These and other variations of a hardware processor subsystem are also contemplated in accordance with embodiments of the present invention.

[0070] Referring now to FIG. 6, additional detail is shown on the peptide sequence generation 540A. An autoencoder 606 is trained by autoencoder training 604, using a set of training data 602. The training data 602 may be stored, for example in the memory 530 and may be accessed by the peptide sequence generation 540A. As described above, autoencoder training 604 may use the training data 602 as inputs to the autoencoder 606, and may compare the training data 602 to reconstructed outputs from the autoencoder 606, varying parameters of the autoencoder 606 in accordance with the comparison.

[0071] During operation, a new peptide input 608 may be applied to the autoencoder 606. Modifications may be made to the disentangled representations, between the operation of the encoder and the decoder of the autoencoder 606. When the decoder of the autoencoder 606 operates on the modified disentangled representations, a new peptide sequence 612 may be generated.

[0072] The autoencoder 606 may be implemented in the form of a neural network. In particular, the encoder part and the decoder part may be implemented as respective neural networks of any appropriate depth, with the parameters of each being set to effect the transformation of peptide sequences into embedded representations and the transformation of embedded representations into peptide sequences.

[0073] Referring now to FIG. 7, an exemplary neural network architecture is shown. In layered neural networks, nodes are arranged in the form of layers. A simple neural network has an input layer 720 of source nodes 722, a single computation layer 730 having one or more computation nodes 732 that also act as output nodes, where there is a single node 732 for each possible category into which the input example could be classified. An input layer 720 can have a number of source nodes 722 equal to the number of data values 712 in the input data 710. The data values 712 in the input data 710 can be represented as a column vector. Each computational node 730 in the computation layer generates a linear combination of weighted values from the input data 710 fed into input nodes 720, and applies a non-linear activation function that is differentiable to the sum. The simple neural network can perform classification on linearly separable examples (e.g., patterns).

[0074] Referring now to FIG. 8, a deep neural network architecture is shown. A deep neural network, also referred to as a multilayer perceptron, has an input layer 720 of source nodes 722, one or more computation layer(s) 730 having one or more computation nodes 732, and an output layer 740, where there is a single output node 742 for each possible category into which the input example could be classified. An input layer 720 can have a number of source nodes 722 equal to the number of data values 712 in the input data 710. The computation nodes 732 in the computation layer(s) 730 can also be referred to as hidden layers because they are between the source

nodes 722 and output node(s) 742 and not directly observed. Each node 732, 742 in a computation layer generates a linear combination of weighted values from the values output from the nodes in a previous layer, and applies a non-linear activation function that is differentiable to the sum. The weights applied to the value from each previous node can be denoted, for example, by w_1, w_2, w_{n-1} w_n. The output layer provides the overall response of the network to the inputted data. A deep neural network can be fully connected, where each node in a computational layer is connected to all other nodes in the previous layer. If links between nodes are missing the network is referred to as partially connected.

[0075] Training a deep neural network can involve two phases, a forward phase where the weights of each node are fixed and the input propagates through the network, and a backwards phase where an error value is propagated backwards through the network.

[0076] The computation nodes 732 in the one or more computation (hidden) layer(s) 730 perform a nonlinear transformation on the input data 712 that generates a feature space. The feature space the classes or categories may be more easily separated than in the original data space.

[0077] The neural network architectures of FIGs. 7 and 8 may be used to implement, for example, any of the models shown in FIG. 2. To train a neural network, training data can be divided into a training set and a testing set. The training data includes pairs of an input and a known output. During training, the inputs of the training set are fed into the neural network using feed-forward propagation. After each input, the output of the neural network is compared to the respective known output. Discrepancies between the output of the neural network and the known output that is associated with that particular input are used to generate an error value, which may be backpropagated

through the neural network, after which the weight values of the neural network may be updated. This process continues until the pairs in the training set are exhausted.

[0078] After the training has been completed, the neural network may be tested against the testing set, to ensure that the training has not resulted in overfitting. If the neural network can generalize to new inputs, beyond those which it was already trained on, then it is ready for use. If the neural network does not accurately reproduce the known outputs of the testing set, then additional training data may be needed, or hyperparameters of the neural network may need to be adjusted.

[0079] Reference in the specification to "one embodiment" or "an embodiment" of the present invention, as well as other variations thereof, means that a particular feature, structure, characteristic, and so forth described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, the appearances of the phrase "in one embodiment" or "in an embodiment", as well any other variations, appearing in various places throughout the specification are not necessarily all referring to the same embodiment. However, it is to be appreciated that features of one or more embodiments can be combined given the teachings of the present invention provided herein.

[0080] It is to be appreciated that the use of any of the following "/", "and/or", and "at least one of", for example, in the cases of "A/B", "A and/or B" and "at least one of A and B", is intended to encompass the selection of the first listed option (A) only, or the selection of the second listed option (B) only, or the selection of both options (A and B). As a further example, in the cases of "A, B, and/or C" and "at least one of A, B, and C", such phrasing is intended to encompass the selection of the first listed option (A) only, or the selection of the second listed option (B) only, or the selection of the third listed option (C) only, or the selection of the first and the second listed

options (A and B) only, or the selection of the first and third listed options (A and C) only, or the selection of the second and third listed options (B and C) only, or the selection of all three options (A and B and C). This may be extended for as many items listed.

[0081] The foregoing is to be understood as being in every respect illustrative and exemplary, but not restrictive, and the scope of the invention disclosed herein is not to be determined from the Detailed Description, but rather from the claims as interpreted according to the full breadth permitted by the patent laws. It is to be understood that the embodiments shown and described herein are only illustrative of the present invention and that those skilled in the art may implement various modifications without departing from the scope and spirit of the invention. Those skilled in the art could implement various other feature combinations without departing from the scope and spirit of the invention. Having thus described aspects of the invention, with the details and particularity required by the patent laws, what is claimed and desired protected by Letters Patent is set forth in the appended claims.

WHAT IS CLAIMED IS:

1. A computer-implemented method of generating a peptide sequence, comprising:

transforming (208) an input peptide sequence into disentangled representations, including a structural representation and an attribute representation, using an autoencoder model;

modifying (210) one of the disentangled representations; and

transforming (212) the disentangled representations, including the modified disentangled representation, to generate a new peptide sequence using the autoencoder model.

2. The computer-implemented method of claim 1, further comprising training a neural network of the autoencoder model using a set of training peptide sequences.

3. The computer-implemented method of claim 2, wherein training the neural network of the autoencoder model includes minimizing a mutual information between the structural representation and the attribute representation.

4. The computer-implemented method of claim 1, wherein modifying the disentangled representations includes modifying a binding affinity.

5. The computer-implemented method of claim 4, wherein the binding affinity is a binding affinity between a peptide and a major histocompatibility complex.

6. The computer-implemented method of claim 1, wherein modifying the disentangled representations includes modifying an antigen processing score.

7. The computer-implemented method of claim 1, wherein modifying the disentangled representations includes modifying a T-cell receptor interaction score.

8. The computer-implemented method of claim 1, wherein modifying the disentangled representations includes altering an attribute to improve vaccine efficacy against a predetermined pathogen.

9. The computer-implemented method of claim 1, wherein modifying the disentangled representations includes changing coordinates of a vector representation of the disentangled representations within an embedding space.

10. The computer-implemented method of claim 1, wherein transforming the input peptide sequence is performed using an encoder of the autoencoder model and transforming the disentangled representations is performed using a decoder of the autoencoder model.

11. A computer-implemented method of generating a peptide sequence, comprising:

training (410) a Wasserstein neural network model using a set of training peptide sequences by minimizing a mutual information between a structural representation and an attribute representation of the training peptide sequences;

transforming (208) an input peptide sequence into disentangled structural and attribute representations, using an encoder of the Wasserstein autoencoder neural network model;

modifying (210) one of the disentangled representations to alter an attribute to improve vaccine efficacy against a predetermined pathogen, including changing coordinates of a vector representation of the disentangled representations within an embedding space; and

transforming (212) the disentangled representations, including the modified disentangled representation, to generate a new peptide sequence using a decoder of the Wasserstein autoencoder neural network model.

12. A system for generating a peptide sequence, comprising:

a hardware processor (510); and

a memory (530) that stores a computer program product (540A), which, when executed by the hardware processor, causes the hardware processor to:

transform (208) an input peptide sequence into disentangled representations, including a structural representation and an attribute representation, using an autoencoder model;

modify (210) one of the disentangled representations; and

transform (212) the disentangled representations, including the modified disentangled representation, to generate a new peptide sequence using the autoencoder model.

13. The system of claim 12, wherein the computer program product further causes the hardware processor to train a neural network of the autoencoder model using a set of training peptide sequences.

14. The system of claim 13, wherein the computer program product further causes the hardware processor to minimize a mutual information between the structural representation and the attribute representation.

15. The system of claim 12, wherein the computer program product further causes the hardware processor to modify a binding affinity.

16. The system of claim 12, wherein the computer program product further causes the hardware processor to modify an antigen processing score.

17. The system of claim 12, wherein the computer program product further causes the hardware processor to modify a T-cell receptor interaction score.

18. The system of claim 12, wherein the computer program product further causes the hardware processor to alter an attribute to improve vaccine efficacy against a predetermined pathogen.

19. The system of claim 12, wherein the computer program product further causes the hardware processor to change coordinates of a vector representation of the disentangled representations within an embedding space.

20. The system of claim 12, wherein transformation of the input peptide sequence is performed using an encoder of the autoencoder model and transformation of the disentangled representations is performed using a decoder of the autoencoder model.

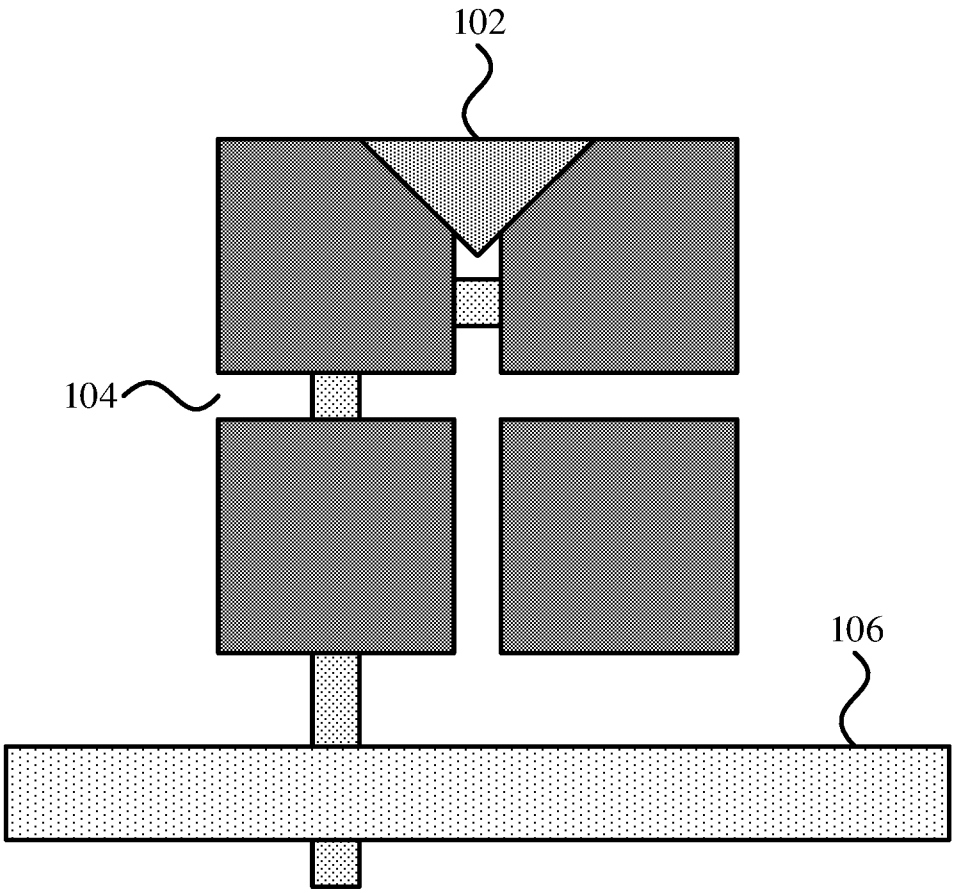

FIG. 1

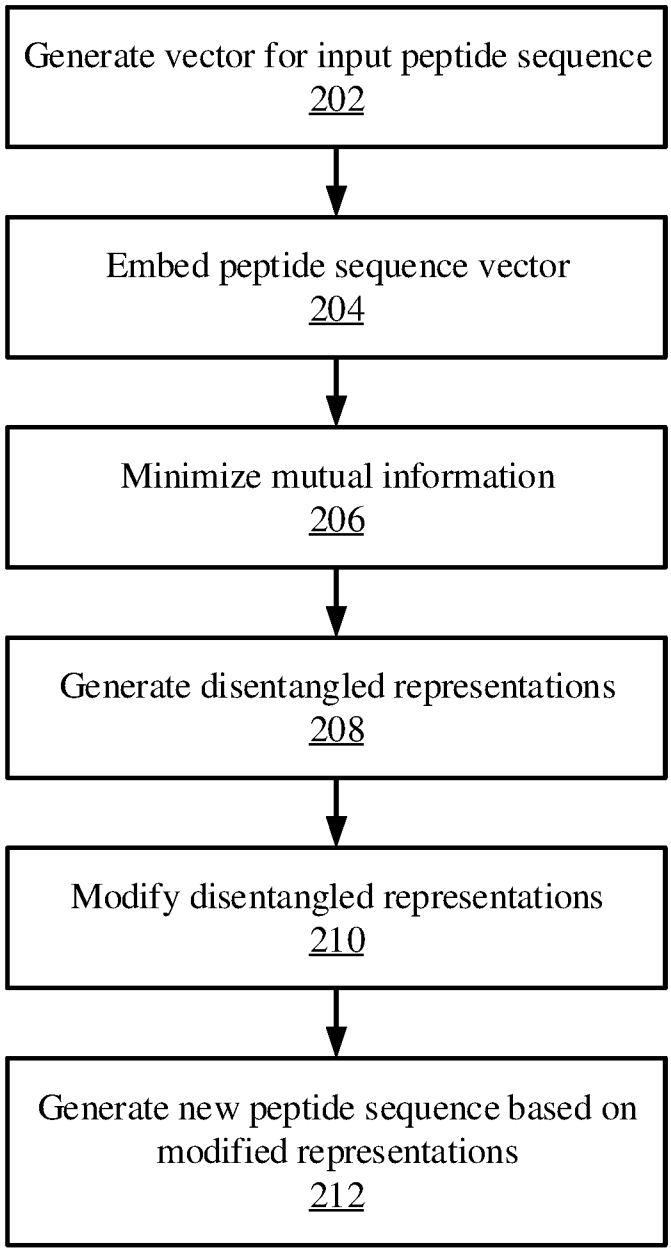
Generate vector for input peptide sequence
202
Embed peptide sequence vector
204
Minimize mutual information
206
Generate disentangled representations
208
Modify disentangled representations
210
Generate new peptide sequence based on modified representations
212

FIG. 2

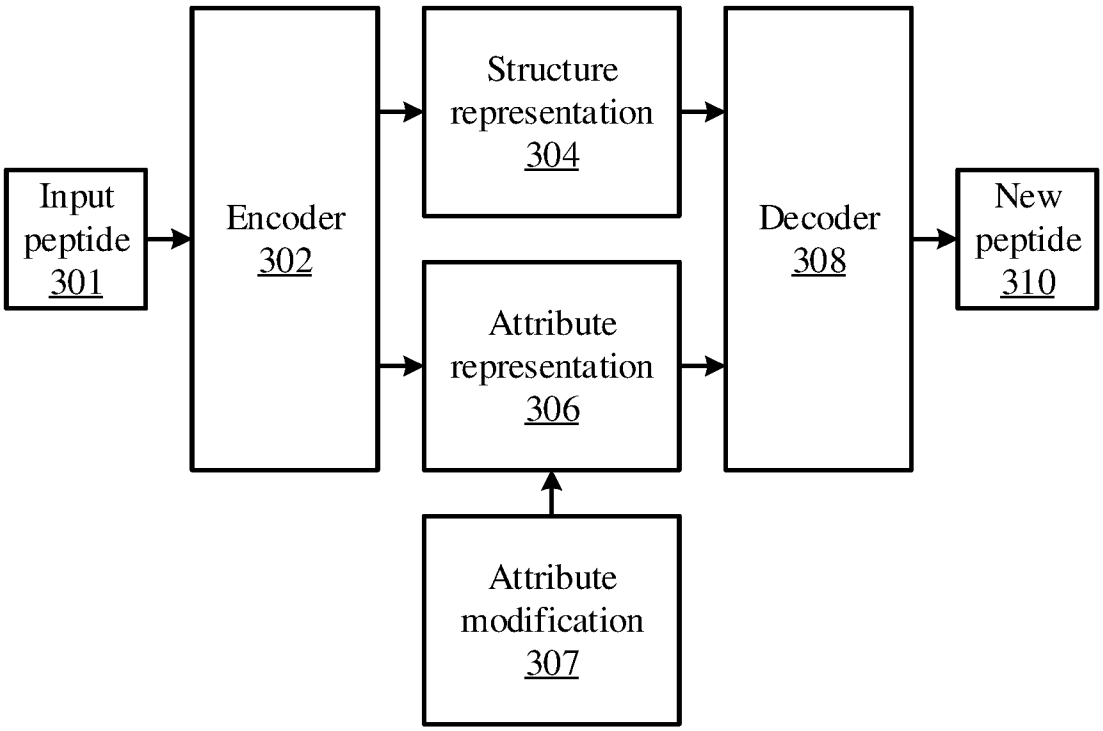
Input peptide 301
Encoder 302
Structure representation 304
Attribute representation 306
Attribute modification 307
Decoder 308
New peptide 310

FIG. 3

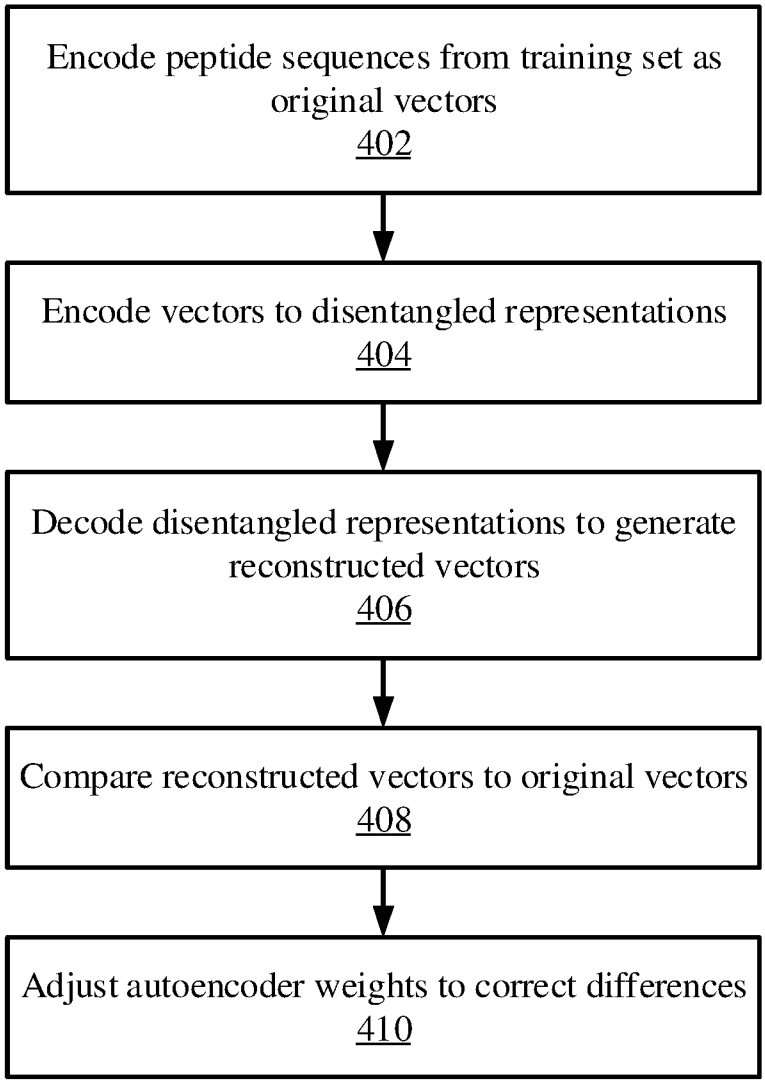
Encode peptide sequences from training set as original vectors
402
Encode vectors to disentangled representations
404
Decode disentangled representations to generate reconstructed vectors
406
Compare reconstructed vectors to original vectors
408
Adjust autoencoder weights to correct differences
410

FIG. 4

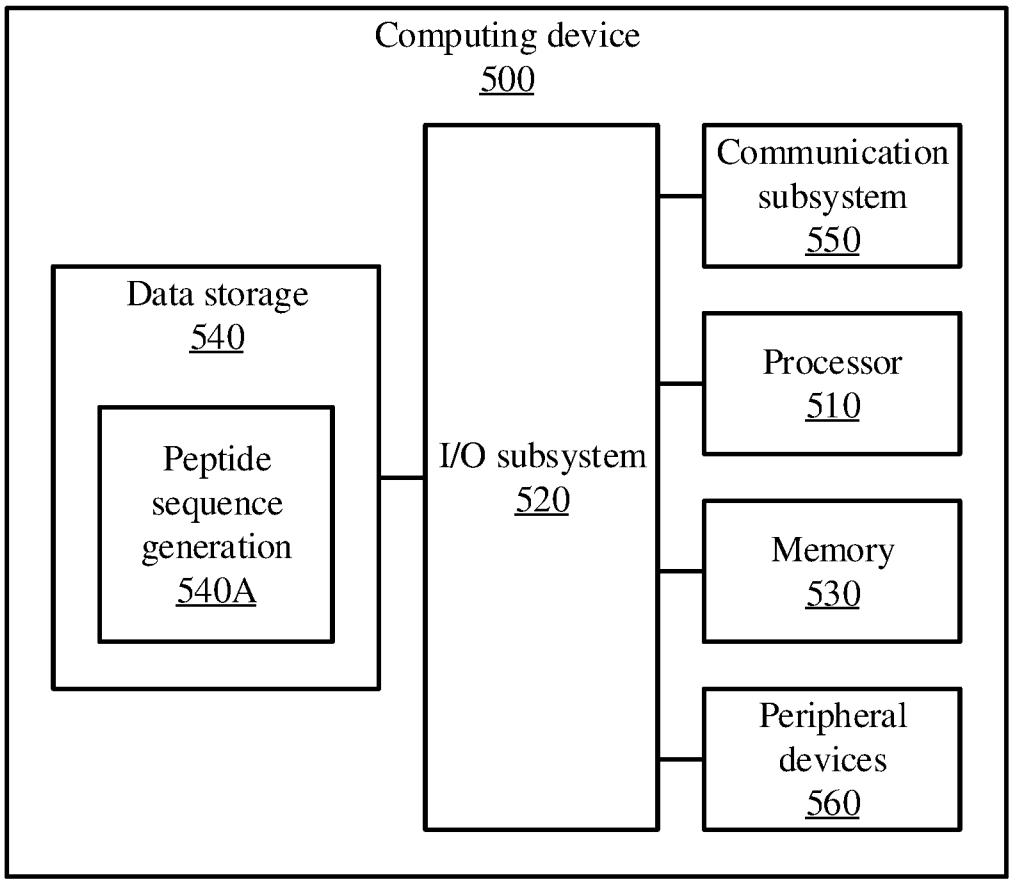
Computing device
500
Communication subsystem
550
Data storage
540
Processor
510
Peptide sequence generation
540A
I/O subsystem
520
Memory
530
Peripheral devices
560

FIG. 5

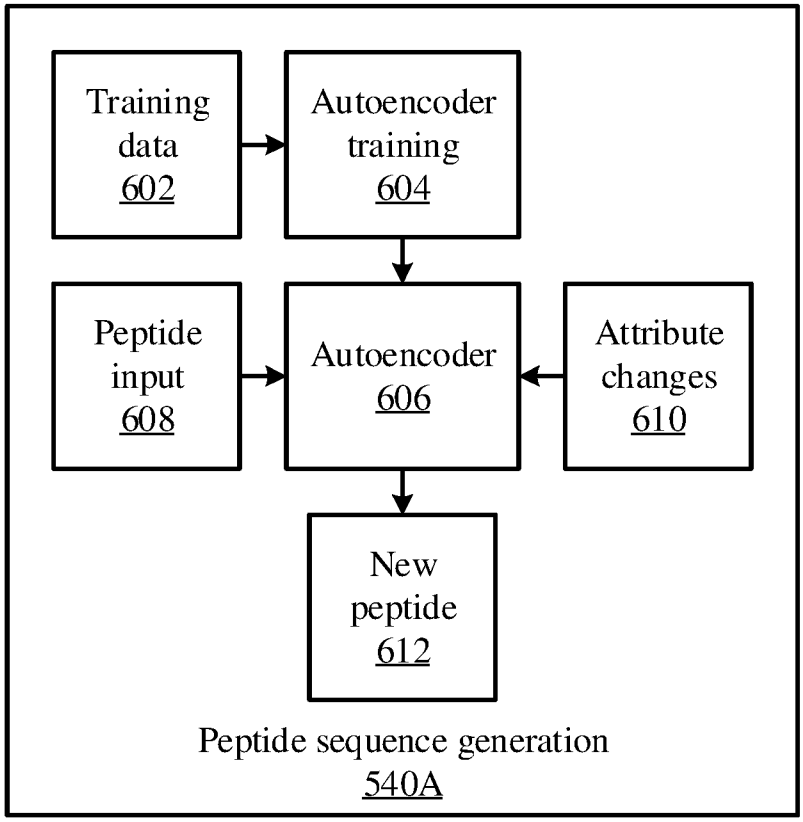
Training data 602
Autoencoder training 604
Peptide input 608
Autoencoder 606
Attribute changes 610
New peptide 612
Peptide sequence generation 540A

FIG. 6

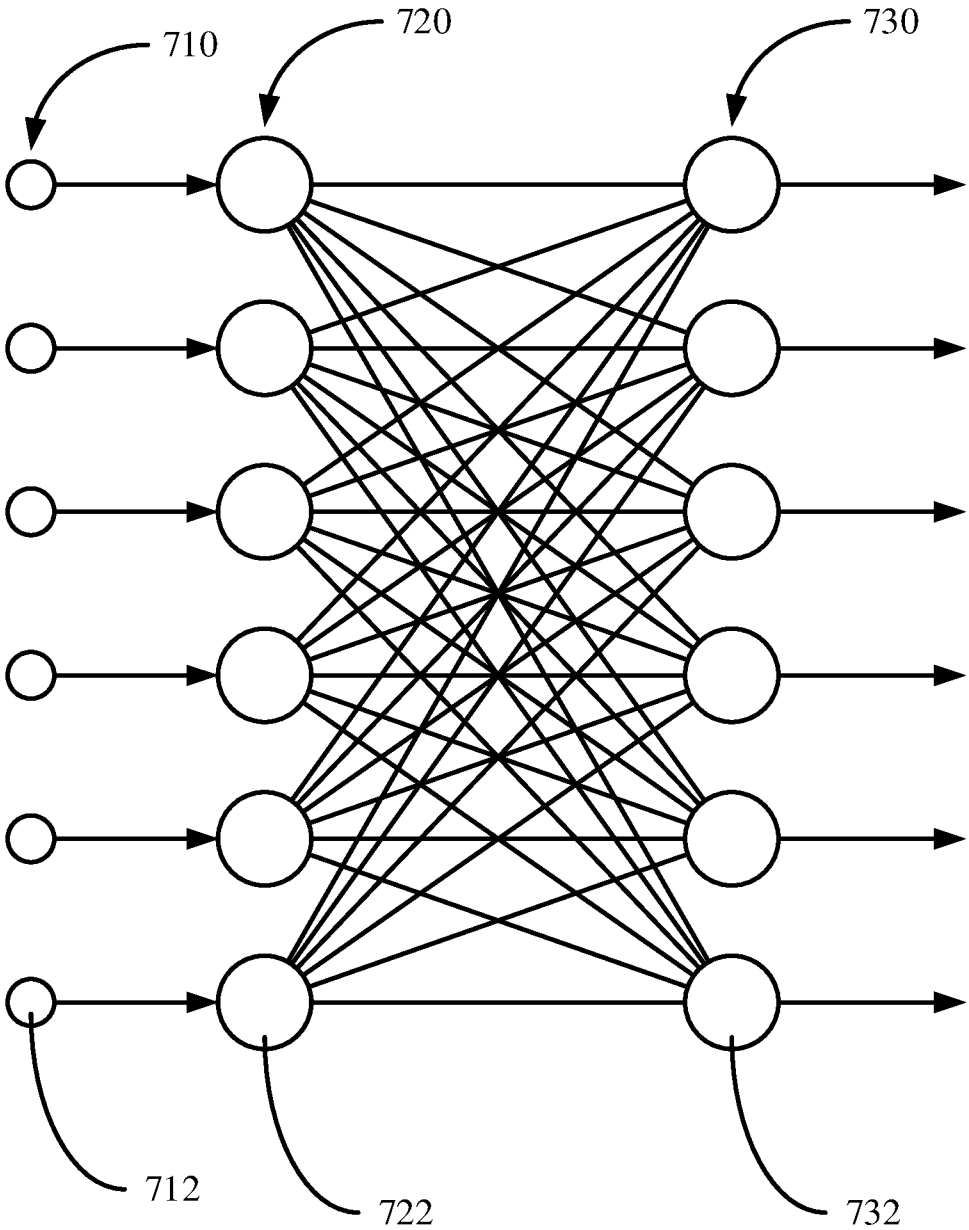

FIG. 7

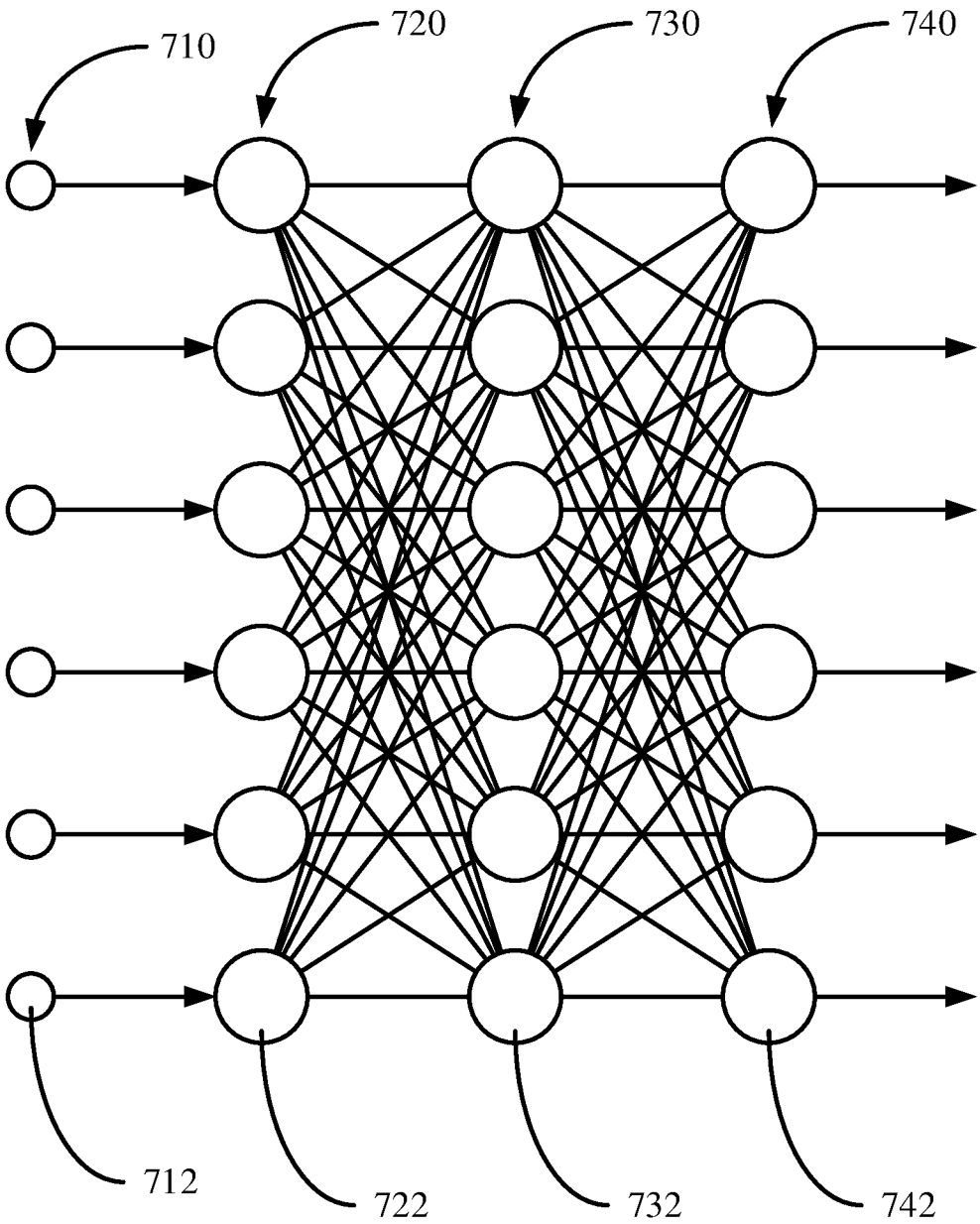

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2021/056879

A. CLASSIFICATION OF SUBJECT MATTER

G16B 30/20(2019.01)i; **G16B 15/20**(2019.01)i; **G16B 40/20**(2019.01)i; **G16B 35/10**(2019.01)i; **G06N 3/08**(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G16B 30/20(2019.01); A61K 39/395(2006.01); C07H 21/04(2006.01); C07K 5/00(2006.01); G06F 15/18(2006.01); G06F 19/12(2011.01); G16B 30/00(2019.01); G16B 40/20(2019.01)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: peptide, vaccine, disentangled representations, structural representation, attribute representation, autoencoder, wasserstein neural network

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2013-0330335 A1 (BREMEL, R. D. et al.) 12 December 2013 (2013-12-12) abstract; claims 1-32; figure 1	1-20
A	US 2019-0311781 A1 (ONCOIMMUNITY AS) 10 October 2019 (2019-10-10) claims 1-9	1-20
A	US 2015-0205911 A1 (BAYER HEALTHCARE LLC) 23 July 2015 (2015-07-23) claims 1-13	1-20
A	US 2008-0071706 A1 (HONDA, H. et al.) 20 March 2008 (2008-03-20) the whole document	1-20
A	MALONIS, R. J. et al., `Peptide-Based Vaccines: Current Progress and Future Challenges', Chem. Rev., 05 December 2019, vol. 120, pp. 3210-3229 the whole document	1-20

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
23 February 2022	**23 February 2022**

Name and mailing address of the ISA/KR	Authorized officer
Korean Intellectual Property Office **189 Cheongsa-ro, Seo-gu, Daejeon** **35208, Republic of Korea**	**HEO, Joo Hyung**
Facsimile No. **+82-42-481-8578**	Telephone No. **+82-42-481-5373**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US2021/056879

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
US 2013-0330335 A1	12 December 2013	EP 2550529 A1	30 January 2013
		EP 2550529 B1	17 November 2021
		EP 2771349 A2	03 September 2014
		EP 2771349 B1	26 February 2020
		US 10706955 B2	07 July 2020
		US 2017-0039314 A1	09 February 2017
		US 2021-0375393 A1	02 December 2021
		WO 2011-119484 A1	29 September 2011
		WO 2013-040142 A2	21 March 2013
		WO 2013-040142 A3	13 February 2014
US 2019-0311781 A1	10 October 2019	CA 3022390 A1	02 November 2017
		CN 109416929 A	01 March 2019
		EP 3449405 A1	06 March 2019
		JP 2019-518295 A	27 June 2019
		JP 6953515 B2	27 October 2021
		WO 2017-186959 A1	02 November 2017
US 2015-0205911 A1	23 July 2015	AU 2013-266850 A1	04 December 2014
		BR 112014029235 A2	29 January 2019
		CA 2874542 A1	28 November 2013
		CN 104487979 A	01 April 2015
		EP 2856374 A1	08 April 2015
		IL 235815 A	29 January 2015
		JP 2015-526775 A	10 September 2015
		KR 10-2015-0021528 A	02 March 2015
		MX 2014014199 A	12 February 2015
		NZ 702084 A	30 September 2016
		RU 2014152463 A	20 July 2016
		SG 11201407459 A	30 December 2014
		WO 2013-176756 A1	28 November 2013
US 2008-0071706 A1	20 March 2008	JP 2008-063285 A	21 March 2008