IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 15, 2019, accepted May 7, 2019, date of publication May 17, 2019, date of current version September 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2917442

Modeling the Game of Go by Ising Hamiltonian, Deep Belief Networks and Common Fate Graphs

ALFONSO ROJAS-DOMÍNGUEZ[1], DIDIER BARRADAS-BAUSTISTA[2], AND MATÍAS ALVARADO[3]

[1]Tecnológico Nacional de México (Campus León), León 37290, Mexico
[2]Catalysis Center, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia
[3]Department of Computer Science, Center for Research and Advanced Studies (CINVESTAV) – IPN, Mexico City 07360, Mexico

Corresponding author: Matías Alvarado (matias@cs.cinvestav.mx)

This work was supported in part by the ABACUS and CONACYT Grant EDOMEX-2011-C01-165873 (D. Barradas) and Grant CÁTEDRAS-2598 (A. Rojas).

ABSTRACT Three different models of the game of Go are developed by establishing an analogy between this game and physical systems susceptible to analysis under the well-known Ising model in two dimensions. The Ising Hamiltonian is adapted to measure the energy of the Go boards generated by the interaction of the game pieces (stones) as players make their moves in an attempt to control the board or to capture rival stones. The proposed models are increasingly complex. The first or *Atomic-Go* model consists of the straightforward measurement of local energy employing the adapted Ising Hamiltonian. The second or *Generative Atomic-Go* model employs a Deep Belief Network (a generative graphical model popular in machine learning) to generate board configurations and compensate for the lack of information in mostly-empty boards. The third or *Molecular-Go* model incorporates Common Fate Graphs, which are an alternative representation of the Go board that offers advantages in pattern analysis. The simulated games between different Go playing systems were used to test whether the models are able to capture the energy changes produced by moves between players of different skills. The results indicate that the latter two models reflect said energy differences correctly. These positive results encourage further development of analysis tools based on the techniques discussed.

INDEX TERMS Artificial intelligence, graphical model, machine learning, Monte Carlo methods, pattern analysis, unsupervised learning.

I. INTRODUCTION

The game of Go is a two-player board game with a long history. Although its precise origin is unknown, its use can be traced back at least 2,500 years, making Go one of the oldest strategy-oriented board games, together with draughts, chess and backgammon. Go is probably the oldest board game that has been continuously played to the present day and is currently played by around 20 million players globally. More importantly, nowadays the analysis of Go is made relevant because the mathematical and algorithmic methods for the automation of Go-playing are useful to deal with major challenges in natural sciences, like biology, genomics and medicine; in engineering and computer science for pattern analysis and data analysis; and in social sciences to obtain conclusions from the analysis of interactions in social nets.

The objective of each of the players in Go is to control as much of the board as possible, by strategically placing their pieces, named stones, in such a way that completely (or as thoroughly as possible) surround rival stones while defending against the rival player that attempts to do the same. Traditionally, Go is played with black stones against white stones, on a square board with 19×19 locations (strictly speaking the board is a grid and the locations correspond to its crossings) where the players place their stones in alternating turns, with black playing first. Once a stone is placed on the board it will not be moved anymore, except if *captured* by rival stones placed in immediately neighboring locations (Go only considers 4-connectivity, and a group of connected allied stones becomes a single larger stone),

The associate editor coordinating the review of this manuscript and approving it for publication was Massimo Cafaro.

in which case the captured stone will be removed from the board. A stone's *liberty* is one empty location on the board immediately adjacent to that stone. So, each individual stone may have up to 4 liberties, and keeping these liberties is essential for a player to maintain a competitive condition. The game ends when both players are unable or unwilling to make another move. After the game has ended, the winner is defined by counting the controlled territory and the rival stones captured by each player. There are a few more rules that apply in particular situations, but what has been described so far are the basic rules that allow the understanding of the game. For an extensive discussion of Go, please consult [50] and [12].

From a game theory point of view, the game of Go is a zero-sum and complete information game; despite the simplicity of its rules, Go has a vast state space. Since each board location can be in one of three states (it can either be empty or occupied by a black or a white stone), the state space is of size $3^{19\times 19}$, giving the game a very high combinatorial complexity. This makes Go a challenging game to learn and to analyze. Serious Go players are ranked according to their ability in the game, and the strategies of top-ranked players are often studied by rivals and enthusiasts wishing to improve their skills. Several situations and *tactics* can be easily identified; the essential ones are referred to by the following terms in Go parlance:

- *Connection*: when a stone occupies a liberty shared by other stones of the same color, a bigger stone is produced by considering the group of connected stones as a single stone with its size equal to the number of individual connected stones.
- *Eye*: a single empty space inside a group is called an eye. In other words, an eye is a liberty that is shared by four allied stones that are connected to each other (through other allied stones). A group of stones with one single liberty can be captured by filling its single eye. In contrast, a group of stones that contains at least two eyes, cannot be captured.
- *Net*: allied stones surrounding rival stones form a net; liberties may be left to the surrounded stones.
- *Ladder*: a net that leaves only one liberty to the surrounding stones, forms a ladder.
- *Atari*: a stone is in atari if it has only one liberty left and is thus in danger of being captured.
- *Life & death*: a stone is considered alive if it cannot be captured, and it is considered dead if it cannot avoid being captured (even if it is not actually captured yet). Dead stones equal captured stones when computing a player's score at the end of a game.

Strategies in Go are sequences of tactics that each player employs towards winning the game. Efficient strategies are hard to master due to the number of potential moves and because they require a good understanding of the board configurations. From a computational point of view, designing efficient strategies is also a hard computer challenge [2], [16]. Considering the size of the state space and an average of 200 moves per game, the automation of Go strategies to win a game is vastly complex [2], [7], [11], [31], [53].

Computer systems able to play Go have been created [9], [36] and ranked under similar criteria used to rank human players. These computer players are built around different techniques, such as heuristic-search [14], [46], machine learning [31], [55], [56] and pattern recognition methods for identifying eyes, ladders and nets [53], [57], [58]. Monte Carlo Tree Search (MCTS) has been extensively used for simulation-based search algorithms [10], [14], [15], but this approach is highly time-consuming. A-priori knowledge-based heuristics have been added to MCTS to construct strategies to advance computer Go [27]. In 2016, the then top-ranked system AlphaGo [44] employed intelligent data mining of Go-game repositories to identify successful gaming patterns. To classify these patterns and to learn from them, AlphaGo and subsequent systems created by Google's DeepMind,[1] were built around deep neural networks [44]. Currently, AlphaZero, another system from DeepMind, can be considered the strongest Go player in history, after defeating even the broadly celebrated AlphaGoZero [45]. Interestingly, AlphaGoZero and AlphaZero developed their gaming skills primarily by playing thousands of games against different versions of similar systems, thus making them Artificial Intelligence systems that can truly learn from experience. Beyond the required and learned abilities to play Go, it is desirable to develop a comprehension of the nature of Go playing as well as the commonalities that share with complex phenomena in nature and social matters. Because of this, the focus of this work is not on the techniques used to produce effective and efficient Go-playing systems, but rather, in developing novel techniques for analysis of the game.

Interest in the game of Go from a computational perspective can be traced back to at least 1962. In that year, Remus published a paper entitled "Simulation of a learning machine for playing GO" where he described a computerized system for playing GO composed of a lexicon, a heuristic computer, and a random number generator [40]. Said study includes probably one of the earliest mathematical formulations of the rules of Go. In the 21st century, several more studies have been devoted to the game, and we distinguish at least two separate avenues of interest: one including sophisticated learning-based techniques for implementing efficiently computerized/AI-based Go players; and a second one grouping the mathematical models of the game, developed for its analysis as a case study of different theories [59], not necessarily to produce competitive Go playing systems. In this second group we can locate the work of Graepel and colleagues [17], [18], [49] around Common Fate Graphs (also used in [19], [38]) and other graph models, as well as the work of Sato and colleagues [41]–[43] regarding the connectedness of stones for analysis of the game. The paper by Harré and colleagues is worth noticing [20] because of the similarity between its purpose and that of the present work. Namely,

[1] https://deepmind.com/research/

the authors examine game trees for Go and then apply concepts from information theory, particularly *Entropy* and *Mutual Information* (MI), to quantitatively evaluate and study the complexity of decisions made by (human) Go players with different playing skills. Their positive results regarding the detection of trends in the entropy and MI as functions of the players' skills encouraged the present work.

Particularly, in this paper, the game of Go is studied by establishing an analogy between the game and physical systems susceptible to analysis under the well-known Ising model. The important concepts and techniques needed to develop our models for analysis of Go are discussed below.

II. MATERIALS AND METHODS

A. THE ISING MODEL

The Ising model (Ising, 1945) was developed in the field of statistical mechanics to study ferromagnetism [35]. Originally, the model considers several discrete variables, each representing a particle with a magnetic spin that can take one of two states ($+1$ or -1). The interaction between these particles is described by an energy function that summarizes the state of the system. This energy function is often referred to as the Hamiltonian (a term inherited from Hamiltonian mechanics). Essential characteristics of a system can be studied through its Hamiltonian. For instance, a given material such as nickel is ferromagnetic when the magnetic moments of its atoms are aligned to each other, a situation which occurs at low temperatures [8]. In this case, the energy of the system is nonzero. At higher temperatures, the magnetic moments are no longer aligned but tend to cancel each other, resulting in the total energy of the system equal to zero, and the material becomes paramagnetic.

Thus, monitoring the value of the Hamiltonian can be used to detect phase transitions, such as the change in the magnetic properties of a material that becomes ferromagnetic or paramagnetic. State of matter transitions, such as from solid to liquid and liquid to gas, can also be studied through the Hamiltonian under the Ising model. Other examples include quantum phase transitions, dynamic phase transitions, and topological (structural) phase transitions. In these types of systems, other control parameters take the place of the temperature. In this work, the Hamiltonian is employed to measure the 'energy' in Go boards, based on the interaction between the stones.

The Ising model in two dimensions is perhaps the simplest statistical model to show a phase transition and typically considers that the particles in the system are arranged in a squared lattice. For this model, the energy interactions are described by the Hamiltonian:

$$H(x_1, \ldots, x_n) = -\sum_{ij} w_{ij} x_i x_j - \mu \sum_i h_i x_i \qquad (1)$$

where w_{ij} stands for the interaction between particles x_i and x_j, μ represents the magnitude of an external magnetic field (if not present, $\mu = 0$), and h_i is the magnetic field contribution at location i (for a homogeneous external field, $h_i = 1$).

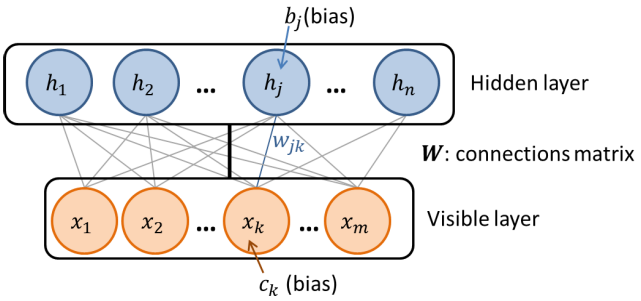


FIGURE 1. A Restricted Boltzmann Machine. Relate this figure to Eqs. (2) and (3).

Since the Ising model of interaction between elements in a system is quite general and simple [8], it has been adapted to describe the emergence of ordering in numerous systems in physics [39], chemistry [29], thermodynamics [52], biology [34], medicine [51], sociology & economy [3], technology applications in medical images [37], or systems-biology for medical tools [54]. All of these systems are assumed as constituted by discrete variables arranged in lattices and subject to extended Ising-like interaction rules. In this work, following our previous efforts [4], [59], the Ising model is applied to the analysis of the game of Go. The resulting models are described in Section III.

B. DEEP BELIEF NETWORKS

A Deep Belief Network (DBN) can be thought of as being composed of several Restricted Boltzmann Machines (RBMs), stacked one on top of another [24]. RBMs were invented circa 1985 by Hinton, Sejnowski and Ackley [1], [25] following the work of Hopfield [28] and Little [32] in Energy Models [26]. Each RBM is a network composed of two layers of units, called visible units and hidden units (see Fig. 1). The units of an RBM are binary and stochastic, with a joint probability distribution following a Boltzmann distribution $p(\mathbf{z}, \mathbf{h}) = \exp(-E(\mathbf{z}, \mathbf{h}))/\mathcal{Z}$, where $\mathcal{Z}$ represents the partition function. The energy of the RBM is defined as:

$$E(\mathbf{x}, \mathbf{h}) = -\mathbf{h}^{\mathrm{T}}\mathbf{W}\mathbf{x} - \mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{h} \qquad (2)$$

Notice that (2) is in fact the Ising Hamiltonian (1) in vector form, where the effect of the external magnetic field has been split into two parts, one for the visible units $\mathbf{x}$ and another for the hidden units $\mathbf{h}$ in the network. This fact can be made obvious by presenting the energy in a scalar form:

$$E(\mathbf{x}, \mathbf{h}) = -\sum_k \sum_j w_{jk} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j \qquad (3)$$

from which we can see that RBMs fall under the Ising model and the main difference between an RBM and the standard Ising model is that the variables in the RBM are organized in layers and no connections between the variables within each layer are allowed (hence the name *restricted*). Thus, this type of networks are ideally suited for learning models and particularly, Ising models.

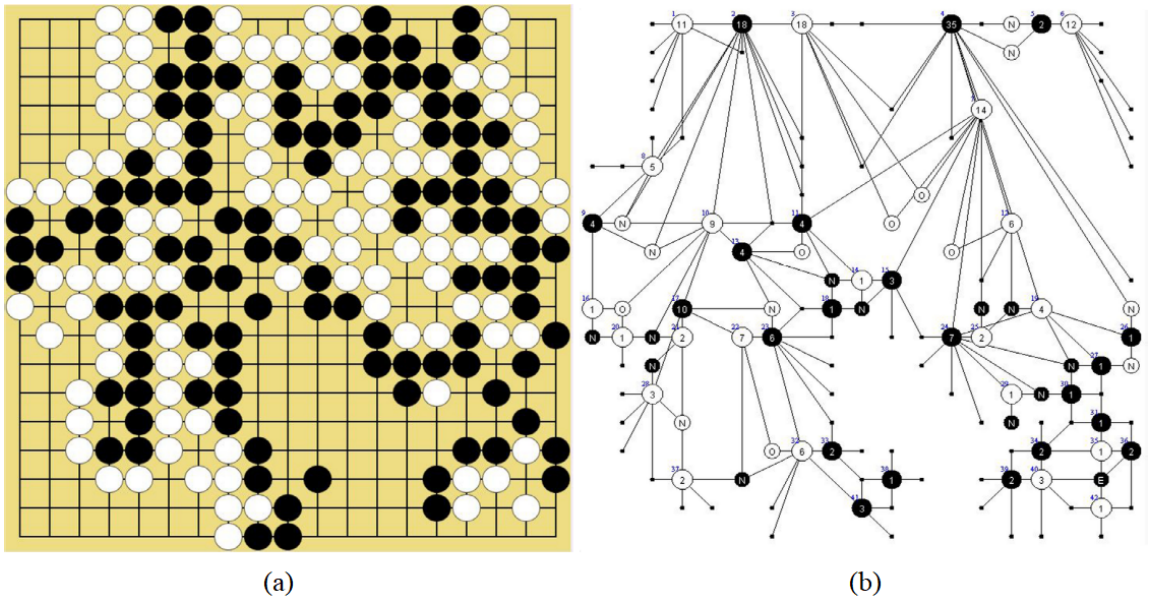


FIGURE 2. (a) A Go board and (b) its corresponding representation using a CFG. See main text for an explanation.

The parameters of an RBM can be tuned by repeatedly setting its visible units to some values (an input vector), computing the states of its hidden units and performing an adjustment to the parameters. This is called *training* the network and occurs in an unsupervised manner via a particular algorithm known as Contrastive Divergence [5], [6], [21], [22]. The objective of the training is to get the network to learn relationships between the variables represented by the visible units, or in other words, their probability distribution. Once the network has been trained, then we can use it to sample said variables, which means generating new ''input'' vectors. This is why RBMs belong to a family of models known as *generative* models.

By stacking several RBMs together, the learning is improved, and the network is thus known as a DBN [23], [24]. In this work, a DBN is employed to generate Go board configurations that will help to measure the energy in boards with mostly empty locations, which appear at the beginning of Go games. Details are provided in Section III-B.

C. COMMON FATE GRAPHS

The definition of the elements of the Ising energy function can help algorithms to compute the energy of stone patterns at the successive Go moves, accounting for each state of dominance. A Common Fate Graph (CFG) is an alternative, sophisticated representation of the stones on a Go board [17]. As the name implies, the CFG consists of a graph where connected stones of the same color are merged into one vertex. The graph enables better handling of tactics and strategies for analysis of the game [19].

This representation is shown in Fig. 2 where each Go stone is a CFG principal node, labeled with the number of single stones that compose it, and each stone's liberty is a CFG secondary node. For instance, the white stone in the superior left corner in Fig. 2 is 11 single stones, it possesses six liberties (one shared with the 5-stones group below), and one black liberty (shared with the 18-stones group to the right). All of this is more easily appreciated through the CFG than on the raw board.

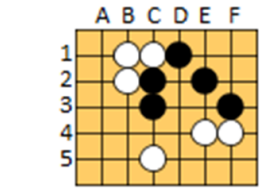
Also, a Go gaming state representation by CFGs embraces each stone's linked relationship with allies, adversaries, and liberties. By using CFGs, the Go sequence of moves (tactics deployment) during a game, is easily logged, as well as the follow up of the evolution of game interaction (depicted in a lattice graph). By regarding the relative board position among allies and adversaries on the base of the CFG, this technique permits to define the Ising energy function and the design of algorithms to quantify the force of interactions among black and white stones. CFGs are employed to build the model described in Section III-C.

III. ISING-GO MODELS

The Ising model can be applied to modeling the dynamics of the interactions between allied or rival stones in their struggling for territory control in GO. In this section, three different models are proposed, two more 'traditional' models that consider each of the stones in the game as individual particles, thus called *atomic-Go* models, and a third one that employs CFGs [17] to model connected stones, and is called a *molecular-Go* model.

A. ATOMIC-GO ISING MODEL

Our first model considers only the interaction between adjacent stones, taking individually (thus $w_{ij} = 1$ and $\mu = 0$). Recalling that in Go only 4-connectivity exists, this means that any stone will interact with other stones located imme-



(a)

$$\mathbf{A} = \begin{array}{c|cccccc} & A & B & C & D & E & F \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 \\ 3 & 0 & 0 & 1 & 0 & 0 & 1 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

(b)

$$\mathbf{B} = \begin{array}{c|cccccc} & A & B & C & D & E & F \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 1 & 1 \\ 5 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}$$

(c)

$$\mathbf{C} = \begin{array}{c|cccccc} & A & B & C & D & E & F \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 0 & 1 & 0 & 1 \\ 3 & 1 & 1 & 0 & 1 & 1 & 0 \\ 4 & 1 & 1 & 1 & 1 & 0 & 0 \\ 5 & 1 & 1 & 0 & 1 & 1 & 1 \end{array}$$

(d)

FIGURE 3. Representation of a Go board used to train an RBM.

diately to the north, west, south or east of its location. For an arbitrary board configuration (representing an arbitrary time during a game), the interaction between a central stone x_i and stones x_j in its neighborhood $\mathcal{N}$, results in local energy:

$$s_i = \sum_{j \in \mathcal{N}} x_i x_j = x_i \sum_{j \in \mathcal{N}} x_j \tag{4}$$

where $x_i, x_j \in \{-1, +1\}$ for a black stone or a white stone, respectively, and equals zero if the board location is empty.

The total energy of the board at a time t is simply the sum of the local energies of the corresponding board configuration.

$$H_{Atomic}(t) = -\sum_i s_i^{(t)} - \mu \sum_i h_i^{(t)} x_i^{(t)} \tag{5}$$

where h_i represents the number of stone liberties (empty board locations in the neighborhood of a stone), and contribute with an amount of energy $\mu \geq 0$ aligned with x_i.

B. GENERATIVE ATOMIC-GO ISING MODEL

Our second model inherits the features of our first model, augmented by a deep generative model that is used to fill the empty positions of Go boards, mainly in the early stages of a game. The generative model is a DBN consisting of three stacked RBMs, as described in Section II-B.

An ad-hoc encoding of the Go boards is introduced to train the RBMs. A small illustrative Go board with some black and white stones is shown in Fig. 3a; real Go boards contain 19×19 crossings, the vertical lines are labeled A to T (historically, the letter I is not used), the horizontal lines are labeled with numbers 1 to 19, and the margins of the board are included (we have not included the margin in our illustration, for simplicity). This small board can be represented by three matrices of binary values, shown in Fig. 3b - 3d. The values in the first matrix are equal to 1 if the corresponding crossing on the board has a black stone and are zero otherwise. In the second matrix, the values equal to 1 represent the presence of white stones at the corresponding crossings, and in the third matrix, the values equal to 1 represent the empty crossings on the board.

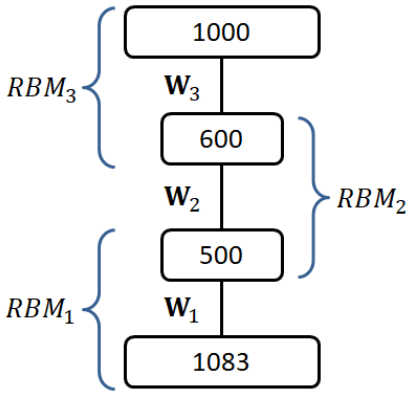


FIGURE 4. A DBN is composed of 3 RBMs stacked up together.

The elements of each of the matrices in Fig. 3 can be rearranged to form a vector of 30 binary numbers. For instance, $\mathbf{A}' = [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, \ldots, 0]$ are the elements of $\mathbf{A}$ reordered into a single row vector. We define our input vector $\mathbf{z}$ as the vector resulting from concatenation of $\mathbf{A}'$, $\mathbf{B}'$ and $\mathbf{C}'$: $\mathbf{z} = [\mathbf{A}', \mathbf{B}', \mathbf{C}']$. As can be appreciated, $\mathbf{z}$ contains all the information required to reconstruct the original Go board.

In this work, 3 RBMs were stacked together as illustrated in Fig. 4. Examining this network from the bottom to the top, we have a visible layer of $19 \times 19 \times 3 = 1083$ units, a first hidden layer of 500 units, a second hidden layer of 600 units and a final third hidden layer of 1000 units. The number of units in the visible layer corresponds to the number of variables in our representation of a Go board, as explained above. The number of units for each hidden layer was found empirically, by looking at the reconstruction error obtained for each of the RBMs during its training, see Fig. 5. For the training, we used a total of 69,100 boards in their final (i.e. most complete) configuration from Go games of expert players.[2]

Once trained, the DBN becomes a model for the probability distribution of the data from which the network was trained. Thus, the data can be sampled using the expressions:

$$p(h_j|\mathbf{z}) = 1/(1 + \exp(-(b_j + \mathbf{W}_j \mathbf{z})) \tag{6}$$

$$p(z_k|\mathbf{h}) = 1/(1 + \exp(-(c_k + \mathbf{z}^{\mathrm{T}} \mathbf{W}_k)) \tag{7}$$

where h_j represents the j-th hidden unit, z_k the k-th visible unit, $\mathbf{W}_j$ and $\mathbf{W}_k$ represent the j-th row and k-th column of $\mathbf{W}$, respectively (notation is overloaded for simplicity), and b_j and c_k are the visible and hidden biases, respectively.

Thus, $p(\mathbf{z}) = \sum_{\mathbf{h}} p(\mathbf{z}|\mathbf{h})$ is the marginal distribution of $\mathbf{z}$ if the sum is performed over all possible configurations of the hidden units, $\mathbf{h}$. In our problem, with 1000 hidden binary units in the last layer of our RBM, we have $2^{1000} \approx$ 1E301 possible configurations, so computing $p(\mathbf{z})$ in that way becomes intractable. Besides, there are many configurations of our hidden units that are either trivial or uninteresting for

[2]https://senseis.xmp.net/

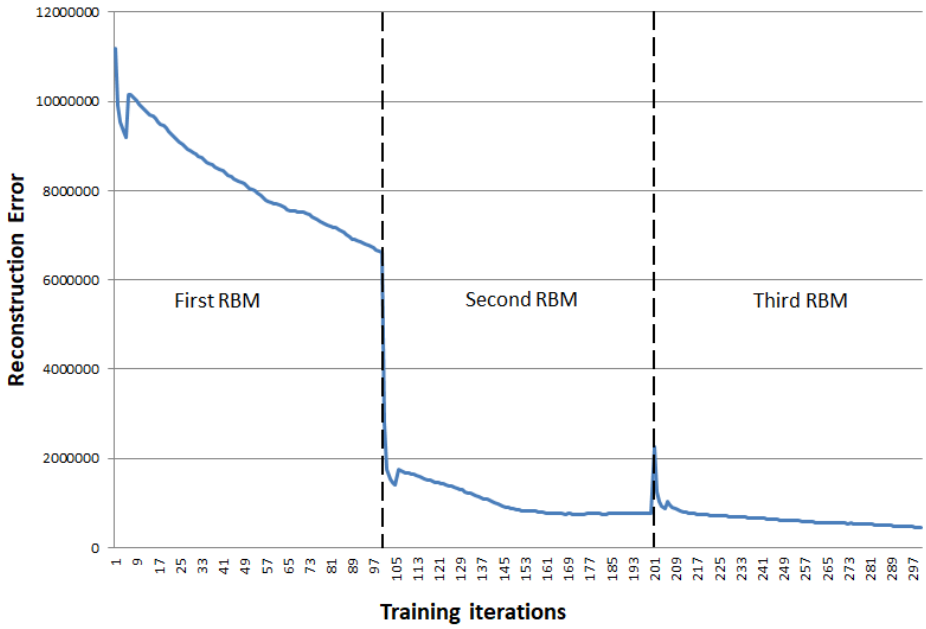


FIGURE 5. Training of a DBN composed of 3 RBMs.

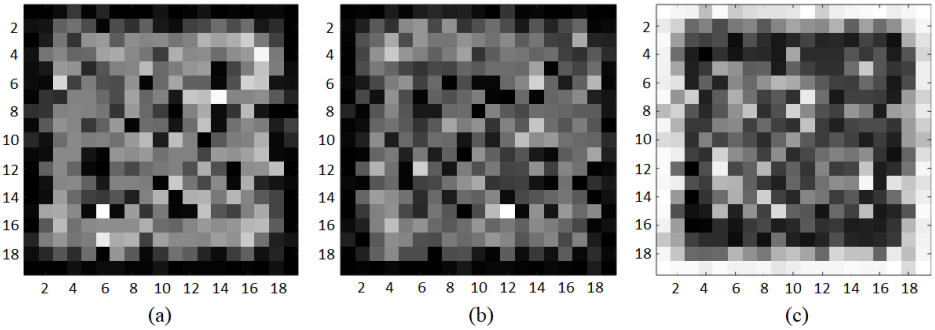


FIGURE 6. (a) Probability of black stone; (b) probability of white stone; and (c) probability of empty crossing.

our purposes, so instead of trying to use every possible configuration, we use our network to approximate the probability distribution of our hidden units and then use this to generate new instances of our visible units. By presenting each of the 69,100 boards to our trained network we can compute the states of the hidden units in the final layer of our DBN and produce an estimate of the probability distribution of those hidden units $\hat{p}(\mathbf{h})$. Next, we sample from $\hat{p}(\mathbf{h})$ and run the DBN backward (from top to bottom) to obtain new instances of $\mathbf{z}$. After generating enough samples, we obtain a good estimate $\hat{p}(\mathbf{z})$, of our desired distribution. This represents the most probable configuration in a game of Go, as learned from expert games. The estimate obtained from one million instances generated in this fashion is shown in Fig. 6. The probabilities of occurrence of each state for each position can be read directly from these matrices. It should be noticed that the probabilities recorded in these three matrices sum to 1 for every position, since there are only three possible states of the board (black stone, white stone or empty crossing). In our second model, called Generative Atomic-Go Ising model, we use said probabilities of occurrence to fill the empty positions of the boards before computing the energy in the way described in section III-A.

C. MOLECULAR-GO ISING MODEL

Our third model considers the existence of compound stones, formed by connected individual stones. For this purpose, we employ a more efficient representation of the Go board throughout a game, by means of CFGs. In this CFG-based representation, a *molecular stone* $\mathbf{x}_i$ is defined by:

$$\mathbf{x}_i = c_i(n_i^{k_i+1}) \tag{8}$$

where $n_i \in \mathbb{Z}^+$ is the size of the molecular stone (equal to the number of connected atomic stones forming it), $k_i \in \{0, \mathbb{Z}^+\}$ is the number of eyes in the molecular stone, and $c_i \in \{-1, +1\}$ corresponds to the stone color (black or white, respectively).

In this expression, the strength of the stone is weighted by the number of its eyes, so that the value of a stone with one eye is squared, and that of a stone with two eyes (a stone that

TABLE 1. Proposed weight values for the Go tactics.

Tactic	Value of r_t
Ladder	0.8
Net	0.6
Eye	1.4

cannot be captured) is cubed. If a stone has no eye, $\mathbf{x}_i$ just indicates the size and color of that stone.

Besides the strength of a molecular stone, it is also important to quantify its interaction with other stones considering their involvement in some of the different tactics described in Section I. Said interaction is represented by w_{ij} in the Ising model (1), and is now defined in our third Ising-Go model as:

$$w_{ij} = r_t(i,j) \tag{9}$$

where $r_t(i,j)$ is a multiplicatory factor corresponding to a weight assigned to each possible tactic in which the stones $\mathbf{x}_i$ and $\mathbf{x}_j$ participate. In other words, the factor r_t weights the power of each tactic t: eye (r_{eye}), net (r_{net}) or ladder (r_{ladder}). The proposed values for r_t are reported in Table 1; these are based on the experience of well-ranked human Go players[3] and our own understanding of the game.

Introducing (8) and (9) into the Hamiltonian (1), we have, at a time t:

$$H_{molecular}(t) = -\sum_{ij} w_{ij}^{(t)} \mathbf{x}_i^{(t)} \mathbf{x}_j^{(t)} - \mu \sum_i h_i^{(t)} \mathbf{x}_i^{(t)} \tag{10}$$

where, just as in (5), h_i represents the sum of liberties that stone i possesses.

The energy function (10) can be employed to compute the power of the adversary groups of stones in a Go board and thus quantify their corresponding board dominance. Usually, a group of three stones or more appear as a *net*, so the existence of the net tactic throughout a game is frequent. Also, from the middle of a game onward, at least one large stone in a net is present on the board, so the power of the net tactic is significant. A *ladder* is not a frequent tactic, but its occurrence results in a strong position on the board. One compound stone may have one, two, or more internal *eyes*, which 'multiply' its strength: a stone with one eye is alive, a stone with two eyes cannot be captured, etc. The *connection* tactic results in small stones becoming a larger one; thus, a connection is indirectly quantified in the size of a compound stone. As before, the 'magnetic field' influence h_i on each stone is made analogous to the sum of liberties that stone i possesses. For simplicity, the scaling parameter $\mu = 1$ in this proposal.

IV. EXPERIMENTAL EVALUATION

In this section, the three different Go-Ising models are tested by applying each of them to compute the energy of Go games and verify whether the models can detect energy differences or not. In order to introduce controlled differences between the skills of the players, several thousand Go games were simulated employing three different artificial players (2,500 games per player.[4]) A brief description of these players is provided before reporting the experimental results.

A. GO ARTIFICIAL PLAYERS

— GNU Go.- is a sophisticated program that plays the game of Go, developed by the Free Software Foundation. A complete list of contributors and extensive documentation can be found in the project website.[5] GNU Go is a strong player that includes dedicated routines for moves generation/valuation; tactical analysis; influence computation; analysis of strings and groups of connected stones; pattern analysis, etc. The strength of the player is controlled by the *level* parameter. The default level is 10, which is the maximum strength level. GNU Go is the strongest player of the three used in this work.

— SmartGo.- is a commercially developed go-playing program (Smart Go, Inc.)[6] which includes an advanced graphical user interface and a large variety of user tools for game analysis. As a player, it analyzes and selects the best moves (by default it looks at the top 24 moves) for a given scenario. This is possible through a customizable library of games from which SmartGo can draw its moves. Although the playing strength of SmartGo cannot be adjusted, according to its authors, the program will play marginally better when given more time to analyze the games. We consider SmartGo to be a medium/strong level player.

— Fuego.- is a collection of C++ libraries for the game of Go.[7] It was initially developed by the Computer Go group at the University of Alberta, Canada. It includes a Go player Monte Carlo tree search [44] with the Upper Confidence Bound applied to Trees algorithm [30]. Fuego is the weakest of the three players considered in this work.

In every case, the programs described above simulate games that can be stored as text files. The files comply with the purpose-specific Smart Game Format (SGF)[8] and can be easily parsed for analysis. Each SGF file contains the moves performed by each of the players in order of occurrence, together with the corresponding position on the board. From this information, the per-move energy change produced in a game can be computed under our different Go-Ising models.

B. RESULTS

Three pairs of player combinations are reported, and the changes in energy (with inverted sign, so that positive energy is considered favorable), measured employing the models described in Section III, are presented in box and whiskers diagrams (outliers were excluded from the plots for improved clarity). These diagrams are a summary or simplified representation of the data distributions; the horizontal segments represent, from bottom to top: the minimum value, first quar-

[3]Please see the Acknowledgments Section.

[4]Beyond a few hundred games, adding more games does not change the data distributions. Using 2,500 games results in very stable distributions.

[5]https://www.gnu.org/software/gnugo/

[6]https://smartgo.com/

[7]http://fuego.sourceforge.net/

[8]https://www.red-bean.com/sgf/

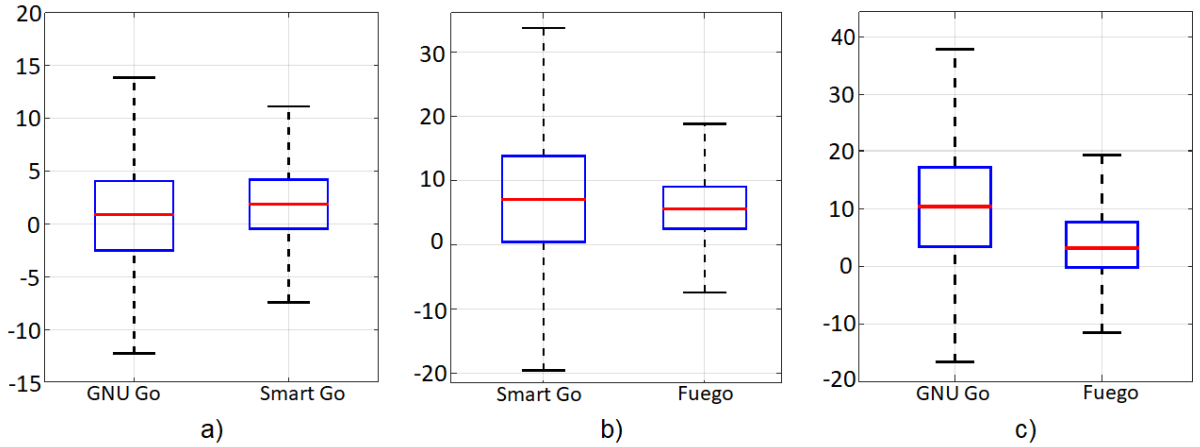


FIGURE 7. Energy comparison of different artificial players under the Atomic-Go Ising model.

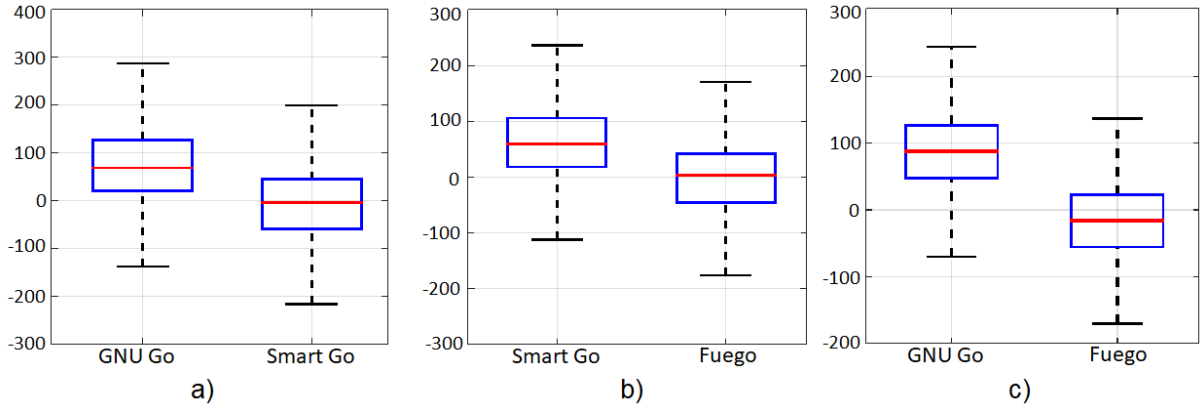


FIGURE 8. Energy comparison of different artificial players under the Generative Atomic-Go Ising model.

tile (in blue), second quartile (median, in red), third quartile (in blue), and maximum value. If the models proposed in this work are adequate, then differences in the strength between different players (relative to their rivals) should be reflected by the changes in energy that their moves produce. Stronger players make moves that result in positive energy changes, and this should be captured by the models and made apparent in the box and whiskers diagrams.

The results of applying the Atomic-Go Ising model, the Generative Atomic-Go Ising model and the Molecular-Go Ising model are presented in Figs. 7 to 9, respectively. On first examination, the box diagrams in Fig. 7 reflect some differences between the data distributions. In statistical terms, when the median line of one distribution (red line in the box and whiskers diagrams) is located inside the range of the box of another distribution, there is a possibility that the two distributions are not different from each other. This is the case in the data shown in Fig. 7a and 7b. The median lines of the distributions in Fig. 7c do not fall within the body of each other, but the distributions overlap significantly. A second aspect examined is the variance of the data distributions, which is reflected in their interquartile range (the height of the boxes). As can be observed, the larger variance corresponds to the stronger player in each panel. This means that the moves made by stronger players have larger effects on the energy, as measured by the Atomic-Go model. Combining the two aspects above, the conclusion is that this model does not capture the strength of the players sufficiently well, since the data distributions of players that are known to be different overlap each other and only differences in the range of the energy can be observed.

The box and whiskers diagrams in Fig. 8 show the results of using the Generative Atomic-Go Ising model to analyze the games. A very different scenario to that of the Atomic-Go model can be readily observed. In this case, the medians are not located inside the body of the other boxes in the comparison. This is evidence to the fact that the medians of the distributions are significantly different from each other. Also, there is significantly less overlap between the boxes, and in the case of the two more different players (GNU vs Fuego) the boxes do not overlap at all. Furthermore, according to this model, the data distribution of the stronger player is always above that of the weaker player, indicating that the moves of the stronger player result in positive energy differences with much more frequency that the moves of the weaker player. This is the behavior that the model was

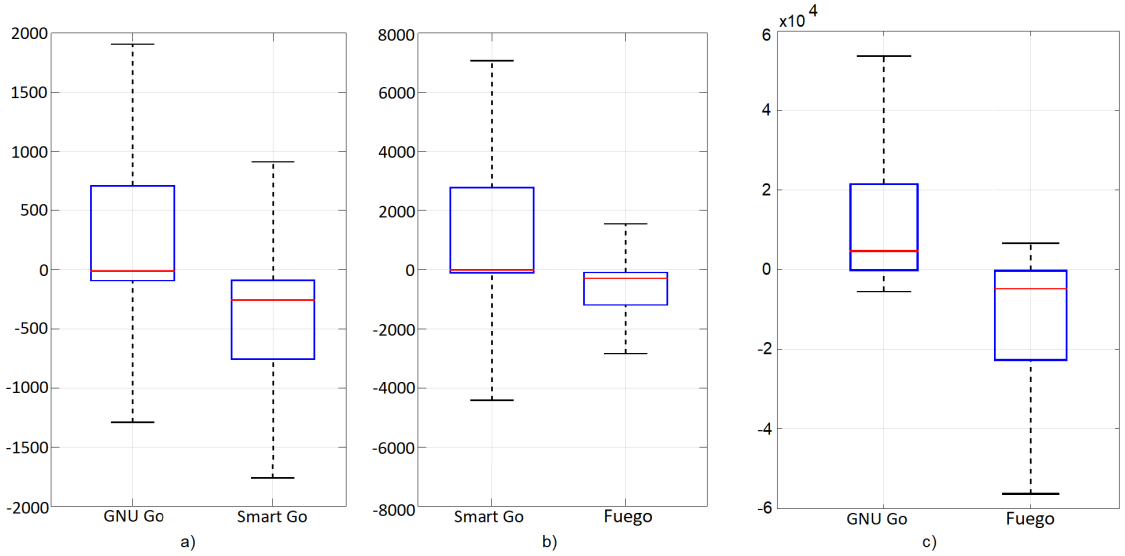


FIGURE 9. Energy comparison of different artificial players under the Molecular-Go Ising model.

expected to capture. Notice as well that the scale of the energy differences is nearly one order of magnitude larger than those found by the Atomic model. In other words, the Generative Atomic model amplifies the differences between the players. Finally, the variances of the data compared are very similar and the boxes are symmetric around the median, indicating that the data follow Normal distributions.

Regarding the results presented in Fig. 9, which correspond to the application of the Molecular-Go Ising model, it can be observed that, taken by pairs, none of the boxes representing the data distributions overlap each other. This means that, according to this model, the game moves performed by one player produce significantly different changes in energy. As with the Generative-Go model, in the Molecular-Go model the stronger player in each case produces a data distribution that is above that of the weaker player. Furthermore, under this model, the medians of the distributions are not in the middle of the boxes, but are skewed towards one end (the one end of each box that is closer to zero). In other words, the data distributions of the stronger players are right-skewed distributions whereas the distributions of the weaker players are left-skewed distributions. This is a feature of the data that the Atomic-Go and the Generative Atomic-Go models do not capture. Finally, the range of the energy difference values under the Molecular-Go model is about three orders of magnitude larger than the Atomic-Go model. This is understandable, because this model groups stones before computing the corresponding Hamiltonian, amplifying the energy values of the stone configurations in a board, depending on the number of eyes they possess. At this point it is worth noticing that the energy values measured employing the models described in this work could be normalized, depending on the user needs.

To test whether the apparent differences (or lack of difference) between the data distributions are statistically real to a certain significance level, the data was analyzed using the Wilcoxon rank sum statistical test, against the null hypothesis that the data samples come from continuous distributions with equal medians. The dependent variable is the energy difference measured and the independent variable is the pair of players, in each case. The p-values obtained are reported in Table 2 and indicate that the null hypothesis can be rejected for every case tested.

TABLE 2. p-values for the Wilcoxon rank sum test.

Model	GNU vs Smart	Smart vs Fuego	GNU vs Fuego
Atomic	5.57E-25	6.86E-8	3.19E-142
Generative Atom.	1.49E-293	1.02E-215	0
Molecular	0	0	0

Although the statistical tests show that all of our Go Ising models manage to measure the difference in the strength of players, the different models are not equally efficient nor equally useful. Under the Atomic model the energy differences are subdued to the point that it would be hard to use this model for any practical purpose of analysis of the Go games. Strictly speaking, the distributions in Fig. 7a indicate that the Smart Go player is slightly stronger than GNU Go, which is incorrect. Under the Generative Atomic-Go model, the energy differences between players are clearly distinguishable and correctly reflect the strength of the players; this model can be used to evaluate the strength of the moves in a Go match. The Molecular-Go Ising model is the most sophisticated one and the one that more strongly amplifies the differences between the players. Comparing Fig. 9a with Fig. 9b it can be observed that the Smart Go energy distribution changes signs and shapes, from a negative left-tailed distribution to a positive and right-tailed distribution. This is because the model does not simply rely on the local infor-

mation around a stone that has been played (like the Atomic models do) but also includes the interactions occurring on the rest of the board. In contrast, the Atomic models generate essentially Normal distributions. For this reason, the Molecular model is preferred.

V. CONCLUSION

The highly intricate interactions that occur between the stones in the game of Go were studied utilizing three models developed from the starting point of the 2-D Ising model originally proposed for the study of ferromagnetism. The Ising Hamiltonian was adapted to measure the 'energy' between allied and rival stones. Through the proposed models, the Hamiltonian allows quantification of a stone's power as it is placed on the board, and later on as part of identified tactics.

The models were applied to several thousand games played between artificial players. Supported by statistical tests, we can conclude that two of the Go Ising models (Generative Atomic-Go and Molecular Go) were able to adequately measure the energy differences produced on the Go boards by each of the moves of the different Go players. The remaining model (which is the simplest of the three proposed) does not measure the energy in a useful way to separate players according to their strength in the game. The most complex model (called Molecular Go Ising model), measures the energy in the interaction between ally and adversarial stones, but also weights that interaction according to the contribution of the stones to each eye, ladder or net tactics. The nature of Go gaming captured by the models suggests that low-cost algorithms based on these models could be deployed to identify the relevant strategies for success.

In this work, the models described were used to detect statistically significant differences between the skills of different players. However, the models could also be used to perform a play-by-play analysis of a game in order to predict the winner. This can be done by application of the Hamiltonian, either (5) or (10), to estimate the relative strength of each player. After a sequence of moves, the player whose moves result in the largest energy differences has the highest probability of winning the match. Similarly, the statistical analysis of the games with the help of the models could identify tactics and sequences of tactics that form effective strategies in the long run, potentially setting the foundation for Ising-model-based artificial players that employ energy measures as part of their move-selection algorithms.

Although initially, the interaction between individual Go stones is fairly simple, complex behavior emerges from the association among groups of those basic elements. Such behavior is characteristic of many large systems found in natural and social sciences. Thus, through the study of these models, advancement in the comprehension of emerging complex behaviors from other domains can be achieved, particularly to those where territory control is the main mechanism for the success of the involved entities [34], [51]. Areas for potential application are social science (for example to model business confidence, segregation, language change and stock market speculation [47], [48]) and socio-political networks [13] (like the US Senate where the party affiliations place Senators in adversarial positions [33]), etc. Another potential area for the application of the Go Ising models is the study of cancer metastasis, where the immune system and the cancer cells play the role of two rival players and the patient's tissue is the territory that the adversaries try to dominate [4]. These ideas present opportunities for future work.

ACKNOWLEDGMENTS

Carlos Villarreal Lujan (Physics Institute, UNAM) provided advice on the use and meaning of the Ising model and Hamiltonian. Ricardo Quintero Zazueta, world Go senior master, suggested test simulations using games of human experts. Mark Agostino (Curtin University) provided suggestions to improve the manuscript.

REFERENCES

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cogn. Sci.*, vol. 9, no. 1, pp. 147–169, 1985.
[2] L. V. Allis, *Searching for Solutions in Games and Artificial Intelligence*. Wageningen, The Netherlands: Ponsen & Looijen, 1994.
[3] M. Aoki, *New Approaches to Macroeconomic Modeling: Evolutionary Stochastic Dynamics, Multiple Equilibria, and Externalities as Field Effects*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[4] D. Barradas-Bautista, M. Alvarado-Mentado, M. Agostino, and G. Cocho, "Cancer growth and metastasis as a metaphor of go gaming: An ising model approach," *PLoS ONE*, vol. 13, no. 5, 2018, Art. no. e0195654.
[5] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
[6] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 153–160.
[7] D. B. Benson, "Life in the game of go," *Inf. Sci.*, vol. 10, no. 1, pp. 17–29, 1976.
[8] E. Bonaccorsi, S. Merlino, M. Pasero, and G. Macedonio, "Microsommite: Crystal chemistry, phase transitions, ising model and Monte Carlo simulations," *Phys. Chem. Minerals*, vol. 28, no. 8, pp. 509–522, 2001.
[9] B. Bouzy and T. Cazenave, "Computer go: An AI oriented survey," *Artif. Intell.*, vol. 132, no. 1, pp. 39–103, 2001.
[10] C. B. Browne *et al.*, "A survey of Monte Carlo tree search methods," *IEEE Trans. Comput. Intell. AI Games*, vol. 4, no. 1, pp. 1–43, Mar. 2012.
[11] K. Chen and Z. Chen, "Static analysis of life and death in the game of go," *Inf. Sci.*, vol. 121, nos. 1–2, pp. 113–134, 1999.
[12] J. Davies and R. Bozulich, *An Introduction to Go*. Tokyo, Japan: Ishi Press, 1984.
[13] L. M. Floría, C. Gracia-Lázaro, J. Gómez-Gardenes, and Y. Moreno, "Social network reciprocity as a phase transition in evolutionary cooperation," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 79, no. 2, 2009, Art. no. 026106.
[14] S. Gelly, L. Kocsis, M. Schoenauer, M. Sebag, D. Silver, C. Szepesvári, and O. Teytaud, "The grand challenge of computer go: Monte Carlo tree search and extensions," *Commun. ACM*, vol. 55, no. 3, pp. 106–113, 2012.
[15] S. Gelly and D. Silver, "Monte-Carlo tree search and rapid action value estimation in computer go," *Artif. Intell.*, vol. 175, no. 11, pp. 1856–1875, 2011.
[16] D. Wolfe and E. Berlekamp, *Mathematical Go: Chilling Gets the Last Point*. Wellesley, MA, USA: A K Peters, 1994.
[17] T. K. H. Graepel, M. Goutrie, M. Krüger, and R. Herbrich, "Learning on graphs in the game of go," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2001, pp. 347–352.
[18] T. K. H. Graepel, R. Herbrich, and D. Stern, "Learning belief distributions for game moves," U.S. Patent 7 647 289, Jan. 12, 2010.
[19] T. Graf and M. Platzner, "Common fate graph patterns in Monte Carlo tree search for computer go," in *Proc. IEEE Conf. Comput. Intell. Games*, Aug. 2014, pp. 1–8.
[20] M. S. Harré, T. Bossomaier, A. Gillett, and A. Snyder, "The aggregate complexity of decisions in the game of go," *Eur. Phys. J. B*, vol. 80, no. 4, pp. 555–563, 2011.

[21] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.
[22] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 599–619.
[23] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[25] G. E. Hinton, T. J. Sejnowski, and D. H. Ackley, "Boltzmann machines: Constraint satisfaction networks that learn," Dept. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMS-CS-84-119, 1984.
[26] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, no. 2. Boston, MA, USA: MIT Press, 1986, pp. 282–317.
[27] J.-B. Hoock, C.-S. Lee, A. Rimmel, F. Teytaud, M.-H. Wang, and O. Teytaud, "Intelligent agents for the game of go," *IEEE Comput. Intell. Mag.*, vol. 5, no. 4, pp. 28–42, Nov. 2010.
[28] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, 1982.
[29] M. Hue, M. Riffle, J.-P. Vert, and W. S. Noble, "Large-scale prediction of protein-protein interactions from structures," *BMC Bioinf.*, vol. 11, no. 1, p. 144, 2010.
[30] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo planning," in *Proc. Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, 2006, pp. 282–293.
[31] N. Kohl and R. Miikkulainen, "Evolving neural networks for strategic decision-making problems," *Neural Netw.*, vol. 22, no. 3, pp. 326–337, 2009.
[32] W. A. Little, "The existence of persistent states in the brain," in *From High-Temperature Superconductivity to Microminiature Refrigeration*. Boston, MA, USA: Springer, 1974, pp. 145–164.
[33] S. Liu, L. Ying, and S. Shakkottai, "Influence maximization in social networks: An ising-model-based approach," in *Proc. 48th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2010, pp. 570–576.
[34] H. Matsuda, "The ising model for population biology," *Prog. Theor. Phys.*, vol. 66, no. 3, pp. 1078–1080, 1981.
[35] B. M. McCoy and T. T. Wu, *The Two-Dimensional Ising Model*. North Chelmsford, MA, USA: Courier Corporation, 2014.
[36] M. Müller, "Computer go," *Artif. Intell.*, vol. 134, nos. 1–2, pp. 145–179, 2002.
[37] J. Peter, R. Freyer, M. F. Smith, C. Scarfone, R. E. Coleman, and R. J. Jaszczak, "Nuclear medicine image segmentation using a connective network," *IEEE Trans. Nucl. Sci.*, vol. 44, no. 4, pp. 1583–1590, Aug. 1997.
[38] L. Ralaivola, L. Wu, and P. Baldi, "SVM and pattern-enriched common fate graphs for the game of go," in *Proc. ESANN*, 2005, pp. 27–29.
[39] J. Raymond and K. M. Wong, "Next nearest neighbour ising models on random graphs," *J. Stat. Mech., Theory Exp.*, vol. 2012, no. 9, 2012, Art. no. P09007.
[40] H. Remus, "Simulation of a learning machine for playing go," in *Proc. IFIP Congr.*, 1962, pp. 428–432.
[41] M. Sato, K. Anada, and M. Tsutsumi, "A mathematical formulation based on the connectedness of stones for the game of go," *Int. J. Comput. Inf. Sci.*, vol. 16, no. 4, pp. 1–10, 2015.
[42] M. Sato, K. Anada, and M. Tsutsumi, "A model for the connectedness of stones in the game of go," in *Proc. IEEE/ACIS 16th Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, Jun. 2015, pp. 1–6.
[43] M. Sato, K. Anada, and M. Tsutsumi, "A mathematical formulation to efficiently determine 'life status' in the game of go," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Mar. 2016, pp. 1684–1689.
[44] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.
[45] D. Silver *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.
[46] D. Silver, R. S. Sutton, and M. Müller, "Temporal-difference search in computer go," *Mach. Learn.*, vol. 87, no. 2, pp. 183–219, 2012.
[47] D. Sornette, "Stock market speculation: Spontaneous symmetry breaking of economic valuation," *Phys. A, Stat. Mech. Appl.*, vol. 284, nos. 1–4, pp. 355–375, 2000.
[48] D. Stauffer, "Social applications of two-dimensional ising models," *Amer. J. Phys.*, vol. 76, no. 4, pp. 470–473, Aug. 2008.
[49] D. H. Stern, T. Graepel, and D. MacKay, "Modelling uncertainty in the game of go," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1353–1360.
[50] S. Takagawa, *How to Play Go*. Tokyo, Japan: Nihon Ki-in Japanese Go Association, 1974.
[51] G. Tkacik, E. Schneidman, M. J. Berry, II, and W. Bialek. (2006). "Ising models for networks of real neurons." [Online]. Available: https://arxiv.org/abs/q-bio/0611072
[52] M.-Y. Tsai, J.-M. Yuan, and S.-H. Lin, "Thermodynamic insight into protein aggregation using a kinetic ising model," *J. Chin. Chem. Soc.*, vol. 62, no. 1, pp. 21–25, 2015.
[53] E. C. van der Werf, H. J. van den Herik, and J. W. Uiterwijk, "Learning to estimate potential territory in the game of go," in *Proc. Int. Conf. Comput. Games*. Berlin, Germany: Springer, 2004, pp. 81–96.
[54] O. Wolkenhauer *et al.*, "Sysbiomed report: Advancing systems biology for medical applications," *IET Syst. Biol.*, vol. 3, no. 3, pp. 131–136, 2009.
[55] L. Wu and P. Baldi, "Learning to play go using recursive neural networks," *Neural Netw.*, vol. 21, no. 9, pp. 1392–1400, 2008.
[56] L. Wu and P. F. Baldi, "A scalable machine learning approach to go," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1521–1528.
[57] A. Yee and M. Alvarado, "Pattern recognition and Monte-Carlo tree search for go gaming better automation," in *Proc. Ibero-Amer. Conf. Artif. Intell.* Berlin, Germany: Springer, 2012, pp. 11–20.
[58] A. Yee and M. Alvarado, "Well-time pattern recognition in go gaming automation," in *Mathematical Methods in Science and Engineering*. Montclair, NJ, USA: Institue for Natural Sciences and Egineering, 2014, pp. 174–181.
[59] A. Yee and M. Alvarado, "Patterns of go gaming by ising model," in *Proc. Mexican Conf. Pattern Recognit.* Cham, Switzerland: Springer, 2018, pp. 3–11.

ALFONSO ROJAS-DOMÍNGUEZ received the M.Sc. degree in intelligence engineering and the Ph.D. degree in electronics engineering from the University of Liverpool, U.K., in 2003 and 2007, respectively. Since 2014, he has been a Research Fellow with the National Council of Science and Technology of Mexico, and is a member of the National Researchers System, Mexico. His main research interests include machine learning, deep learning, computational intelligence, computational optimization, and artificial vision.

DIDIER BARRADAS-BAUSTISTA received the Ph.D. degree from Barcelona University, in 2017. He has been working in the areas of bioinformatics, protein modeling, and mathematical modeling of protein interactions using machine learning and high-performance computing. He is currently a Postdoctoral Fellow with the King Abdullah University of Science and Technology.

MATÍAS ALVARADO received the bachelor's degree in mathematics from the National Autonomous University of Mexico (UNAM), and the Ph.D. degree from the Technical University of Catalonia, both in mathematics. He is currently a Research Scientist with the Department of Computer Science, CINVESTAV-IPN. His ongoing research is focused on the formal modeling and computer simulation of strategic reasoning for sport team games through Nash equilibrium, as well as, modeling the board game of Go under the Ising model with the aim to apply this knowledge to the modeling of cancer metastasis. Previously, he has worked on Artificial Intelligence for vehicles speed adaptation, knowledge management, and epistemic logics.