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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 June 2016 (09.06.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/089711 A1

(51) International Patent Classification:
G06N 99/00 (2010.01)

(21) International Application Number:
PCT/US2015/062848

(22) International Filing Date:
28 November 2015 (28.11.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/088,409 5 December 2014 (05.12.2014) US

(71) Applicant: **MICROSOFT TECHNOLOGY LICENSING, LLC** [US/US]; Attn: Patent Group Docketing (Bldg. 8/1000), One Microsoft Way, Redmond, Washington 98052-6399 (US).

(72) Inventors: **WIEBE, Nathan**; Microsoft Technology Licensing, Llc, Attn: Patent Group Docketing (Bldg. 8/1000), One Microsoft Way, Redmond, Washington 98052-6399 (US). **SVORE, Krysta**; Microsoft Technology Licensing, Llc, Attn: Patent Group Docketing (Bldg. 8/1000), One Microsoft Way, Redmond, Washington 98052-6399 (US). **KAPOOR, Ashish**; Microsoft Technology Licensing, Llc, Attn: Patent Group Docketing (Bldg. 8/1000), One Microsoft Way, Redmond, Washington 98052-6399 (US).

(74) Agents: **MINHAS, Sandip** et al.; MICROSOFT CORPORATION, Attn: Patent Group Docketing (Bldg. 8/1000), One Microsoft Way, Redmond, Washington 98052-6399 (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, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, 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, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, 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).

[Continued on next page]

(54) Title: QUANTUM DEEP LEARNING

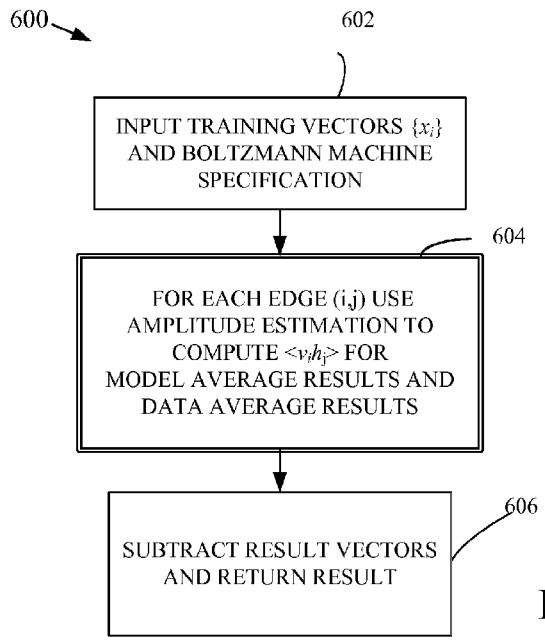


FIG. 6

(57) Abstract: Boltzmann machines are trained using an objective function that is evaluated by sampling quantum states that approximate a Gibbs state. Classical processing is used to produce the objective function, and the approximate Gibbs state is based on weights and biases that are refined using the sample results. In some examples, amplitude estimation is used. A combined classical/quantum computer produces suitable weights and biases for classification of shapes and other applications.

WO 2016/089711 A1

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

QUANTUM DEEP LEARNING

Technical Field

[001] The disclosure pertains to training Boltzmann machines using quantum computers.

Background

[002] Deep learning is a relatively new paradigm for machine learning that has substantially impacted the way in which classification, inference and artificial intelligence (AI) tasks are performed. Deep learning began with the suggestion that in order to perform sophisticated AI tasks, such as vision or language, it may be necessary to work on abstractions of the initial data rather than raw data. For example, an inference engine that is trained to detect a car might first take a raw image and decompose it first into simple shapes. These shapes could form the first layer of abstraction. These elementary shapes could then be grouped together into higher level abstract objects such as bumpers or wheels. The problem of determining whether a particular image is or is not a car is then performed on the abstract data rather than the raw pixel data. In general, this process could involve many levels of abstraction.

[003] Deep learning techniques have demonstrated remarkable improvements such as up to 30% relative reduction in error rate on many typical vision and speech tasks. In some cases, deep learning techniques approach human performance, such as in matching two faces. Conventional classical deep learning methods are currently deployed in language models for speech and search engines. Other applications include machine translation and deep image understanding (i.e., image to text representation).

[004] Existing methods for training deep belief networks use contrastive divergence approximations to train the network layer by layer. This process is expensive for deep networks, relies on the validity of the contrastive divergence approximation, and precludes the use of intra-layer connections. The contrastive divergence approximation is inapplicable in some applications, and in any case, contrastive divergence based methods are incapable of training an entire graph at once and instead rely on training the system one layer at a time, which is costly and reduces the quality of the model. Finally, further crude approximations are needed to train a full Boltzmann machine, which potentially has connections between all hidden and visible units and may limit the quality of the optima found in the learning algorithm. Approaches are needed that overcome these limitations.

Summary

[005] The disclosure provides methods and apparatus for training deep belief networks in

machine learning. The disclosed methods and apparatus permit efficient training of generic Boltzmann machines that are currently untrainable with conventional approaches. In addition, the disclosed approaches can provide more rapid training in fewer steps. Gradients of objective functions for deep Boltzmann machines are determined using a quantum computer in combination with a classical computer. A quantum state encodes an approximation to a Gibbs distribution, and sampling of this approximate distribution is used to determine Boltzmann machine weights and biases. In some cases, amplitude estimation and fast quantum algorithms are used. Typically, a classical computer receives a specification of a Boltzmann machine and associated training data, and determines an objective function associated with the Boltzmann machine. A quantum computer determines at least one gradient of the objective function, and based on the gradient of the objective function, at least one hidden value or a weight of the Boltzmann machine is established. A mean-field approximation can be used to define an objective function, and gradients can be determined based on the sampling.

[006] These and other features of the disclosure are set forth below with reference to the accompanying drawings.

Brief Description of the Drawings

[007] FIG. 1 illustrates a representative example of a deep Boltzmann machine.

[008] FIG. 2 illustrates a representative method of training a Boltzmann machine, typically based on an objective function associated with a log-likelihood.

[009] FIG. 3 illustrates a method of gradient calculation for a deep Boltzmann machine using quantum-based sampling.

[010] FIG. 4 illustrates a method of quantum-based sampling for determining a model average for use in training a Boltzmann machine using quantum computation.

[011] FIG. 5 illustrates a method of quantum-based sampling for determining a data average for use in training a Boltzmann machine using quantum computation.

[012] FIG. 6 illustrates a method of gradient calculation for a deep Boltzmann machine using amplitude estimation in a quantum computer.

[013] FIG. 7 illustrates a method of determining a model average for a Boltzmann machine using amplitude estimation in a quantum computer.

[014] FIG. 8 illustrates an alternative method of determining a model average for a Boltzmann machine using amplitude estimation in a quantum computer.

[015] FIG. 9 illustrates a representative processor-based quantum circuit design environment for training a deep Boltzmann machine.

[016] FIG. 10 illustrates a representative classical computer that produces a quantum circuit arrangement that is coupled to a quantum processor so as to produce a quantum state that approximates a Gibbs distribution.

Detailed Description

[017] As used in this application and in the claims, the singular forms "a," "an," and "the" include the plural forms unless the context clearly dictates otherwise. Additionally, the term "includes" means "comprises." Further, the term "coupled" does not exclude the presence of intermediate elements between the coupled items.

[018] The systems, apparatus, and methods described herein should not be construed as limiting in any way. Instead, the present disclosure is directed toward all novel and non-obvious features and aspects of the various disclosed embodiments, alone and in various combinations and sub-combinations with one another. The disclosed systems, methods, and apparatus are not limited to any specific aspect or feature or combinations thereof, nor do the disclosed systems, methods, and apparatus require that any one or more specific advantages be present or problems be solved. Any theories of operation are to facilitate explanation, but the disclosed systems, methods, and apparatus are not limited to such theories of operation.

[019] Although the operations of some of the disclosed methods are described in a particular, sequential order for convenient presentation, it should be understood that this manner of description encompasses rearrangement, unless a particular ordering is required by specific language set forth below. For example, operations described sequentially may in some cases be rearranged or performed concurrently. Moreover, for the sake of simplicity, the attached figures may not show the various ways in which the disclosed systems, methods, and apparatus can be used in conjunction with other systems, methods, and apparatus. Additionally, the description sometimes uses terms like "produce" and "provide" to describe the disclosed methods. These terms are high-level abstractions of the actual operations that are performed. The actual operations that correspond to these terms will vary depending on the particular implementation and are readily discernible by one of ordinary skill in the art.

[020] In some examples, values, procedures, or apparatus' are referred to as "lowest", "best", "minimum," or the like. It will be appreciated that such descriptions are intended to indicate that a selection among many functional alternatives can be made, and such selections need not be better, smaller, or otherwise preferable to other selections.

[021] The methods and apparatus described herein generally use a classical computer coupled to a quantum computer to train a deep Boltzmann machine. In order for the classical computer to update a model for the deep Boltzmann machine given training data, certain expectation values are computed. A quantum computer is arranged to accelerate this process. In typical examples, a classically tractable approximation to the state provided by a mean field approximation, or a related approximation, is used to prepare a quantum state that is close to the distribution that yields the desired expectation values. The quantum computer is then used to efficiently refine this approximation into precisely the desired distribution. The required expectation values are then learned by sampling from this quantum distribution.

[022] In alternative examples, amplitude estimation is used. Instead of preparing the quantum computer in a state corresponding to a single training vector, the state is prepared in a quantum superposition of every training example in the set Amplitude estimation is used to find the required expectation values.

Boltzmann Machines

[023] The Boltzmann machine is a powerful paradigm for machine learning in which the problem of training a system to classify or generate examples of a set of training vectors is reduced to the problem of energy minimization of a spin system. The Boltzmann machine consists of several binary units that are split into two categories: (a) visible units and (b) hidden units. The visible units are the units in which the inputs and outputs of the machine are given. For example, if a machine is used for classification, then the visible units will often be used to hold training data as well as a label for that training data. The hidden units are used to generate correlations between the visible units that enable the machine either to assign an appropriate label to a given training vector or to generate an example of the type of data that the system is trained to output. FIG. 1 illustrates a deep Boltzmann machine 100 that includes a visible input layer 102 for inputs v_i, and output layer 110 for outputs l_j, and hidden unit layers 104, 106, 108 that couple the visible input layer 102 and the visible output layer 104. The layers 102, 104, 106, 108, 110 can be connected to an adjacent layer with connections 103, 105, 107, 109 but in a deep Boltzmann machine such as shown in FIG. 1, there are no intralayer connections. However, the disclosed methods and apparatus can be used to train Boltzmann machines with such intralayer connections, but for convenient description, training of deep Boltzmann machines is described in detail.

[024] Formally, the Boltzmann machine models the probability of a given configuration (v,h) of hidden and visible units via the Gibbs distribution:

$$P(v,h) = e^{-E(v,h)} / Z,$$

wherein Z is a normalizing factor known as the partition function, and v,h refer to visible and hidden unit values, respectively. The energy E of a given configuration of hidden and visible units is of the form:

$$E(v,h) = \sum_i v_i b_i - \sum_j h_j d_j - \sum_{i,j} w_{ij} v_i h_j,$$

wherein vectors v and h are visible and hidden unit values, vectors b and d are biases that provide an energy penalty for a bit taking a value of *1* and $w_{i,j}$ is a weight that assigns an energy penalty for the hidden and visible units both taking on a value of *1*. Training a Boltzmann machine reduces to estimating these biases and weights by maximizing the log-likelihood of the training data. A Boltzmann machine for which the biases and weights have been determined is referred to as a trained Boltzmann machine. A so-called L2-regularization term can be added in order to prevent overfitting, resulting in the following form of an objective function:

$$O_{\mathrm{ML}} := \frac{1}{N_{\mathrm{train}}} \sum_{v \in x_{\mathrm{train}}} \log\left(\sum_h P(v,h)\right) - \frac{\lambda}{2} w^T w.$$

This objective function is referred to as a maximum likelihood-objective (ML-objective) function and λ represents the regularization term. Gradient descent provides a method to find a locally optimal value of the ML-objective function. Formally, the gradients of this objective function can be written as:

$$\frac{\partial O_{\mathrm{ML}}}{\partial w_{ij}} = \langle v_i h_j \rangle_{\mathrm{data}} - \langle v_i h_j \rangle_{\mathrm{model}} - \lambda w_{i,j} \tag{1a}$$

$$\frac{\partial O_{\mathrm{ML}}}{\partial b_i} = \langle v_i \rangle_{\mathrm{data}} - \langle v_i \rangle_{\mathrm{model}} \tag{1b}$$

$$\frac{\partial O_{\mathrm{ML}}}{\partial d_j} = \langle h_j \rangle_{\mathrm{data}} - \langle h_j \rangle_{\mathrm{model}}. \tag{1c}$$

The expectation values for a quantity $x(v,h)$ are given by:

$$\langle x \rangle_{\mathrm{data}} = \frac{1}{N_{\mathrm{train}}} \sum_{v \in x_{\mathrm{train}}} \sum_h \frac{x(v,h) e^{-E(\mathrm{v},h)}}{Z_v},$$

wherein $Z_v = \sum_h e^{-E(\mathrm{v},h)}$, and

$$\langle x \rangle_{\text{model}} = \sum_{v,h} \frac{x(v,h)e^{-E(v,h)}}{Z},$$

wherein $Z = \sum_{v,h} e^{-E(v_{data},h)}$.

[025] Note that it is non-trivial to compute any of these gradients: the value of the partition function Z is #P-hard to compute and cannot generally be efficiently approximated within a specified multiplicative error. This means modulo reasonable complexity theoretic assumptions, neither a quantum nor a classical computer should be able to directly compute the probability of a given configuration and in turn compute the log-likelihood of the Boltzmann machine yielding the particular configuration of hidden and visible units.

[026] In practice, approximations to the likelihood gradient via contrastive divergence or mean-field assumptions have been used. These conventional approaches, while useful, are not fully theoretically satisfying as the directions yielded by the approximations are not the gradients of any objective function, let alone the log-likelihood. Also, contrastive divergence does not succeed when trying to train a full Boltzmann machine which has arbitrary connections between visible and hidden units. The need for such connections can be mitigated by using a *deep restricted Boltzmann machine* (shown in FIG. 1) which organizes the hidden units in layers, each of which contains no intra-layer interactions or interactions with non-consecutive layers. The problem with this is that conventional methods use a greedy layer by layer approach to training that becomes costly for very deep networks with a large number of layers. Disclosed herein are methods and apparatus based on quantum computation for training deep restricted Boltzmann machines, without relying on the contrastive divergence approximation. The disclosed methods can be used with Boltzmann machines in general, and are not restricted to deep restricted Boltzmann machines. The approaches disclosed below are suitable for any objective function that can be efficiently computed based on sampling from a Gibbs Distribution.

[027] Boltzmann machines can be used in a variety of applications. In one application, data associated with a particular image, a series of images such as video, a text string, speech or other audio is provided to a Boltzmann machine (after training) for processing. In some cases, the Boltzmann provides a classification of the data example. For example, a Boltzmann machine can classify an input data example as containing an image of a face, speech in a particular language or from a particular individual, distinguish spam from desired email, or identify other patterns in the input data example such as identifying

shapes in an image. In other examples, the Boltzmann machine identifies other features in the input data example or other classifications associated with the data example. In still other examples, the Boltzmann machine preprocesses a data example so as to extract features that are to be provide to a subsequent Boltzmann machine. In typical examples, a trained Boltzmann machine can process data examples for classification, clustering into groups, or simplification such as by identifying topics in a set of documents. Data input to a Boltzmann machine for processing for these or other purposes is referred to as a data example. In some applications, a trained Boltzmann machine is used to generate output data corresponding to one or more features or groups of features associated with the Boltzmann machine. Such output data is referred to as an output data example. For example, a trained Boltzmann machine associated with facial recognition can produce an output data example that is corresponding to a model face.

Quantum algorithm for state preparation

[028] Quantum computers can draw unbiased samples from the Gibbs distribution, thereby allowing probabilities to be computed by sampling (or by quantum sampling). As disclosed herein, a quantum distribution is prepared that approximates the ideal probability distribution over the model or data. This approximate distribution is then refined by rejection sampling into a quantum distribution that is, to within numerical error, the target probability distribution. Layer by layer training is unnecessary, and approximations required in conventional methods can be avoided. Beginning with a uniform prior over the amplitudes of the Gibbs state, preparing the state via quantum rejection sampling is likely to be inefficient. This is because the success probability depends on a ratio of the partition functions of the initial state and the Gibbs state which is generally exponentially small for machine learning problems. In some examples, a mean-field approximation is used over the joint probabilities in the Gibbs state, rather than a uniform prior. This additional information can be used to boost the probability of success to acceptable levels for numerically tractable examples.

[029] The required expectation values can then be found by sampling from the quantum distribution. A number of samples needed to achieve a fixed sampling error can be quadratically reduced by using a quantum algorithm known as amplitude estimation. Disclosed below are methods by which an initial quantum distribution is refined into a quantum coherent Gibbs state (often called a coherent thermal state or CTS). Mean-field approaches or generalizations thereof can be used to provide suitable initial states for the quantum computer to refine into the CTS. All units are assumed to be binary-valued in the

following examples, but other units (such as Gaussian units) can be approximated within this framework by forming a single unit out of a string of several qubits.

Mean-Field Approximation

[030] The mean-field approximation to the joint probability distribution is referred to herein as $Q(v,h)$. The mean-field approximation is a variational approach that finds an uncorrelated distribution $Q(v,h)$ that has minimal Kullback-Leibler (KL) divergence with the joint probability distribution $P(v,h)$ given by the Gibbs distribution. The main benefit of using Q instead of P is that $\langle v_i h_j \rangle_{\text{model}}$ and $\log(Z)$ can be efficiently estimated using mean-field approximations. A secondary benefit is that the mean-field state can be efficiently prepared using single-qubit rotations.

[031] More concretely, the mean-field approximation is a distribution such that

$$Q(v,h) = \left(\prod_i \mu_i^{v_i} (1-\mu_i)^{1-v_i} \right) \left(\prod_j \nu_j^{h_j} (1-\nu_j)^{1-h_j} \right),$$

where μ_i and ν_j are chosen to minimize $\mathrm{KL}(Q \| P)$. The parameters μ_i and ν_j are called mean-field parameters.

[032] Using the properties of the Bernoulli distribution, it can be shown that:

$$\begin{aligned}
\mathrm{KL}(Q \| P) &= \sum_{v,h} -Q(v,h)\ln(P(v,h)) + Q(v,h)\ln(Q(v,h)), \\
&= \sum_{v,h} Q(v,h) \left(\sum_i v_i b_i + \sum_j h_j d_j + \sum_{i,j} w_{i,j} v_i h_j + \ln Z \right) + Q(v,h)\ln(Q(v,h)), \\
&= \sum_i \mu_i b_i + \sum_j \nu_j d_j + \sum_{i,j} w_{i,j} \mu_i \nu_j + \ln(Z) \\
&\quad + \sum_i \mu_i \ln(\mu_i) + (1-\mu_i)\ln(1-\mu_i) + \sum_j \nu_j \ln(\nu_j) + (1-\nu_j)\ln(1-\nu_j).
\end{aligned}$$

The optimal values of μ_i and ν_j are can be found by differentiating this equation with respect to μ_i and ν_j are and setting the result equal to zero. The solution to this is

$$\mu_i = \sigma(-b_i - \sum_j w_{i,j} \nu_j)$$

$$\nu_j = \sigma(-d_j - \sum_i w_{i,j} \mu_i),$$

wherein $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function.

[033] These equations can be implicitly solved by fixed point iteration, which involves initializing the μ_i and ν_j arbitrarily and iterating until convergence is reached.

Convergence is guaranteed provided that the norm of the Jacobian of the map is bounded above by 1. Solving the mean-field equations by fixed point iteration is analogous to Gibbs sampling with the difference being that here there are only a polynomial number of configurations to sample over and so the entire process is efficient.

[034] Mean-field approximations to distributions such $P(v,h) = \delta_{v,x} \exp^{-E(x,h)} / Z_x$ can be computed using the exact same methodology. The only difference is that in such cases the mean-field approximation is only taken over the hidden units. Such approximations are needed to compute the expectations over the data that are needed to estimate the derivatives of O_{ML} used below. It can also be shown that among all product distributions, Q is the distribution that leads to the least error in the approximation to the log-partition function.

[035] Experimentally, mean-field approximations can estimate the log-partition function within less than 1% error, depending on the weight distribution and the geometry of the graph used. The mean-field approximation to the partition function is sufficiently accurate for small restricted Boltzmann machines. Structured mean-field approximation methods can be used to reduce such errors if needed, albeit at a higher classical computational cost. It can be shown that the success probability of the disclosed state preparation methods approach unity in the limit in which the strengths of the correlations in the model vanish.

[036] The mean-field distribution is used to compute a variational approximation to the necessary partition functions. These approximations are shown below. If Q is the mean-field approximation to the Gibbs distribution P, then a mean field partition function Z_{MF} is defined as:

$$Z_{\mathrm{MF}} := \sum_{v,h} Q(v,h) \log\left(\frac{e^{-E(v,h)}}{Q(v,h)}\right).$$

Furthermore, for any $x \in x_{\mathrm{train}}$, let Q_x be the mean-field approximation to the Gibbs distribution found for a Boltzmann machine with the visible units clamped to x and further define $Z_{x,\mathrm{MF}}$ as:

$$Z_{x,\mathrm{MF}} := \sum_{h} Q_x(x,h) \log\left(\frac{e^{-E(x,h)}}{Q_x(x,h)}\right).$$

To use a quantum algorithm to prepare P from Q, an upper bound κ on the ratio of the approximation $P(v,h) \approx e^{-E(v,h)} / Z_{\mathrm{MF}}$ to $Q(v,h)$ is needed. Let $\kappa > 0$ be a constant that

satisfies for all visible and hidden configurations (v,h):

$$\frac{e^{-E(v,h)}}{Z_{\mathrm{MF}}} \leq \kappa Q(v,h),$$

wherein Z_{MF} is the approximation to the partition function given above. Then for all configurations of hidden and visible units,

$$P(v,h) \leq \frac{e^{-E(v,h)}}{Z_{\mathrm{MF}}} \leq \kappa Q(v,h).$$

The mean-field approximation can also be used to provide a lower bound for the log-partition function. For example, Jensen's inequality can be used to show that

$$\begin{aligned} \log(Z) &= \log\left(\sum_{v,h} \frac{Q(v,h)e^{-E(v,h)}}{Q(v,h)}\right), \\ &\geq \sum_{v,h} Q(v,h)\log\left(\frac{e^{-E(v,h)}}{Q(v,h)}\right) = \log(Z_{\mathrm{MF}}). \end{aligned}$$

Thus, $Z_{\mathrm{MF}} \leq Z$ and $P(v,h) \leq e^{-E(v,h)} / Z_{\mathrm{MF}}$.

[037] A coherent analog of the Gibbs state for a Boltzmann machine can be prepared with a probability of success of $\frac{Z}{\kappa Z_{\mathrm{MF}}}$. Similarly, the Gibbs state corresponding to the visible units being clamped to a configuration x can be prepared with success probability $\frac{Z_x}{\kappa_x Z_{x,\mathrm{MF}}}$. The mean-field parameters μ_i and ν_j can be determined as shown above and uniquely specify the mean-field distribution Q. The mean-field parameters are then used to approximate the partition functions Z and Z_x prepare a coherent analog of $Q(v,h)$, $|\psi_{\mathrm{MF}}\rangle$, by performing a series of single-qubit rotations:

$$|\psi_{\mathrm{MF}}\rangle := \prod_i R_y(2\arcsin(\sqrt{\mu_i}))|0\rangle \prod_j R_y(2\arcsin(\sqrt{\nu_j}))|0\rangle = \sum_{v,h} |v\rangle|h\rangle\sqrt{Q(v,h)}.$$

Rejection sampling can be used to refine this approximation to $\sum_{v,h} |v\rangle|h\rangle\sqrt{P(v,h)}$.

Define $\mathrm{P}(v,h) := \frac{e^{-E(v,h)}}{\kappa Z_{\mathrm{MF}} Q(v,h)}$. Note that this quantity can be computed efficiently from the mean-field parameters and so there is an associated efficient quantum circuit, and $0 \leq \mathrm{P}(v,h) \leq 1$.

[038] Since quantum operations are linear, if this is applied to a state $\sum_v \sum_h \sqrt{Q(v,h)}|v\rangle|h\rangle|0\rangle$, the state $\sum_v \sum_h \sqrt{Q(v,h)}|v\rangle|h\rangle|\mathrm{P}(v,h)\rangle$ is obtained. An additional quantum bit is added, and a controlled rotation of the form $R_y\left(2\sin^{-1}(\mathrm{P}(v,h))\right)$ is performed on this qubit to enact the following transformation:

$$\sum_{v,h} \sqrt{Q(v,h)}|v\rangle|h\rangle|\mathrm{P}(v,h)\rangle|0\rangle \mapsto \sum_{v,h} \sqrt{Q(v,h)}|v\rangle|h\rangle|\mathrm{P}(v,h)\rangle\left(\sqrt{1-\mathrm{P}(v,h)}|0\rangle + \sqrt{\mathrm{P}(v,h)}|1\rangle\right).$$

The register that contains the qubit string $\mathrm{P}(v,h)$ is then reverted to the $|0\rangle$ state by applying the same operations used to prepare $\mathrm{P}(v,h)$ in reverse. This process is possible because all quantum operations, save measurement, are reversible. Since $\mathrm{P}(v,h) \in [0,1]$, this is a properly normalized quantum state and its square is a properly normalized probability distribution. If the rightmost quantum bit is measured and a result of 1 is obtained (projective measurements always result in a unit vector) then the remainder of the state will be proportional to

$$\sum_{v,h} \sqrt{Q(v,h)\mathrm{P}(v,h)} = \sqrt{\frac{Z}{\kappa Z_{\mathrm{MF}}}} \sum_{v,h} \sqrt{\frac{e^{-E(v,h)}}{Z}}|v\rangle|h\rangle = \sqrt{\frac{Z}{\kappa Z_{\mathrm{MF}}}} \sum_{v,h} \sqrt{P(v,h)}\ |v\rangle|h\rangle,$$

which is the desired state up to a normalizing factor. The probability of measuring 1 is the square of this constant of proportionality:

$$P(1 \mid \kappa, Z_{\mathrm{MF}}) = \frac{Z}{\kappa Z_{\mathrm{MF}}}.$$

[039] Preparing a quantum state that can be used to estimate the expectation values over the data requires a slight modification to this algorithm. First, for each $x \in x_{\mathrm{train}}$ needed for the expectation values, $Q(v,h)$ is replaced with the constrained mean-field distribution $Q_x(x,h)$. Then using this data, the quantum state

$$\sum_h \sqrt{Q_x(x,h)}\ |x\rangle|h\rangle$$

can be prepared. The same procedure is can be followed using Q_x in place of Q, Z_x instead of Z, and $Z_{x,\mathrm{MF}}$ rather than Z_{MF}. The success probability of this algorithm is:

$$P(1 \mid \kappa, Z_{x,\mathrm{MF}}) = \frac{Z_x}{\kappa_x Z_{x,\mathrm{MF}}},$$

wherein κ_v is the value of κ that corresponds to the case where the visible units are clamped to x.

[040] This approach to state preparation problem uses a mean-field approximation rather than an infinite temperature Gibbs state as an initial state. This choice of initial state is important because the success probability of the state preparation process depends on the distance between the initial state and the target state. For machine learning applications, the inner product between the Gibbs state and the infinite temperature Gibbs state is often exponentially small; whereas the mean-field and Gibbs states typically have large overlaps.

[041] As shown below, if an insufficiently large value of κ is used, then the state preparation algorithm can still be used, but at the price of reduced fidelity with the ideal coherent Gibbs state. Using relaxed assumptions, such that $\kappa Q(v,h) \geq e^{-E(v,h)} / Z_{\mathrm{MF}}$ for all $(v,h) \in \text{good}$, $\kappa Q(v,h) < e^{-E(v,h)} / Z_{\mathrm{MF}}$ for all $j \in \text{bad}$, and $\sum_{(v,h)\in \text{bad}} \left(e^{-E(v,h)} - Z_{\mathrm{MF}} \kappa Q(v,h)\right) \leq \delta Z$, then a state can be prepared that has fidelity at least $1-\delta$ with the target Gibbs state with probability at least $Z(1-\delta)/(\kappa Z_{\mathrm{MF}})$.

[042] Prior to the measurement of the register that projects the state onto the success or failure branch, the state is:

$$\sum_{(v,h)\in \text{good}} \sqrt{Q(v,h)}\,|v\rangle|h\rangle\left(\sqrt{\frac{e^{-E(v,h)}}{Z_{\mathrm{MF}}\kappa Q(v,h)}}|1\rangle + \sqrt{1-\frac{e^{-E(v,h)}}{Z_{\mathrm{MF}}\kappa Q(v,h)}}|0\rangle\right) + \sum_{(v,h)\in \text{bad}} \sqrt{Q(v,h)}\,|v\rangle|h\rangle|1\rangle .$$

The probability of successfully preparing the approximation to the state is then:

$$\sum_{(v,h)\in \text{good}} \frac{e^{-E(v,h)}}{\kappa Z_{\mathrm{MF}}} + \sum_{(v,h)\in \text{bad}} Q(v,h) = \frac{Z - \left(\sum_{(v,h)\in \text{bad}} e^{-E(v,h)} - \sum_{(h,v)\in \text{bad}} \kappa Z_{\mathrm{MF}} Q(v,h)\right)}{Z_{\mathrm{MF}}\kappa} \geq \frac{Z(1-\delta)}{\kappa Z_{\mathrm{MF}}}.$$

The fidelity of the resultant state with the ideal state $\sum_{v,h} \sqrt{e^{-E(v,h)}/Z}\,|v\rangle|h\rangle$ is:

$$\frac{\sum_{(v,h)\in \text{good}} e^{-E(v,h)} + \sum_{(v,h)\in \text{bad}} \sqrt{Q(v,h) Z_{\mathrm{MF}} \kappa e^{-E(v,h)}}}{\sqrt{Z\left(\sum_{(v,h)\in \text{good}} e^{-E(v,h)} + \sum_{(v,h)\in \text{bad}} Q(v,h)\right)}} \geq \frac{\sum_{(v,h)\in \text{good}} e^{-E(v,h)} + \sum_{(v,h)\in \text{bad}} Q(v,h)}{\sqrt{Z\left(\sum_{(v,h)\in \text{good}} e^{-E(v,h)} + \sum_{(v,h)\in \text{bad}} Q(v,h)\right)}},$$

since $Q(v,h) Z_{\mathrm{MF}} \kappa \leq e^{-E(v,h)}$ and $(v,h) \in \text{bad}$. Using the assumption that $\sum_{(v,h)\in \text{bad}} \left(e^{-E(v,h)} - Z_{\mathrm{MF}} \kappa Q(v,h)\right) \leq \delta Z$, the fidelity is bounded above by:

$$\frac{\sqrt{\sum_{(v,h)\in \text{good}} e^{-E(v,h)} + \sum_{(v,h)\in \text{bad}} Q(v,h)}}{\sqrt{Z}} \geq \sqrt{1-\delta} \geq 1-\delta.$$

[043] Procedures for producing states that can be measured to estimate expectation values over the model and the data for training a deep Boltzmann machine are shown in Tables 1-2, respectively.

Input: Model weights w, visible biases b, hidden biases d, edge set E and κ.
Output: Quantum state that can be measured to obtain the correct Gibbs state for a deep Boltzmann machine.

function qGenModelState(w, b, d, E, κ)
Compute vectors of mean-field parameters μ and ν from w, b and d.
Compute the mean-field partition function Z_{MF}.
Prepare state $\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle := \left(\prod_{i=1}^{n_v} e^{-v_i\sqrt{\kappa_i}Y}|0\rangle\right)\left(\prod_{j=1}^{n_h} e^{-h_j\sqrt{\kappa_j}Y}|0\rangle\right)$
Add qubit register to store energy values and initialize to zero: $\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle \to \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|0\rangle$
for $i = 1 : n_v$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h) + v_i b_i + \ln\left(\mu_i^{v_i}(1-\mu_i)^{1-v_i}\right)\rangle$.
▷ Here an energy penalty is included to handle the bias and the visible units contribution to $Q(v,h)^{-1}$.
end for
for $j = 1 : n_h$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle\left|E(v,h) + h_j d_j + \ln\left(\nu_j^{h_j}(1-\nu_j)^{1-h_j}\right)\right\rangle$.
end for
for $(i,j) \in E$ **do**
$\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h) + v_i h_j w_{i,j}\rangle$.
end for
$\sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h)\rangle \to \sum_{v,h}\sqrt{Q(v,h)}|v\rangle|h\rangle|E(v,h)\rangle\left(\sqrt{\frac{e^{-E(v,h)}}{Z_{\text{MF}}\kappa}}|1\rangle + \sqrt{1-\frac{e^{-E(v,h)}}{Z_{\text{MF}}\kappa}}|0\rangle\right)$.
end function

Table 1. Quantum algorithm for generating states that can be measured to estimate expectation values over model for training a deep Boltzmann machine.

Input: Model weights w, visible biases b, hidden biases d, edge set E and κ, training vector x.
Output: Quantum state that can be measured to obtain the correct Gibbs state for a deep Boltzmann machine with the visible units clamped to x.

function qGenDataState(w, b, d, E, κ, x)
Compute vectors of mean-field parameters ν from w, b and d with visible units clamped to x.
Compute the mean-field partition function $Z_{x,\text{MF}}$.
for $i = 1 : n_v$ **do**
$\sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h)\rangle \to \sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h) + x_i b_i\rangle$.
end for
for $j = 1 : n_h$ **do**
$\sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h)\rangle \to \sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle\left|E(x,h) + h_j d_j + \ln\left(\nu_j^{h_j}(1-\nu_j)^{1-h_j}\right)\right\rangle$.
end for
for $(i,j) \in E$ **do**
$\sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h)\rangle \to \sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h) + x_i h_j w_{i,j}\rangle$.
end for
$\sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h)\rangle \to \sum_h\sqrt{Q(x,h)}|x\rangle|h\rangle|E(x,h)\rangle\left(\sqrt{\frac{e^{-E(x,h)}}{Z_{x,\text{MF}}\kappa}}|1\rangle + \sqrt{1-\frac{e^{-E(x,h)}}{Z_{x,\text{MF}}\kappa}}|0\rangle\right)$.
end function

Table 2. Quantum algorithm for generating states that can be measured to estimate expectation values over data for training a deep Boltzmann machine.

Gradient Calculation by Sampling

[044] One method for estimating the gradients of O_{ML} involves preparing the Gibbs state from the mean-field state and then drawing samples from the resultant distribution in order to estimate the expectation values required in Eqns. (1a)-(1c) above. This approach can be improved using the quantum method known as amplitude amplification, a generalization of Grover's search algorithm that quadratically reduces the mean number of repetitions needed to draw a sample from the Gibbs distribution using the methods discussed above.

[045] There exists a quantum algorithm that can estimate the gradient of O_{ML} using N_{train} samples for a Boltzmann machine on a connected graph with E edges. The mean number of quantum operations required by algorithm to compute the gradient is

$$\tilde{O}\left(N_{\mathrm{train}} E \sqrt{\kappa + \sum_{v \in x_{\mathrm{train}}} \kappa_v}\right),$$

wherein κ_v is the value of κ that corresponds to the Gibbs distribution when the visible units are clamped to v and $f \in \tilde{O}(g)$ implies $f \in O(g)$ up to polylogarithmic factors.

[046] Table 3 illustrates a representative method of computing gradients. The procedure **qGenModelState** and **qGenDataState** (shown in Tables 1 and 2, respectively) represent the only quantum processing in this method. It can be shown that the expected cost of the method of Table 3 is $\tilde{O}(N_{\mathrm{train}} E \sqrt{\kappa + \max_v \kappa_v})$.

Input: Initial model weights w, visible biases b, hidden biases d, edge set E and κ, a set of training vectors x_{train}, a regularization term λ and a learning rate r.
Output: Three arrays containing gradients of weights, hidden biases and visible biases: `gradMLw`, `gradMLb`, `gradMLd`.

```
for i = 1 : N_train do
    success ← 0
    while success = 0 do
        |ψ⟩ ← qGenModelState(w, b, d, E, κ)
        success ← result of measuring last qubit in |ψ⟩
    end while
    modelVisibleUnits[i] ← result of measuring visible qubit register in |ψ⟩.
    modelHiddenUnits[i] ← result of measuring hidden unit register in |ψ⟩ using amplitude amplification.
    success ← 0
    while success = 0 do
        |ψ⟩ ← qGenDataState(w, b, d, E, κ, x_train[i]).
        success ← result of measuring last qubit in |ψ⟩ using amplitude amplification.
    end while
    dataVisibleUnits[i] ← result of measuring visible qubit register in |ψ⟩.
    dataHiddenUnits[i] ← result of measuring hidden unit register in |ψ⟩.
end for
for each visible unit i and hidden unit j do
    gradMLw[i, j] ← r (⟨v_i h_j⟩_data − ⟨v_i h_j⟩_model − λ w_{i,j}).
    gradMLb[i] ← r (⟨v_i⟩_data − ⟨v_i⟩_model).
    gradMLd[j] ← r (⟨h_j⟩_data − ⟨h_j⟩_model).
end for
```

Table 3. Quantum method for computing gradients for training a deep Boltzmann machine.

In contrast, the number of operations and queries to U_O required to estimate the gradients using greedy layer by layer optimization scales as $\tilde{O}(N_{\text{train}}\ell E)$, wherein ℓ is the number of layers in the deep Boltzmann machine. Assuming that κ is a constant, it follows that a quantum sampling approach provides an asymptotic advantage for training deep networks. In practice, the two approaches are difficult to directly compare because they both optimize different objective functions and thus the qualities of the resultant trained models will differ. It is reasonable to expect, however, that the quantum approach will tend to find superior models because it optimizes the maximum likelihood objective function up to sampling error due to taking finite N_{train}.

[047] Note that the method of Table 3 has an important advantage over typical quantum machine learning algorithms in that it does not require that the training vectors be stored in quantum memory. Instead, only $n_v + n_h + 1 + 2\log\left(\frac{1}{E}\right)$ qubits are needed for a numerical precision of E in the evaluation of the energy and $Q(v,h)$. This means that an algorithm that could not be done classically could be performed with fewer than 100 qubits, assuming that 32-bits of precision suffices for the energy and $Q(v,h)$. Recent developments in quantum rotation synthesis could be used to remove the requirement that the energy is explicitly stored as a qubit string which might substantially reduce space requirements. An alternative method is disclosed below in which the quantum computer can coherently access this database via an oracle.

Training via quantum amplitude estimation

[048] An alternative method is based on access to the training data via a quantum oracle which could represent either an efficient quantum algorithm that provides the training data (such as another Boltzmann machine used as a generative model) or a quantum database that stores the memory via a binary access tree. If the training set is $\{x_i \,|\, i = 1,\ldots,N_{\text{train}}\}$, then the oracle is a unitary operation U_O that, for any computational basis state $|i\rangle$ and any bit strings y and x_i of length n_v the operation:

$$U_O|i\rangle|y\rangle := |i\rangle|y \oplus x_i\rangle,$$

A single quantum access to U_O is sufficient to prepare a uniform distribution over all the training data:

$$U_O\left(\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle|0\rangle\right)=\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle|x_i\rangle$$

The state $\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}|i\rangle\,|0\rangle$ can be efficiently prepared using quantum techniques so the entire procedure is efficient.

[049] At first glance, the ability to prepare a superposition over all data from the training set seems to be a powerful resource. However, a similar probability distribution can be generated classically using one query by picking a random training vector. More sophisticated approaches are needed if to leverage computational advantages using such quantum superpositions. The method shown in Table 4 uses such superpositions to provide such advantages for computing gradients of objective functions, under certain circumstances. It can be demonstrated that there exists a quantum algorithm that can compute $r\frac{\partial O_{\text{ML}}}{\partial w_{ij}}$, $r\frac{\partial O_{\text{ML}}}{\partial b_i}$, or $r\frac{\partial O_{\text{ML}}}{\partial d_j}$ for a Boltzmann machine on a connected graph with E edges to within error δ using an expected number of queries to U_O that scales as $\tilde{O}\left(\frac{\kappa+\max_v \kappa_v}{\delta}\right)$ and a number of quantum operations that scales as $\tilde{O}\left(\frac{E(\kappa+\max_v \kappa_v)}{\delta}\right)$ for a constant learning rate r.

[050] The method of computing gradients for a deep Boltzmann machine shown in Table 4 uses amplitude estimation. This method provides a quadratic reduction in the number of samples needed to learn the probability of an event occurring. For any positive integer L, the amplitude estimation algorithm of takes as input a quantum algorithm that does not use measurement and with success probability a and outputs $\tilde{a}$ $(0 \le \tilde{a} \le 1)$ such that $|\tilde{a}-a| \le \frac{\pi(\pi+1)}{L}$ with probability at least $8/\pi^2$, using L iterations of Grover's algorithm. If $a=0$, then $\tilde{a}=0$ with certainty, and if $a=1$ and L is even, then $\tilde{a}=1$ with certainty. Amplitude estimation is described in further detail in Brassard et al. "Quantum amplitude amplification and estimation," available at arxiv.org/quanth-ph/0005055 v1 (2000), which is incorporated herein by reference.

[051] The procedure of Table 4 provides a method for computing the derivative of O_{ML} with respect to the weights. This procedure can be adapted to compute the derivatives with respect to the biases. The first step in this procedure is preparation of a uniform superposition of all training data and then applying U_O to the superposition to obtain:

$$\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle.$$

Any quantum algorithm that does not use measurement is linear and hence applying **qGenDataState** (shown in Table 2 above) to this superposition yields:

$$\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle\sum_h\sqrt{Q(x_p,h)}|h\rangle|\mathrm{P}(x_p,h)\rangle\left(\sqrt{1-\mathrm{P}(x_p,h)}|0\rangle+\sqrt{\mathrm{P}(x_p,h)}|1\rangle\right)$$

$$:=\frac{1}{\sqrt{N_{\text{train}}}}\sum_{p=1}^{N_{\text{train}}}|p\rangle|x_p\rangle\sum_h\sqrt{Q(x_p,h)}|h\rangle|\mathrm{P}(x_p,h)\rangle|\chi(x_p,h)\rangle.$$

If a measurement $\chi=1$ is success then $\tilde{O}((\kappa+\max_v\kappa_v)/\Delta)$ preparations are needed to learn $P(\texttt{success})=P(\chi=1)$ to within relative error $\Delta/8$ with high probability. This is because $P(\texttt{success})\geq 1/(\kappa+\max_v\kappa_v)$. Similarly, success can be associated with an event in which an i^{th} visible unit is 1 and the j^{th} hidden unit is 1 and a successful state preparation is measured. This marking process is exactly the same as the previous case, but requires a Toffoli gate (a doubly-controlled NOT gate, which can be implemented using fundamental gates) and two Hadamard operations. Thus $P(v_i=h_j=\chi=1)$ can be learned within relative error $\Delta/8$ using $\tilde{O}((\kappa+\max_v\kappa_v)/\Delta)$ preparations. It then follows from the laws of conditional probability that

$$\langle v_i h_j\rangle_{\text{data}}=P([x_p]_i=h_j=1\,|\,\chi=1)=\frac{P([x_p]_i=h_j=\chi=1)}{P(\chi=1)} \qquad (2)$$

can be calculated.

[052] In order to ensure that the total error in $\langle v_i h_j\rangle_{\text{data}}$ is at most Δ, the error in the quotient in (2) must be bounded. It can be seen that for $\Delta<1/2$,

$$\left|\frac{P([x_i]_j=h_k=\chi=1)(1\pm\Delta/8)}{P(\chi=1)(1\pm\Delta/8)}-\frac{P([x_i]_j=h_k=\chi=1)}{P(\chi=1)}\right|\leq\frac{\Delta P([x_i]_j=h_k=\chi=1)}{P(\chi=1)}\leq\Delta.$$

Therefore the algorithm gives $\langle v_i h_j\rangle_{\text{data}}$ within error Δ.

[053] The same steps can be repeated using **qGenModelState** (Table 1) as the state preparation subroutine used in amplitude estimation. This allows computation of $\langle v_i h_j\rangle_{\text{data}}$ within error Δ using $\tilde{O}(1/\Delta)$ state preparations. The triangle inequality shows that the maximum error incurred from approximating $\langle v_i h_j\rangle_{\text{data}}-\langle v_i h_j\rangle_{\text{model}}$ is at most 2Δ.

Therefore, with a learning rate of r, the overall error in the derivative is at most $2\Delta r$. If $\Delta = \delta/(2r)$ then the overall algorithm requires $\tilde{O}(1/\delta)$ state preparations for constant r.

[054] Each state preparation requires one query to U_O and $\tilde{O}(E)$ operations assuming that the graph that underlies the Boltzmann machine is connected. This means that the expected query complexity of the algorithm is $\tilde{O}((\kappa + \max_v \kappa_v)/\delta)$ and the number of circuit elements required is $\tilde{O}((\kappa + \max_v \kappa_v)E/\delta)$.

Algorithm 4 Quantum algorithm for computing gradient of weights using amplitude estimation for use in training a deep Boltzmann machine.

Input: Initial model weights w, visible biases b, hidden biases d, edge set E and κ, a set of training vectors x_{train}, a regularization term λ, $1/2 \geq \Delta > 0$, a learning rate r, and a specification of edge (i, j).
Output: $r\frac{\partial O_{\text{NLL}}}{\partial w_{ij}}$ calculated to within error $2r\Delta$.

Call U_O once to prepare state $|\psi\rangle \leftarrow \frac{1}{\sqrt{N_{\text{train}}}} \sum_{p \in x_{\text{train}}} |p\rangle |x_p\rangle$.
$|\psi\rangle \leftarrow$ `qGenDataState`$(w, b, d, E, \kappa, |\psi\rangle)$. ▷ Apply Algorithm 2 using a superposition over x_p rather than a single value.
Use amplitude estimation on state preparation process for $|\psi\rangle$ to learn $P(|x_p\rangle_j = h_j = \texttt{success} = 1)$ within error $\Delta/8$.
Use amplitude estimation on state preparation process for $|\psi\rangle$ to learn $P(\texttt{success} = 1)$ within error $\Delta/8$.
$\langle v_i h_j \rangle_{\text{data}} \leftarrow \frac{P(|x_p\rangle_i = h_j = \texttt{success} = 1)}{P(\texttt{success} = 1)}$.
Use amplitude estimation in exact same fashion on `qGenModelState`(w, b, d, E, κ) to learn $\langle v_i h_j \rangle_{\text{data}}$.
$\frac{\partial O_{\text{NLL}}}{\partial w_{ij}} \leftarrow r(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}})$

Table 4. Quantum procedure for computing gradient of weights using amplitude estimation for training a deep Boltzmann machine.

[055] There are two qualitative differences between the method of Table 4 and that of Table 3. First, the method of Table 4 provides detailed information about one direction of the gradient, whereas samples produced by the method of Table 3 provide limited information about every direction. The method of Table 4 can be repeated for each of the components of the gradient vector in order to perform an update of the weights and biases of the Boltzmann machine. Second, amplitude amplification is not used to reduce the effective value of κ. Amplitude amplification only gives a quadratic advantage if used in an algorithm that uses measurement and feedback unless the probability of success is known.

[056] The quadratic scaling with E means that the method of Table 4 may not be preferable to that of Table 3 for learning all weights. On the other hand, the method of Table 4 can be used to improve previously estimated gradients. In one example, a preliminary gradient estimation step is performed using a direct gradient estimation method using $O(\sqrt{N_{\text{train}}})$ randomly selected training vectors. Then the gradient is estimated by breaking the results into smaller groups and computing the mean and the variance of each component of the gradient vector over each of the subgroups. The

components of the gradients with the largest uncertainty can then be learned using the above method with $\delta \sim 1/\sqrt{N_{\text{train}}}$. This approach allows the benefits of different approaches to be used, especially in cases where the majority of the uncertainty in the gradient comes from a small number of components.

[057] The discussion above is directed to learning on restricted Boltzmann machines and deep Boltzmann machines. The disclosed quantum methods can train full Boltzmann machines given that the mean-field approximation to the Gibbs state has only polynomially small overlap with the true Gibbs state. The intra-layer connections associated with such Boltzmann machines can permit superior models.

Example Implementations

[058] With reference to FIG. 2, a method 200 of training a Boltzmann machine includes providing training data and initializing at 202. At 204, a gradient of an objective function is determined with a quantum computer, and at 206, gradient ascent is used to refine the Boltzmann machine. If a local optimum is reached as determined at 208, an optimized Boltzmann machine specification is returned at 210. Otherwise, additional quantum processing is executed at 204.

[059] A method 300 of gradient calculation is illustrated in FIG. 3. At 302, training data and an initial Boltzmann machine specification (such as numbers of layers, and numbers of units in each layer) are provided. At 304, a quantum-based sampling algorithm (such as that of Table 1 or Table 2 for example) is carried out for the selected model and data expectation using a training vector. At 306, visible unit and hidden unit values (v_i, h_j) are determined for each edge specified by (i,j) for a model average and a data average. At 308, the result vectors are subtracted, and the result returned.

[060] Referring to FIG. 4, a method 400 of establishing and sampling a model average includes receiving a Boltzmann machine specification at 402 and computing a mean field approximation to a Gibbs state at 404. At 406, the mean-field values are used to prepare a mean-field state on qubits in a quantum computer and at 408, a qubit string is prepared that stores energy of each configuration (v,h). This qubit string can be represented as $\sum_{v,h} a(v,h)|v,h\rangle|E(v,h)\rangle$. At 410, a qubit is added, and quantum superposition is used rotate the qubit to $\frac{\sqrt{\mathrm{P}(v,h)}}{a(v,h)}|1\rangle + \sqrt{1-\frac{P(v,h)}{a^2(v,h)}}|0\rangle$. At 412, amplitude estimation is used to measure $|1\rangle$ and at 414, (v,h) is measured. The measured value (v,h) is returned at 416.

[061] A data average can be determined in a similar fashion. Referring to FIG. 5, a method 500 of sampling a data average includes receiving a Boltzmann machine specification at 502 and computing a mean field approximation to a Gibbs state with $v = x_i$ at 504. At 506, a mean-field state is prepared on qubits in a quantum computer, and at 508, a qubit string is prepared that stores energy of each configuration (v,h). This qubit string can be represented as $\sum_{v,h} a(x_i,h)\left|x_i,h\right\rangle\left|E(x_i,h)\right\rangle$. At 510, a qubit is added, and quantum superposition is used rotate the qubit to $\frac{\sqrt{\mathrm{P}(x_i,h)}}{a(x_i,h)}\left|1\right\rangle+\sqrt{1-\frac{P(x_i,h)}{a^2(x_i,h)}}\left|0\right\rangle$. At 512, amplitude estimation is used to measure $\left|1\right\rangle$ and at 514, (v,h) is measured. The measured value (v,h) is returned at 516.

[062] A method 600 of gradient calculation using amplitude estimation is shown in FIG. 6. At 602, training vectors $\{x_i\}$ and an initial Boltzmann machine specification are provided. At 604, the model average and data average needed to compute the derivative of the objective function with respect to the edge weight (i,j) using amplitude estimation in a quantum computer are determined. The process for computing the derivatives with respect to biases is identical. At 606, the result vectors are subtracted, and the result returned.

[063] A model average can be determined using a method 700 shown in FIG. 7. A Boltzmann machine specification and an edge specification (i,j) are obtained at 702, and a mean field approximation to a Gibbs state is determined at 704. At 706, a mean-field state is prepared on qubits in a quantum computer, and at 708, a qubit string is prepared that stores energy of each configurations (v,h). This qubit string can be represented as $\sum_{v,h} a(v,h)\left|v,h\right\rangle\left|E(v,h)\right\rangle$. At 710, a qubit is added, and quantum superposition is used rotate this qubit to $\frac{\sqrt{\mathrm{P}(v,h)}}{a(v,h)}\left|1\right\rangle+\sqrt{1-\frac{P(v,h)}{a^2(v,h)}}\left|0\right\rangle$. At 712, amplitude estimation is used to determine a probability of measuring this qubit as $\left|1\right\rangle$ and at 714, amplitude estimation is used to determine the probability of this qubit being $\left|1\right\rangle$ and $h_i = v_j = 1$. The ratio of the two probabilities is returned at 716.

[064] A model average can also be determined using a method 800 shown in FIG. 8. A Boltzmann machine specification, an edge specification (i,j),and a superposition of

training vectors $\sum_i |i\rangle|x_i\rangle$ are obtained at 802, and a mean field approximation to a Gibbs state is determined at 804. At 806, a mean-field state is prepared on qubits in a quantum computer simultaneously for each $|x_i\rangle$ in the superposition. At 808, a qubit string is prepared that stores the energy of each configuration (v,h). This qubit string can be represented as $\sum_{v,h} a(v,h)|v,h\rangle|E(v,h)\rangle$. At 810, a qubit is added, and quantum superposition is used rotate this qubit to $\frac{\sqrt{P(v,h)}}{a(v,h)}|1\rangle + \sqrt{1-\frac{P(v,h)}{a^2(v,h)}}|0\rangle$. At 812, amplitude estimation is used to determine a probability of measuring this qubit as $|1\rangle$ and at 814, amplitude estimation is used to determine the probability of this qubit being $|1\rangle$ and $h_i = v_j = 1$. The ratio of the two probabilities is returned at 816.

Computing Environments

[065] FIG. 9 and the following discussion are intended to provide a brief, general description of an exemplary computing environment in which the disclosed technology may be implemented. Although not required, the disclosed technology is described in the general context of computer executable instructions, such as program modules, being executed by a personal computer (PC). Generally, program modules include routines, programs, objects, components, data structures, etc., that perform particular tasks or implement particular abstract data types. Moreover, the disclosed technology may be implemented with other computer system configurations, including hand held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The disclosed technology may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices. Typically, a classical computing environment is coupled to a quantum computing environment, but a quantum computing environment is not shown in FIG. 9.

[066] With reference to FIG. 9, an exemplary system for implementing the disclosed technology includes a general purpose computing device in the form of an exemplary conventional PC 900, including one or more processing units 902, a system memory 904, and a system bus 906 that couples various system components including the system

memory 904 to the one or more processing units 902. The system bus 906 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The exemplary system memory 904 includes read only memory (ROM) 908 and random access memory (RAM) 910. A basic input/output system (BIOS) 912, containing the basic routines that help with the transfer of information between elements within the PC 900, is stored in ROM 908.

[067] As shown in FIG. 9, a specification of a Boltzmann machine is stored in a memory portion 916. In addition, a memory portion 918 stores circuit definitions that are used to configure a quantum computer to, for example, establish states that approximate the Gibbs state. Computer-executable instructions are also stored for receiving precisions as well as communicating circuit definitions and states to be used. Instructions for gradient determination and evaluation are stored at 911. In some examples, the PC 900 is provided with Boltzmann machine weights and biases so as to define a trained Boltzmann machine that receives input data examples, or produces output data examples. In alternative examples, a Boltzmann machine trained as disclosed herein can be coupled to another classifier such as another Boltzmann machine or other classifier.

[068] The exemplary PC 900 further includes one or more storage devices 930 such as a hard disk drive for reading from and writing to a hard disk, a magnetic disk drive for reading from or writing to a removable magnetic disk, and an optical disk drive for reading from or writing to a removable optical disk (such as a CD-ROM or other optical media). Such storage devices can be connected to the system bus 906 by a hard disk drive interface, a magnetic disk drive interface, and an optical drive interface, respectively. The drives and their associated computer readable media provide nonvolatile storage of computer-readable instructions, data structures, program modules, and other data for the PC 900. Other types of computer-readable media which can store data that is accessible by a PC, such as magnetic cassettes, flash memory cards, digital video disks, CDs, DVDs, RAMs, ROMs, and the like, may also be used in the exemplary operating environment.

[069] A number of program modules may be stored in the storage devices 930 including an operating system, one or more application programs, other program modules, and program data. Storage of Boltzmann machine specifications, and computer-executable instructions for training procedures, determining objective functions, and configuring a quantum computer can be stored in the storage devices 930 as well as or in addition to the memory 904. A user may enter commands and information into the PC 900 through one or more input devices 940 such as a keyboard and a pointing device such as a mouse.

Other input devices may include a digital camera, microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the one or more processing units 902 through a serial port interface that is coupled to the system bus 906, but may be connected by other interfaces such as a parallel port, game port, or universal serial bus (USB). A monitor 946 or other type of display device is also connected to the system bus 906 via an interface, such as a video adapter. Other peripheral output devices 945, such as speakers and printers (not shown), may be included. In some cases, a user interface is display so that a user can input a Boltzmann machine specification for training, and verify successful training.

[070] The PC 900 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 960. In some examples, one or more network or communication connections 950 are included. The remote computer 960 may be another PC, a server, a router, a network PC, or a peer device or other common network node, and typically includes many or all of the elements described above relative to the PC 900, although only a memory storage device 962 has been illustrated in FIG. 9. The storage device 962 can provide storage of Boltzmann machine specifications and associated training instructions. The personal computer 900 and/or the remote computer 960 can be connected to a logical a local area network (LAN) and a wide area network (WAN). Such networking environments are commonplace in offices, enterprise wide computer networks, intranets, and the Internet.

[071] When used in a LAN networking environment, the PC 900 is connected to the LAN through a network interface. When used in a WAN networking environment, the PC 900 typically includes a modem or other means for establishing communications over the WAN, such as the Internet. In a networked environment, program modules depicted relative to the personal computer 900, or portions thereof, may be stored in the remote memory storage device or other locations on the LAN or WAN. The network connections shown are exemplary, and other means of establishing a communications link between the computers may be used.

[072] With reference to FIG. 10, an exemplary system for implementing the disclosed technology includes computing environment 1000 that includes a quantum processing unit 1002 and one or more monitoring/measuring device(s) 1046. The quantum processor executes quantum circuits that are precompiled by classical compiler unit 1020 utilizing one or more classical processor(s) 1010. Quantum circuits are downloaded into the quantum processing unit via a quantum bus 1006 based on Boltzmann machine

specifications and training instructions, such as quantum state preparation procedures described above.

[073] With reference to FIG. 10, the compilation is the process of translation of a high-level description of a quantum algorithm into a sequence of quantum circuits. Such high-level description may be stored, as the case may be, on one or more external computer(s) 1060 outside the computing environment 1000 utilizing one or more memory and/or storage device(s) 1062, then downloaded as necessary into the computing environment 1000 via one or more communication connection(s) 1050. Alternatively, the classical compiler unit 1020 is coupled to a classical processor 1010 and a procedure library 1021 that contains some or all procedures or data necessary to implement the methods described above such as a Boltzmann machine specification, state preparation procedures (e.g., **qGenModelState**, **qGenDataState**), and mean-field evaluations.

[074] Having described and illustrated the principles of the disclosed technology with reference to the illustrated embodiments, it will be recognized that the illustrated embodiments can be modified in arrangement and detail without departing from such principles. The technologies from any example can be combined with the technologies described in any one or more of the other examples. Alternatives specifically addressed in these sections are merely exemplary and do not constitute all possible examples.

CLAIMS

1. A method of training a Boltzmann machine, comprising:

with a classical computer, receiving a specification of a Boltzmann machine, an objective function, and associated training data;

in a quantum computer, determining at least one gradient of the objective function; and

based on the at least one gradient of the objective function, specifying at least one visible bias, at least one hidden bias or at least one weight of the Boltzmann machine so as to produce a trained Boltzmann machine.

2. The method of claim 1, further comprising:

in the quantum computer, preparing a plurality of qubits so as to represent a Gibbs distribution; and

producing the at least one gradient of the objective function by sampling the states of each of the plurality of qubits.

3. The method of any of claims 1 and 2, wherein the objective function is a sum of an average log-likelihood of the training data and a regularization function.

4. The method of any of claims 1-3, further comprising, in the quantum computer, producing a quantum state associated with model values, and in the classical computer, establishing the model values based on sampling of the quantum state.

5. The method of any of claims 1-4, wherein the model is revised using gradient ascent.

6. The method of any of claims 1-5, further comprising, in the quantum computer, producing a quantum state associated with data values, and in the classical computer, revising the model values based on sampling of the quantum state.

7. The method of claims 6, wherein the quantum state associated with the data values is produced based on a mean-field approximation to a Gibbs distribution.

8. A method, comprising:

preparing at least one quantum state in a quantum computer to approximate a Gibbs state and sampling the at least one quantum state; and

estimating in a classical computer, gradients of an objective function based on the sampling of the at least one quantum state.

9. The method of claim 8, further comprising defining weights or biases for the Boltzmann machine based on the estimated values and updating the weights or biases

based on gradients of at least one of Boltzmann machine weights, hidden biases, and visible biases.

10. The method of any of claims 8-9, further comprising processing a data example based on the defined weights and biases of the Boltzmann machine, wherein the data example is associated with an image, a shape, speech, a text, an audio recording, a video recording, or a quantum state.

11. The method of any of claims 8-10, wherein the objective function is associated with a sum of an average log-likelihood of the training data and a regularization function and the Boltzmann machine is a deep restricted Boltzmann machine having two or more layers.

12. The method of any of claims 8-11, further comprising;

in the quantum computer,

preparing a qubit string storing energy values associated with a plurality of hidden weights and biases;

adding a qubit to the prepared qubit string and applying a rotation operator;

applying amplitude estimation to the prepared qubit string with the added qubit; and

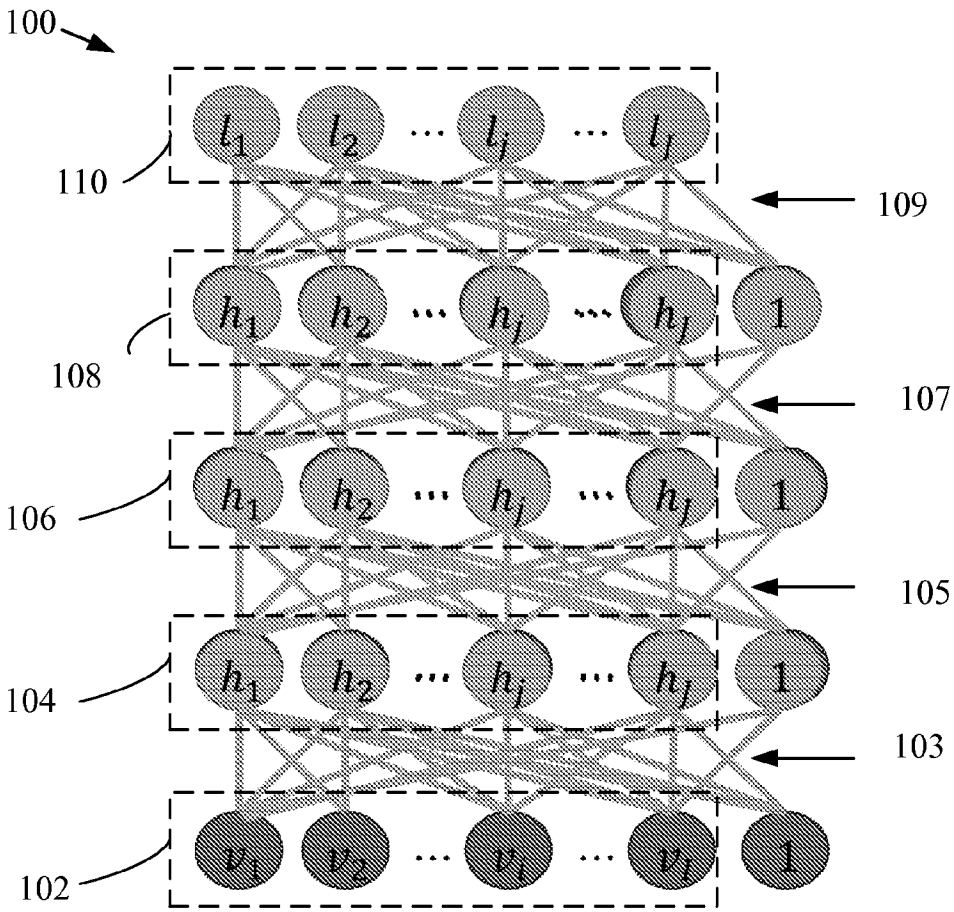
in the classical computer, determining a gradient of an objective function based on the amplitude estimation.

13. The method of any of claims 8-12, wherein the preparing at least one state in the quantum computer comprises preparing a state associated with model weights, and hidden and variable biases, and further comprising computing a mean-field partition function, wherein the at least one state is based in part on the mean-field partition function.

14. The method of any of claims 8-13, wherein the preparing at least one state in the quantum computer comprises preparing a state associated with model weights, and hidden and variable biases, with visible unit fixed so as to correspond to a selected training vector.

15. The method of any of claims 8-14, further comprising computing a mean-field partition function, wherein the at least one state is based in part on the mean-field partition function associated with the selected training vector.

FIG. 1


100
110
109
108
107
106
105
104
103
102
l_1 l_2 ... l_j ... l_J
h_1 h_2 ... h_j ... h_J 1
h_1 h_2 ... h_j ... h_J 1
h_1 h_2 ... h_j ... h_J 1
v_1 v_2 ... v_i ... v_I 1

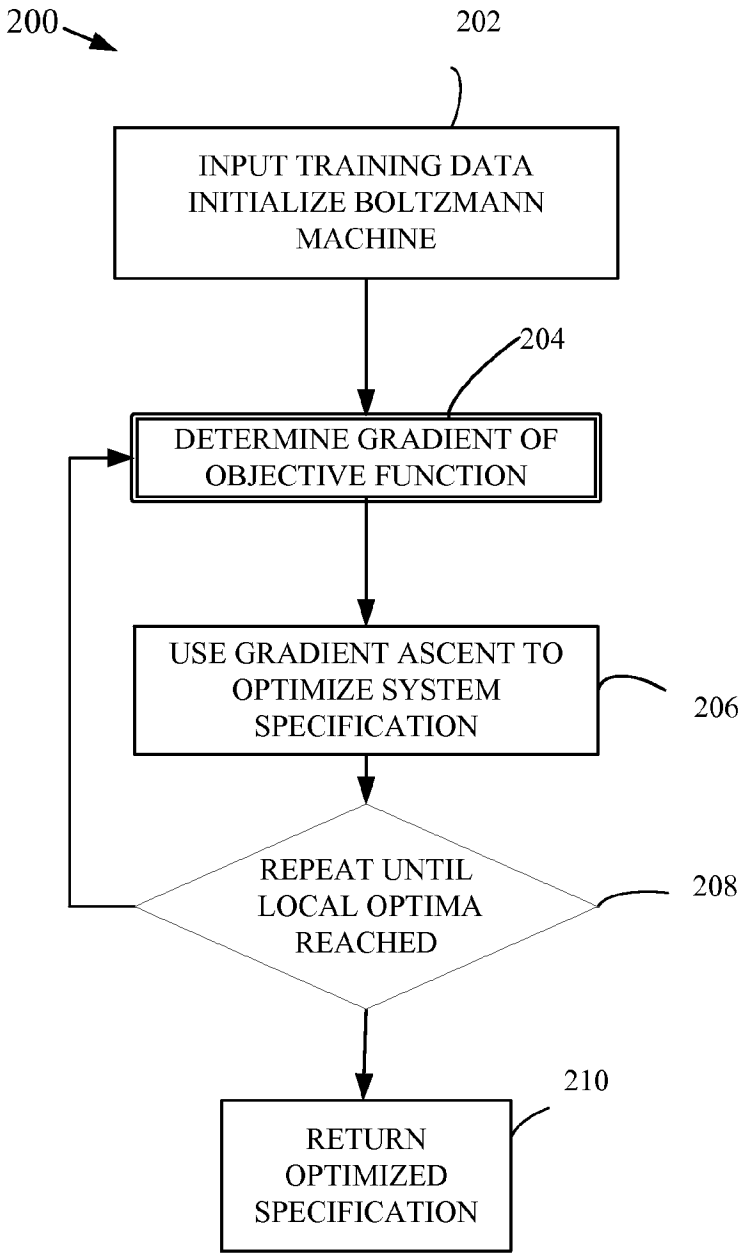

200
202
INPUT TRAINING DATA
INITIALIZE BOLTZMANN MACHINE
204
DETERMINE GRADIENT OF OBJECTIVE FUNCTION
USE GRADIENT ASCENT TO OPTIMIZE SYSTEM SPECIFICATION
206
REPEAT UNTIL LOCAL OPTIMA REACHED
208
210
RETURN OPTIMIZED SPECIFICATION

FIG. 2

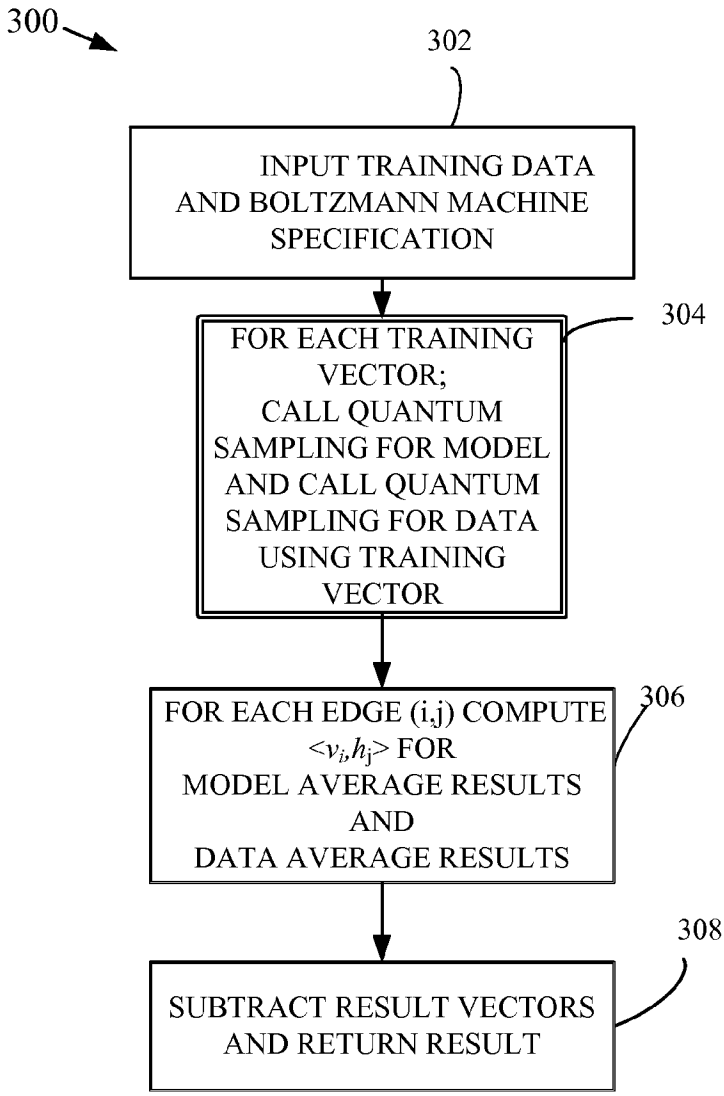

300
302
INPUT TRAINING DATA AND BOLTZMANN MACHINE SPECIFICATION
304
FOR EACH TRAINING VECTOR; CALL QUANTUM SAMPLING FOR MODEL AND CALL QUANTUM SAMPLING FOR DATA USING TRAINING VECTOR
306
FOR EACH EDGE (i,j) COMPUTE <v_i,h_j> FOR MODEL AVERAGE RESULTS AND DATA AVERAGE RESULTS
308
SUBTRACT RESULT VECTORS AND RETURN RESULT

FIG. 3

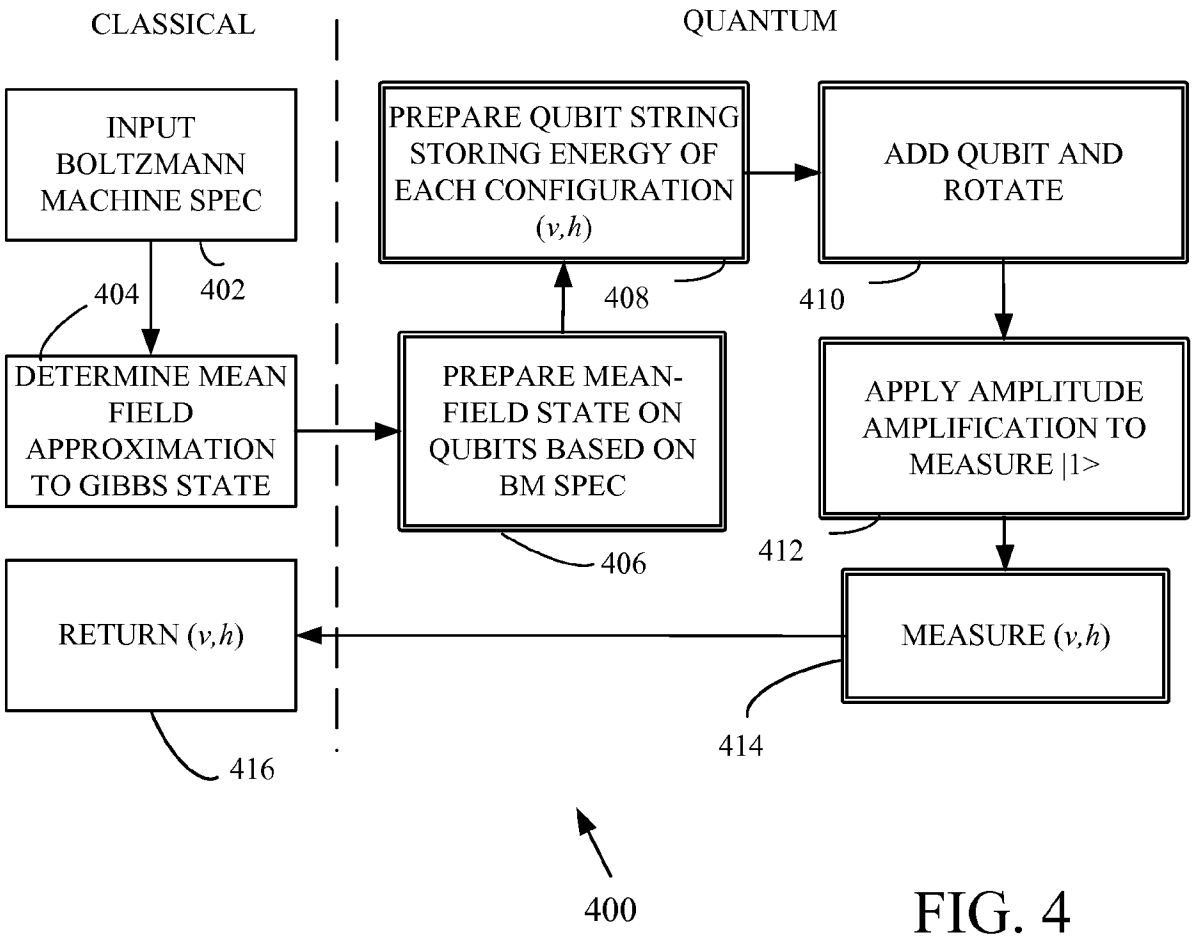

CLASSICAL
QUANTUM
INPUT BOLTZMANN MACHINE SPEC
402
404
DETERMINE MEAN FIELD APPROXIMATION TO GIBBS STATE
PREPARE MEAN-FIELD STATE ON QUBITS BASED ON BM SPEC
406
PREPARE QUBIT STRING STORING ENERGY OF EACH CONFIGURATION (v,h)
408
ADD QUBIT AND ROTATE
410
APPLY AMPLITUDE AMPLIFICATION TO MEASURE |1>
412
MEASURE (v,h)
414
RETURN (v,h)
416
400

FIG. 4

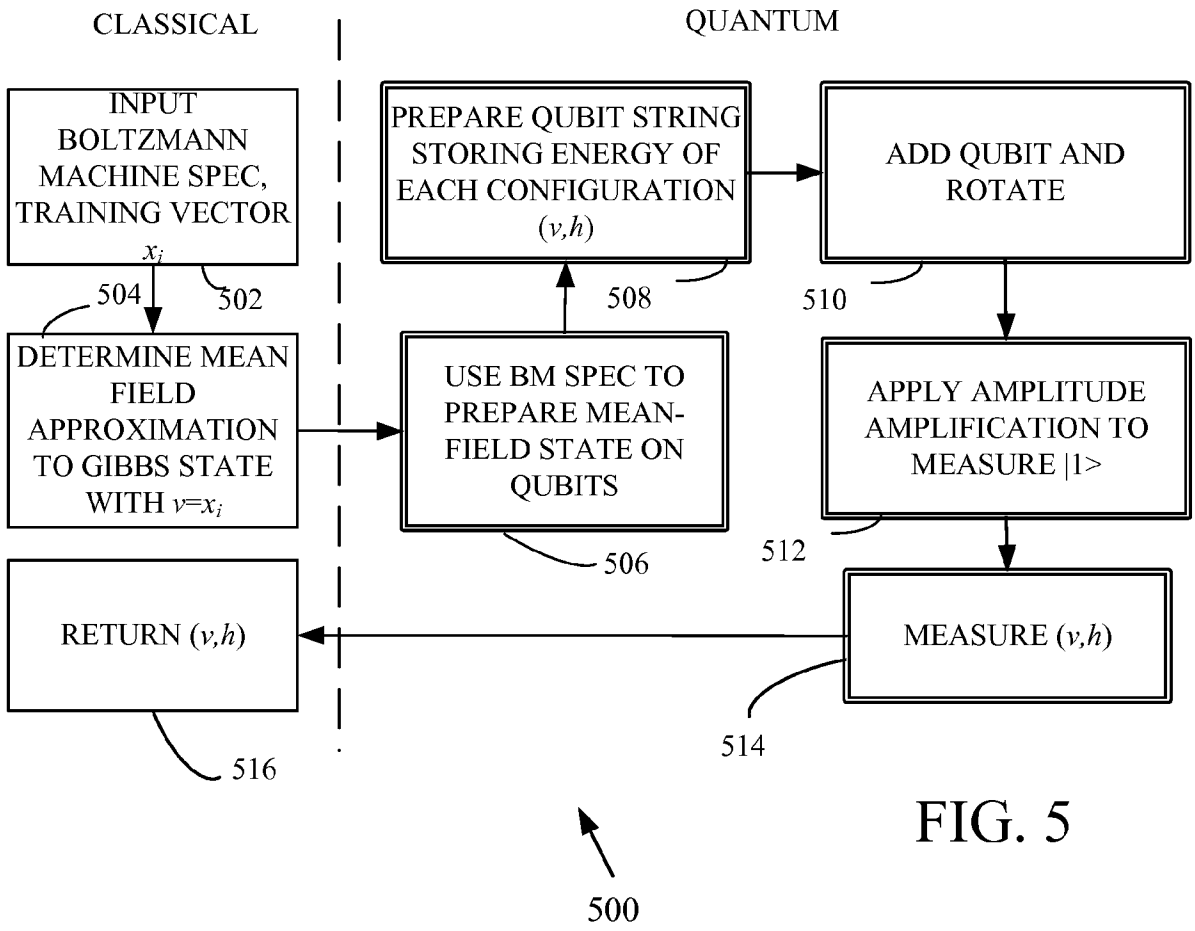

CLASSICAL
QUANTUM
INPUT BOLTZMANN MACHINE SPEC, TRAINING VECTOR x_i
PREPARE QUBIT STRING STORING ENERGY OF EACH CONFIGURATION (v,h)
ADD QUBIT AND ROTATE
504
502
508
510
DETERMINE MEAN FIELD APPROXIMATION TO GIBBS STATE WITH $v=x_i$
USE BM SPEC TO PREPARE MEAN-FIELD STATE ON QUBITS
APPLY AMPLITUDE AMPLIFICATION TO MEASURE |1>
506
512
RETURN (v,h)
MEASURE (v,h)
514
516
500

FIG. 5

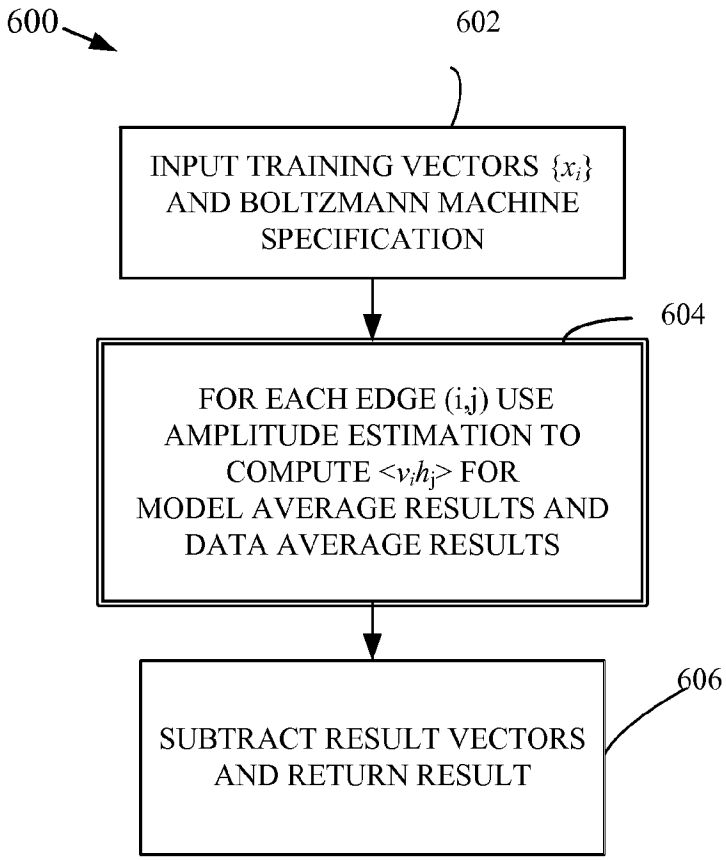

600
602
INPUT TRAINING VECTORS $\{x_i\}$ AND BOLTZMANN MACHINE SPECIFICATION
604
FOR EACH EDGE (i,j) USE AMPLITUDE ESTIMATION TO COMPUTE $<v_i h_j>$ FOR MODEL AVERAGE RESULTS AND DATA AVERAGE RESULTS
606
SUBTRACT RESULT VECTORS AND RETURN RESULT

FIG. 6

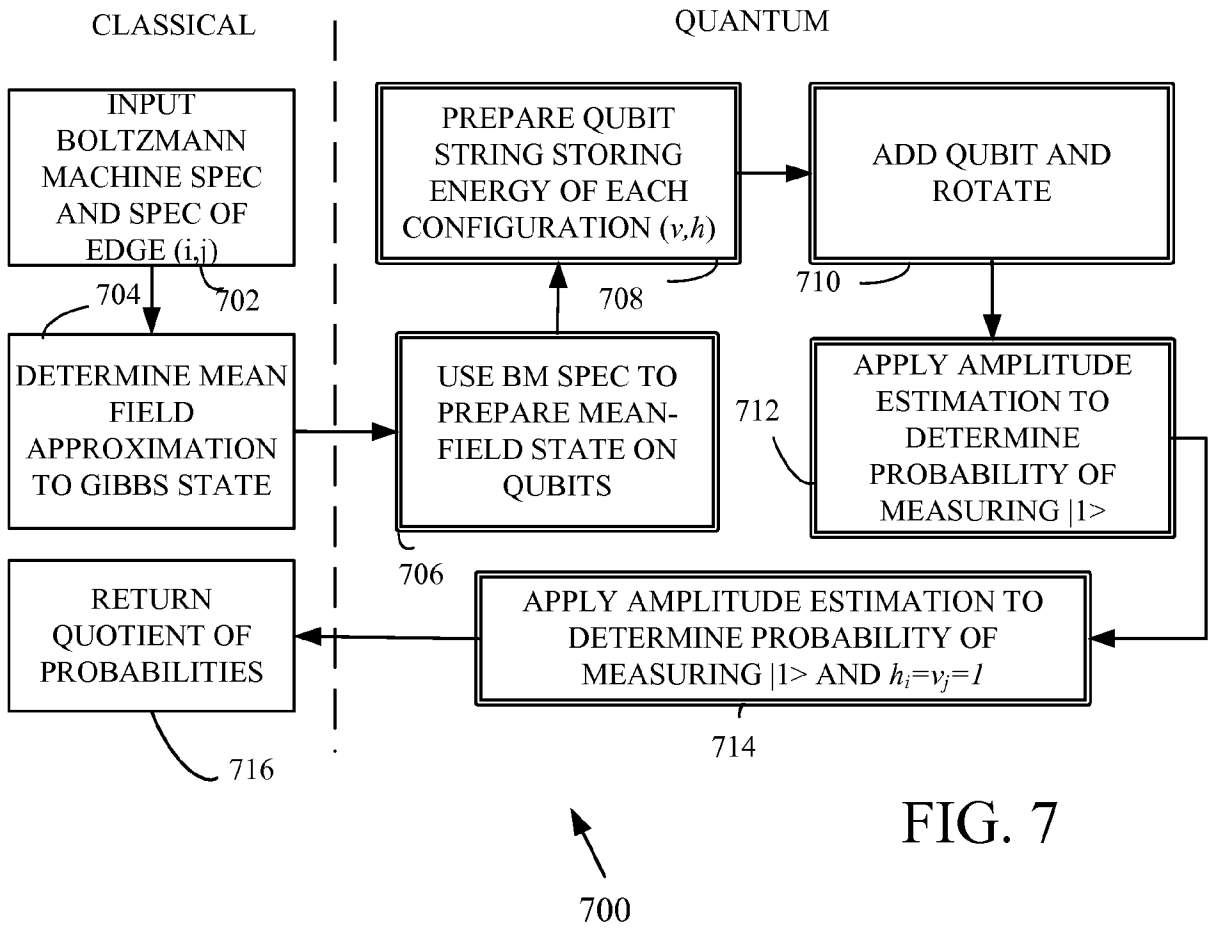

CLASSICAL
QUANTUM
INPUT BOLTZMANN MACHINE SPEC AND SPEC OF EDGE (i,j)
PREPARE QUBIT STRING STORING ENERGY OF EACH CONFIGURATION (v,h)
ADD QUBIT AND ROTATE
704
702
708
710
DETERMINE MEAN FIELD APPROXIMATION TO GIBBS STATE
USE BM SPEC TO PREPARE MEAN-FIELD STATE ON QUBITS
712
APPLY AMPLITUDE ESTIMATION TO DETERMINE PROBABILITY OF MEASURING |1>
706
RETURN QUOTIENT OF PROBABILITIES
APPLY AMPLITUDE ESTIMATION TO DETERMINE PROBABILITY OF MEASURING |1> AND $h_i=v_j=1$
714
716
700

FIG. 7

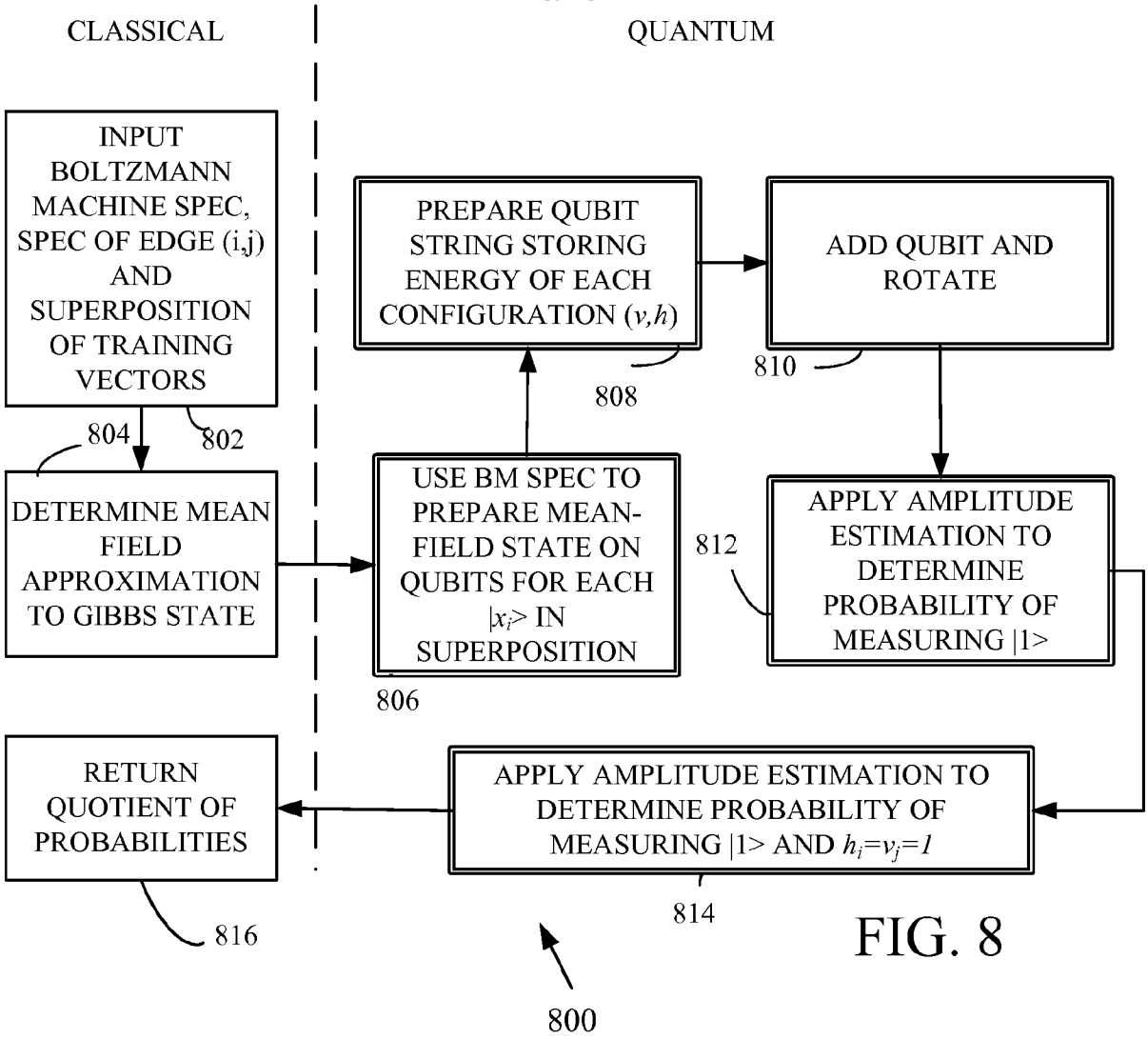

CLASSICAL
QUANTUM
INPUT BOLTZMANN MACHINE SPEC, SPEC OF EDGE (i,j) AND SUPERPOSITION OF TRAINING VECTORS
802
804
DETERMINE MEAN FIELD APPROXIMATION TO GIBBS STATE
USE BM SPEC TO PREPARE MEAN-FIELD STATE ON QUBITS FOR EACH $|x_i\rangle$ IN SUPERPOSITION
806
PREPARE QUBIT STRING STORING ENERGY OF EACH CONFIGURATION (v,h)
808
ADD QUBIT AND ROTATE
810
APPLY AMPLITUDE ESTIMATION TO DETERMINE PROBABILITY OF MEASURING |1>
812
APPLY AMPLITUDE ESTIMATION TO DETERMINE PROBABILITY OF MEASURING |1> AND $h_i=v_j=1$
814
RETURN QUOTIENT OF PROBABILITIES
816
800

FIG. 8

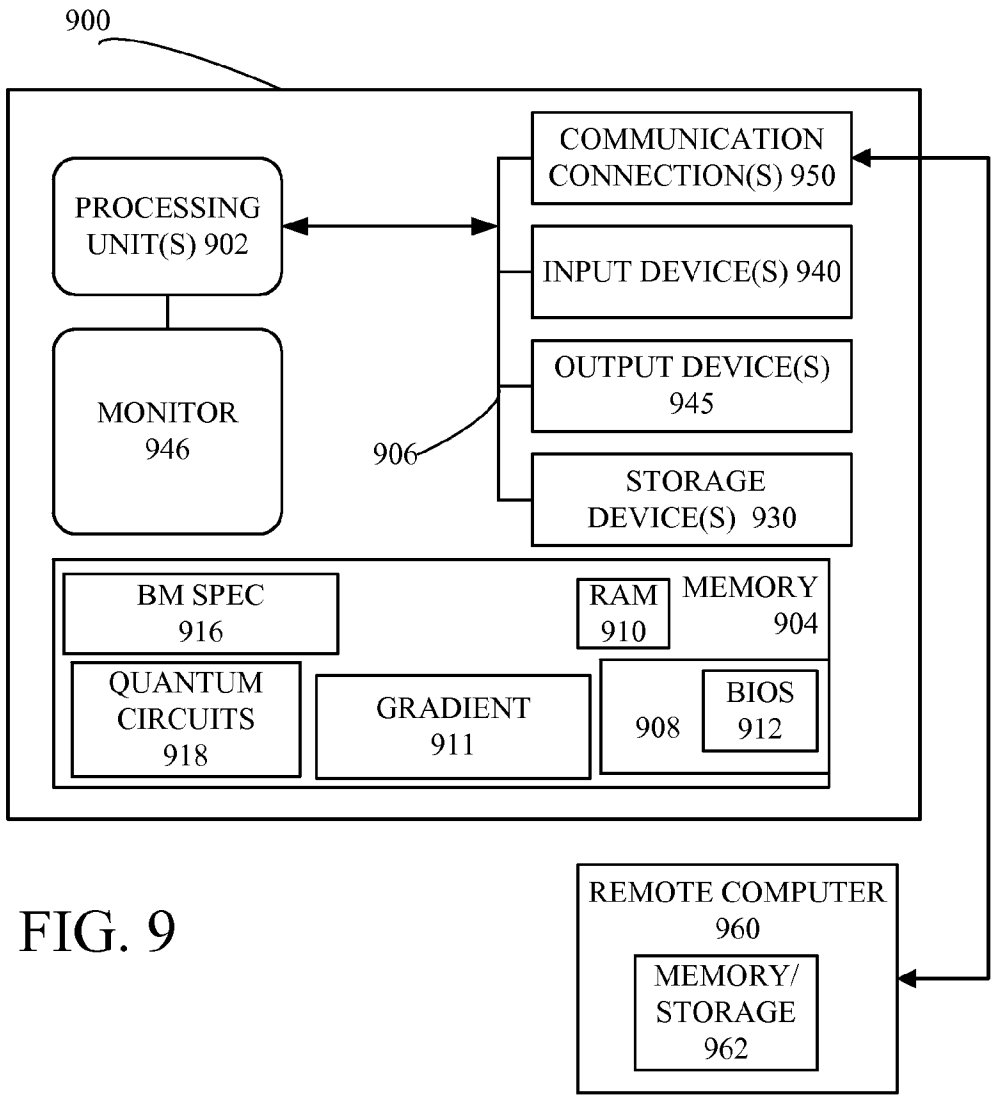

900
COMMUNICATION CONNECTION(S) 950
PROCESSING UNIT(S) 902
INPUT DEVICE(S) 940
OUTPUT DEVICE(S) 945
MONITOR 946
906
STORAGE DEVICE(S) 930
BM SPEC 916
RAM 910
MEMORY 904
QUANTUM CIRCUITS 918
GRADIENT 911
908
BIOS 912
REMOTE COMPUTER 960
MEMORY/ STORAGE 962

FIG. 9

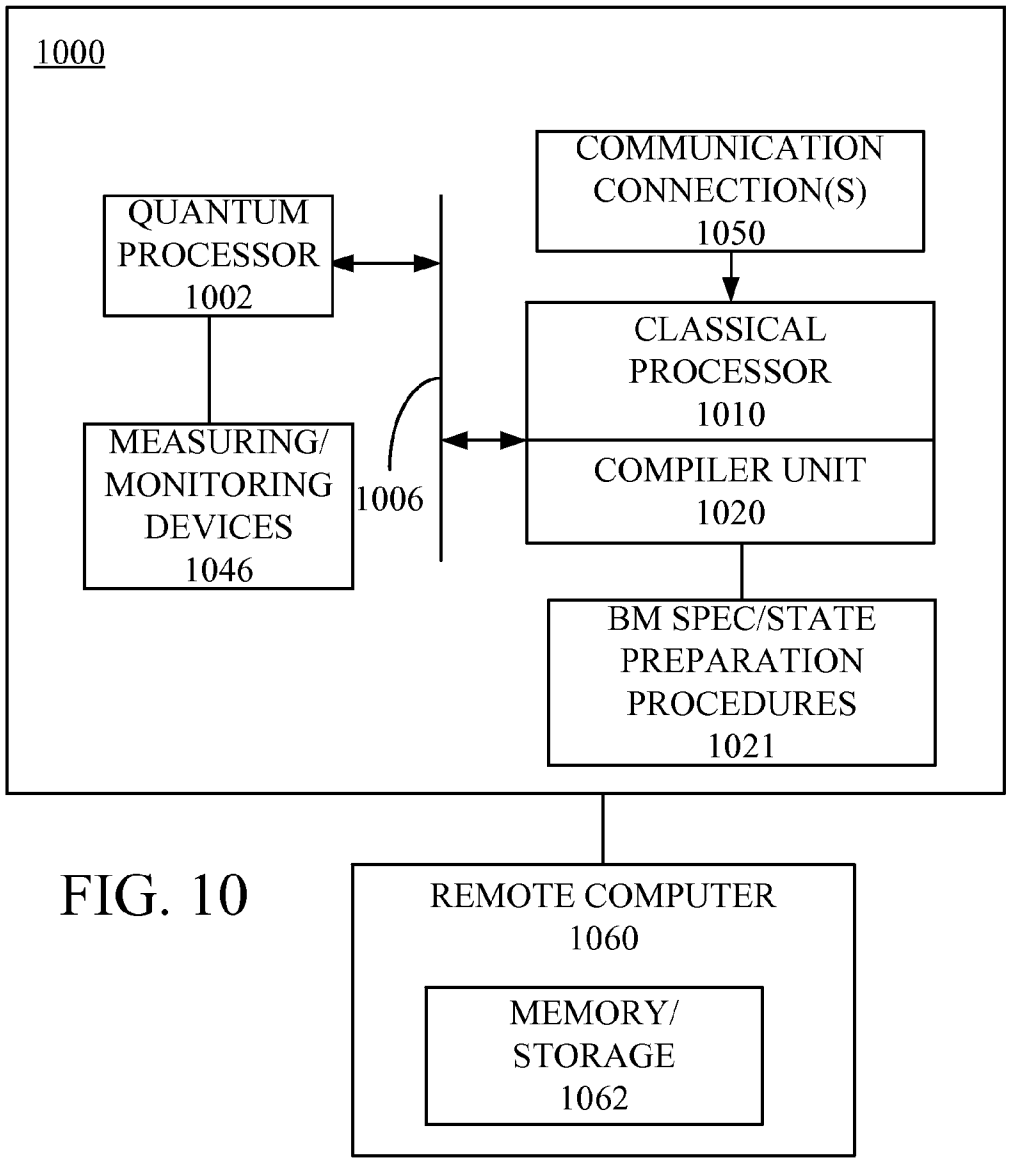

1000
COMMUNICATION CONNECTION(S) 1050
QUANTUM PROCESSOR 1002
CLASSICAL PROCESSOR 1010
COMPILER UNIT 1020
MEASURING/ MONITORING DEVICES 1046
1006
BM SPEC/STATE PREPARATION PROCEDURES 1021
REMOTE COMPUTER 1060
MEMORY/ STORAGE 1062

FIG. 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2015/062848

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06N99/00
ADD.

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)
G06N

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	N. Wiebe et al: "Quantum inspired training for Boltzmann machines", arXiv:1507.02642v1, 9 July 2015 (2015-07-09), XP055252800, Retrieved from the Internet: URL:http://arxiv.org/abs/1507.02642v1 [retrieved on 2016-02-19] paragraphs 2-4 in section 1	1-15
X	V. DUMOULIN ET AL: "On the challenges of physical implementations of RBMs", PROCEEDINGS OF THE 28TH AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE (AAAI'14), vol. 2, 27 July 2014 (2014-07-27), pages 1199-1205, XP055252615, ISBN: 978-1-57735-678-3 the whole document	1-15

-/--

[X] Further documents are listed in the continuation of Box C.

[X] 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

"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
7 March 2016	14/03/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Douarche, Nicolas

3

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014/187427 A1 (MACREADY WILLIAM G [CA] ET AL) 3 July 2014 (2014-07-03) paragraphs 154-156, claim 1 and figure 11	1-15
X	P. LI ET AL: "A hybrid quantum-inspired neural networks with sequence inputs", NEUROCOMPUTING, vol. 117, 4 March 2013 (2013-03-04), pages 81-90, XP028563511, DOI: 10.1016/j.neucom.2013.01.029 sections 1 and 3-5	1-15
X	M. Denil, N. de Freitas: "Toward the implementation of a quantum RBM", NIPS'24 workshop on deep learning and unsupervised feature learning, 16 December 2011 (2011-12-16), XP055252616, Retrieved from the Internet: URL:https://deeplearningworkshopnips2011.files.wordpress.com/2011/12/27.pdf [retrieved on 2016-02-19] sections 1, 3 and 5	1-15
A	J. J. Herman: "General quantum computational networks using nonlinear operators", arXiv:0709.0883v2, 7 September 2007 (2007-09-07), XP055252618, Retrieved from the Internet: URL:http://arxiv.org/abs/0709.0883v2 [retrieved on 2016-02-19] sections 2.1, 2.3 and 2.5	1-15
X	S. P. JORDAN: "Fast quantum algorithm for numerical gradient estimation", PHYSICAL REVIEW LETTERS, vol. 95, no. 5, 050501, 28 July 2005 (2005-07-28), XP055252740, DOI: 10.1103/PhysRevLett.95.050501 abstract	1-15
X	B. RICKS, D. VENTURA: "Training a quantum neural network", PROCEEDINGS OF THE 17TH ANNUAL CONFERENCE ON NEURAL INFORMATION PROCESSING (NIPS'16), 3 December 2003 (2003-12-03), pages 1019-1026, XP055252627, ISBN: 0-262-20152-6 sections 1 and 7	1-15

3

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2015/062848

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2014187427 A1	03-07-2014	CA 2840958 A1	10-01-2013
		EP 2729903 A1	14-05-2014
		US 2014187427 A1	03-07-2014
		US 2016042294 A1	11-02-2016
		WO 2013006836 A1	10-01-2013