



A review on generative Boltzmann networks applied to dynamic systems

Rémi Souriau, Jean Lerbet, Hsin Chen, Vincent Vigneron

► To cite this version:

Rémi Souriau, Jean Lerbet, Hsin Chen, Vincent Vigneron. A review on generative Boltzmann networks applied to dynamic systems. *Mechanical Systems and Signal Processing*, 2021, 147, pp.107072. 10.1016/j.ymssp.2020.107072 . hal-02901415

HAL Id: hal-02901415

<https://hal.science/hal-02901415>

Submitted on 18 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

A Review on Generative Boltzmann Networks applied to Dynamic Systems

Mr Rémi Souriau^{a,*} (PhD student), Pr Jean Lerbet^b (Researcher), Pr Hsin Chen^c (Researcher) and Pr Vincent Vigneron^{a,**} (Researcher)

^aIBISC, Univ Evry, Université Paris-Saclay, 91025, Evry, France

^bLaMME, Univ Evry, Université Paris-Saclay, 91025, Evry, France

^cNational Tsing Hua University, Electrical Engineering Department, 300, Hsinchu, Taiwan

ARTICLE INFO

Keywords:

Artificial Neural Network
Dynamic System Modelling
Boltzmann Machines
Diffusion Network
Stochastic Differential Equation

ABSTRACT

The modelling of dynamic system is a challenging problem in a large number of applications like prediction, bio-data modelling, computer vision or time-series processing. To face the complexity and the non-linearity of data, new models are regularly proposed through the literature. Among proposed models artificial neural network (ANN) have benefit of a large interest in the scientist community. The use of latent variables to extract and diffuse complex features in multi-layer feedforward neural networks provide usually excellent results. In 1982, Hopfield proposes a generative and deterministic neural network to model a physical system. His work leads to the emergence of a large number of generative neural networks: Boltzmann Machine and its extensions. Different applications lead researchers to propose new extensions for the Boltzmann machine to handle dynamic systems, continuous variables or systems with complex features. In parallel, a new model named the Diffusion Network has emerged, also inspired from Hopfield network but with continuous stochastic properties and designed to solve stochastic differential equations. This paper has the objective to review the evolution of the Boltzmann Machine's family with a synthetic and historical vision and their development for dynamic problem. **THE FOLLOWING TEXT (IN YELLOW) WILL BE REMOVED IN THE FINAL VERSION** To write this review, we selected articles from journals/conferences and review articles ($\frac{1}{3}$ are < 7 years) quoted in meta sources (Scopus and Web-of-Sciences). Once a clearly research question was asked – *How generative networks model dynamic systems ?* – we defined our search terms for papers. Note that not all extensions to Boltzmann machines are presented in this paper. Only models related with dynamic applications and most salient models were retained.

Glossary

ANN	Artificial Neural Networks	BEAM	Boltzmann Encoded Adversarial Machine
MLP	Multilayer Perceptron	DBN	Deep Belief Network
GNN	Generative Neural Network	DBM	Deep Boltzmann Machine
GBN	Generative Boltzmann Network	conv-RBM	convolutional Restricted Boltzmann Machine
RNN	Recurrent Neural Network	cDBN	convolutional Deep Belief Network
LSTM	Long-Short Term Memory	GBRBM	Gaussian-Bernoulli Restricted Boltzmann Machine
HN	Hopfield Network	covRBM	covariance Restricted Boltzmann Machine
bHN	binary Hopfield Network	mcRBM	mean and covariance Restricted Boltzmann Machine
cHN	continuous Hopfield Network	ssRBM	spike and slab Restricted Boltzmann Machine
BM	Boltzmann Machine	CssCDBM	Contrastive spike and slab Convolutional Deep Boltzmann Machine
RBM	Restricted Boltzmann Machine	mPoT	Mean Product of Student t -distributions
FRBM	Fuzzy Restricted Boltzmann Machine	PoT	Product of Student t -distributions
WRBM	Wasserstein Restricted Boltzmann Machine		

*This work was partly supported by the doctoral school STIC of Paris-Saclay and the Taiwan Ministry Of Science and Technology (MOST).

 remi.souriau@ibisc.univ-evry.fr (R. Souriau); vincent.vigneron@ibisc.univ-evry.fr (V. Vigneron)

ORCID(s): 0000-0003-0849-9515 (R. Souriau); 0000-0001-5917-6041 (V. Vigneron)

DRBM	Discriminative Restricted Boltzmann Machine	VAR	vector autoregressive
HDRBM	Hybrid Discriminative Restricted Boltzmann Machine	KF	Kalman Filter
CRBM	conditional Restricted Boltzmann Machine	HMM	Hidden Markov Model
TRBM	Temporal Restricted Boltzmann Machine	MIMO	Multiple Input Multiple Output
RTRBM	Recurrent Temporal Restricted Boltzmann Machine	ELM	Extrem Learning Machine
SRTRBM	Structured Recurrent Temporal Restricted Boltzmann Machines	MCD	Minimization of the Contrastive Divergence
RNN-RBM	Recurrent Neural Network - Restricted Boltzmann Machine	CD	Contrastive Divergence
GRBM	Gated Restricted Boltzmann Machine	KL	Kullback-Leibler
DP	Diffusion Processes	SDE	Stochastic Differential Equation
DN	Diffusion Network	MCEM	Monte Carlo Expectation Maximization
DN-RBM	continuous Restricted Boltzmann Machine	MCMC	Markov Chain Monte Carlo
		EM	Expectation Maximization
		ReLU	Rectified Linear Unit
		AIC	Akaike Information Criterion

Introduction

Recent years have witnessed the emergence of large amount of data and technology progresses on computational power. Machine learning advances are paving the way for new possibilities to model complex systems in many applications but may fail because of their limitations, among which *e.g.* the curse of dimensionality [20] or data dependencies modeling. Neural networks are non-linear, graphical models recognized as particularly effective for processing large data in most domains. These models are based on the presence of latent variables to extract hidden informations. Neural Network can be grouped in two wide families: (i) discriminative neural networks that model the conditional probability $\Pr(\mathbf{y}|\mathbf{x})$ between the output vector $\mathbf{y}$ and the input $\mathbf{x}$. The feedforward neural network (or multilayer perceptron) [21] where information is propagated from $\mathbf{x}$ to an $\mathbf{y}$ (ii) generative models that estimate the joint probability distribution $\Pr(\mathbf{x}, \mathbf{y})$. This article is dedicated to the second family of models.

Neurons in a generative neural network are updated iteratively in function of the previous state of the network. Works of Hopfield in the 80's on deterministic neural network [34, 35, 36] leads to the emergence of the very popular Boltzmann Machine proposed by Hinton, which is the stochastic extension of the binary Hopfield Network. The keys of the success of Boltzmann Machines lie in the flexibility of the model and its adaptability to solve problems such as modelling [93], classification [107, 48] or prediction [46, 2]. The Boltzmann Machine has been used in many applications like image processing [80], sound processing [65], bag of word [31] or with several inputs of different natures such as in [90, 18].

In this article, we will review the main features and limitations of such models that all belong to the family of generative Boltzmann networks. One can already find a few reviews on Restricted Boltzmann Machine (RBM) in the literature (see *e.g.* [108, 96]) but none with a synthetic and historical vision. The paper is organized as below. First, the background of generative Boltzmann networks is introduced. Second and third sections present the main features of the Boltzmann machines, the learning procedures and its extensions. Finally, the Diffusion network is detailed in the last section as well as the continuous Restricted Boltzmann machine based on Diffusion network. Fig.1 summarizes the hierarchy of models presented in the paper. Each cell is a model presented in this review. Directed links between models indicate restricted / extension of the new model from the previous one. This graph starts from the top with Hopfield Networks and propagate to the down with more recent models. The RBM is very popular in the neural network community. This popularity is visible in this graph with the many extensions of RBM. The Diffusion Network (DN) did not get the same attention as the RBM but the DN can be seen as a generalization of Boltzmann Machine (BM) and Hopfield networks in different aspects. For the signification of acronyms, refer to the glossary.

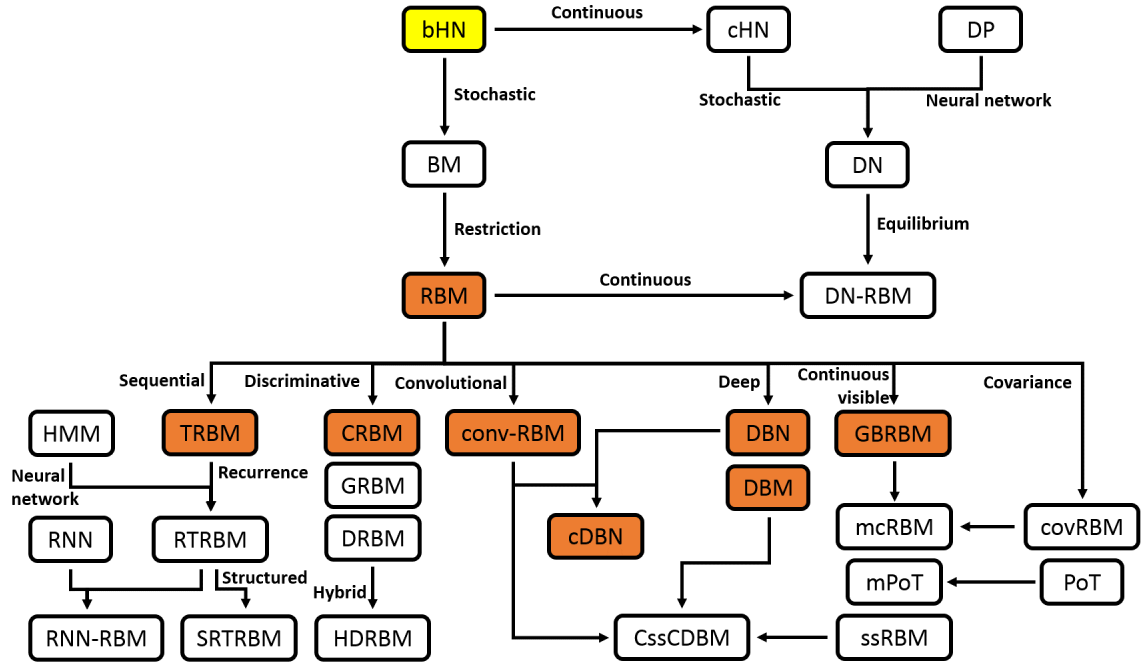


Figure 1: Hierarchy of Generative Boltzmann Network (GBN) from the review. Each cell represents a model and directed links give the kinship relating one model to an extension. The tree starts from the yellow cell with the binary Hopfield Network (bHN) proposed in 1982. Orange cells are models that have impacted significantly the research onto this family. The RBM is the original model. The Gaussian-Bernoulli Restricted Boltzmann Machine (GBRBM) is used in almost all applicative papers because of the continuous behavior of the visible units. The Deep Belief Network (DBN) and the Deep Boltzmann Machine (DBM) are two deep versions of the RBM. The other orange cell models presented in the paper have impacted the research on RBM's extensions.

Methodology

The idea beyond this paper is to build a taxonomy of the Boltzmann machine family to better understand these models and their properties. How Boltzmann machines has been developed? How these models are connected? Which models are the most influent in research on Boltzmann machines?

These models are generative. They address applications for which data are unlabeled. The study focuses on dynamic systems and aims to explain how these models can capture temporal information. Some reference journals (books eventually) on artificial neural networks for dynamic systems are at the basis of this review, *e.g.* [21, 46]. The search was further extended to models cited from main references: original articles for each models and their extension from several publisher databases such as ELVESIER, SCOPUS or GOOGLE SCHOLAR. Using the model name as a keyword, we found many articles that presented models and other review articles ([96, 108] for example). Finally, a research on the different database has been proceed to find recent applications (<5 years) using Boltzmann machines. Seminal articles on Boltzmann machines are around 40 years old, but the articles presenting current Boltzmann models may go back up to 20 years ago to today. In addition, a link is established between the diffusion network and the Boltzmann machines. This model in particular has very few references but the properties of the model make it very promising in the field of dynamic systems.

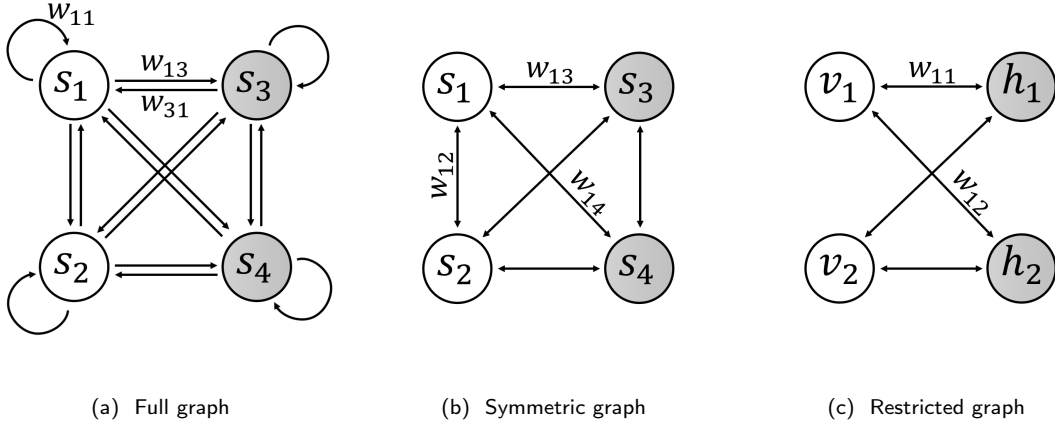


Figure 2: Graphs with 4 neurons. White neurons are visible neurons and gray neurons are hidden neurons. In 2a and 2b, double direction arrow means $W_{ij} = W_{ji}$, else $W_{ij} \neq W_{ji}$. In 2c, the coefficient W_{ij} refers to the weight of the symmetric link between the i -th visible unit v_i and the j -th hidden unit h_j .

1. Background

1.1. Generative Boltzmann Network

An Artificial Neural Networks (ANN) is a bioinspired graphical model where the processing units (neurons) are organized and connected each other. We have now a rather good understanding of the functioning of simple neural networks which consist of two sort of objects: the processing units which generalizes the original McCulloch-Pitts model by transforming the output of a unit as a continuous value, and the weighted connections between these units. The formers make simple computation (summation, thresholding), the latters feed the further units with input values for the computations (see for a review Hertz *et al.* [24]). For a given learning task, building a network requires to choose the network *topology*, *i.e.* the number of inputs, number of layers, connectivity, activation function, etc.

The main strength of ANN is the presence of latent (or hidden) units which synthesizes data by extracting useful features for a given task. The latent variables have become a popular concept in machine learning in models such as Kalman Filter (KF) [41] or the Hidden Markov Model (HMM) [79, 15], but contrary to ANN latent variables in a KF and a HMM have a meaning that can be interpreted.

In a GBN, the state of each neurons is updated according to the antecedent state of the network until the global state of the network becomes stable. This state is called the *equilibrium* state. A full graph of a GBN is given in Fig.2a. The coefficient W_{ij} refers to the weight of the link from the neuron i to the neuron j . In a network with n neurons, we call $W = (W_{ij})_{1 \leq i, j \leq n}$ the transfer matrix between all the neurons of the network and $\xi = (\xi_i)_{1 \leq i \leq n}^T$ the bias vector (not represented in Fig.2). We also note $s(t) = (s_1(t), \dots, s_n(t))^T$ the state vector at time t .

1.2. The father BM: Hopfield Network

In 1982 and 1984, J.J. Hopfield presents two deterministic GBNs: the bHN [34] and the continuous Hopfield Network (cHN) [35]. His work on deterministic networks later inspired many research works on probabilistic networks. Both bHN and cHN have symmetric connection between neurons ($W_{ij} = W_{ji}$) and no feedback link ($W_{ii} = 0$). Fig.2b gives the typic graph of a Hopfield Network (HN) and Fig.3 gives the two neuron structures of bHN and cHN. HN has been used in different fields of study like image processing for noise reduction in [75] or more recently in [52] for super-resolution images or in economics in [76].

The bHN employs binary neurons ($s_j = +1$ or -1). The weighted inputs and the bias pass into the activation function called the *sign* function to provide a binary output.

$$s_j = \text{sgn} \left(\xi_j + \sum_{i=1}^n W_{ij} s_i \right). \quad (1)$$

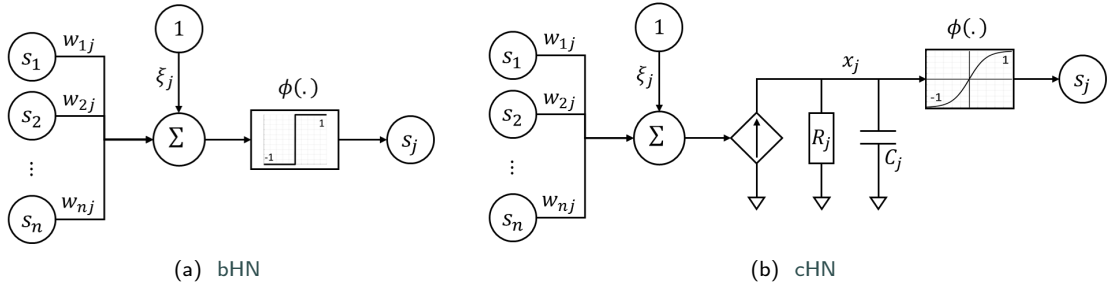


Figure 3: HNs' neuron structure. $\phi(\cdot)$ is the activation function: the *sign* function in **3a** and the *sigmoid* function in **3b**. In **3b** R_j and C_j are, respectively, the resistor and capacitor of neuron j . There is no reason that R_j and C_j are the same for all neurons. See sect.4 for more details.

The energy function of the bHN is:

$$E_{bHN}(s) = - \sum_{\substack{1 \leq i, j \leq n \\ i \neq j}} W_{ij} s_i s_j - \sum_{i=1}^n \xi_i s_i. \quad (2)$$

The main limitation of the binary behavior of these neurons is the capacity of data representation. For instance with n neurons, a bHN can encode 2^n different states. Continuous neurons increase significantly this capacity of representation with only few neurons. In addition, data are barely binary. The benefit of transitioning from a binary network to a continuous network is then double.

In 1984, Hopfield proposed as a generalization of bHN the cHN to get closer to the biological neuron behavior. He then proposed a new architecture for working with continuous neurons varying continuously over time. The structure of the neuron j is given in Fig.3b. The activation function becomes a sigmoid function. The state space of the network becomes a n -dimensional hypercube bounded with the variation range of the activation function. The weighted input is converted into a current and passes an electronic RC filter. x_j is the input voltage of $\phi(\cdot)$. The variation of $x_j(t)$ (see Fig.3b) is given by the differential equation:

$$C_j \frac{dx_j(t)}{dt} = -\frac{x_j(t)}{R_j} + \xi_j + \sum_{i=1}^n W_{ij} \phi(x_i(t)). \quad (3)$$

The state of the cHN tends to an *equilibrium* state characterized by the energy function:

$$E_{cHN}(s) = - \sum_{\substack{1 \leq i, j \leq n \\ i \neq j}} W_{ij} s_i s_j - \sum_{i=1}^n \xi_i s_i + \sum_{i=1}^n \frac{1}{R_i} \int_0^{s_i} \phi^{-1}(s') ds'. \quad (4)$$

The energy function in Eq.4 introduced by Hopfield is a Lyapunov function. E_{cHN} is the sum of E_{bHN} in Eq.2 and an integral term.

The deterministic nature of artificial neurons limits their functioning. First, the evolution of the state s tends to decrease the energy, the equilibrium state corresponding to the lowest possible energy reached. As many local minimal are present in the energy landscape, the state of the network converges to a local minimal. The second limitation is the risk of overfitting during the training step.

In the next section, we introduce stochastic Generative Neural Network (GNN)s and see how those models overcome the previous limitations.

2. Boltzmann Machines

2.1. Description

Introduced by Fahlman and Hinton in 1983 [17], the BM is the stochastic extension of the bHN. It first questions the use of parallel architecture for machine learning. Each neuron can be seen as a Bernoulli random variable. Like in

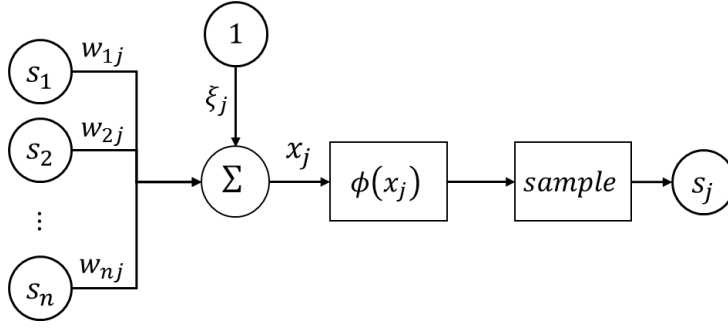


Figure 4: Structure of the neuron j of a BM. $\phi(x_j) = \Pr(s_j = 1|\mathbf{s})$

the bHN, the BM is composed of visible and hidden units. The neuron structure of a BM is depicted in Fig.4 where s_j is the state of the neuron j ('0' or '1').

For each neuron, the result of the linear combination of the inputs **pass into** an activation function $\phi(\cdot)$ such as the sigmoid function:

$$\phi(x) = \frac{1}{1 + \exp(-x)}. \quad (5)$$

As this function is bounded between 0 and 1, the result of $\phi(x_j)$ can be interpreted as the probability $\Pr(s_j = 1|\mathbf{s})$, *i.e.* the probability that $s_j = 1$ given the state of the network. The 'sample' step in Fig.4 consists in sampling an uniform random variable u_j between 0 and 1 and the state s_j depending on the inequality:

$$s_j = \begin{cases} 1 & \text{if } u_j < \Pr(s_j = 1|\mathbf{s}), \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

The sampling step in Eq.6 is at the origin of the stochastic behavior of the network. At a given time, even if the expected state of a neuron is fixed to '0' or '1' by the state of the network itself, there is always a possibility that the state of the neuron will change. The probability of converging to a local minimal is then reduced.

The BM is an energy based model. The energy of a BM is similar to the energy function of the bHN (see Eq.2): $E_{BM}(\mathbf{s}) = E_{cHN}(\mathbf{s})$. The associated joint probability $P_{BM}(\mathbf{s})$ is defined as:

$$P_{BM}(\mathbf{s}) = \frac{1}{Z} \exp(-E_{BM}(\mathbf{s})), \quad (7)$$

where Z is a marginalization constant so that P_{BM} is a probability distribution function, *i.e.*

$$Z = \sum_{\mathbf{s} \in \llbracket 0,1 \rrbracket^n} \exp(-E_{BM}(\mathbf{s})) \quad (8)$$

Z is also called the *partition function*.

2.2. Restricted Boltzmann Machine

Fig.2b represents the architecture of a BM, each neurons being updated independently. For a large network, update neurons sequentially can be time consuming. Smolensky proposed in 1986 the Harmonium model [88], known, today, as the RBM. The graph structure of the RBM in Fig.2c is a bi-part BM with two layers: the visible layer $\mathbf{v} \in \llbracket 0, 1 \rrbracket^n$ and the hidden layer $\mathbf{h} \in \llbracket 0, 1 \rrbracket^m$. Neurons from the same layer are not connected each other. For a given state of one layer, neurons from the second layer are conditionally independent each other and can be update at once. Removing links between neurons reduces the complexity of the model. To balance this restriction on the network structure, neurons or layers can be added (see section 3.1). We note $W \in \mathbb{R}^{(n \times m)}$ the *transfer* matrix between the two layers and ξ^v and ξ^h

the bias vectors of, respectively, the visible layer and the hidden layer. The energy expression of a RBM and of a BM are the same but thanks to the absence of intra-layer links, the energy function of the BM can be simplified into Eq.9:

$$E_{RBM}(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{v}^T \boldsymbol{\xi}^v - \mathbf{h}^T \boldsymbol{\xi}^h. \quad (9)$$

Like for BM, we define the joint probability distribution of the RBM and the marginalized probability distribution over visible units as:

$$P_{RBM}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E_{RBM}(\mathbf{v}, \mathbf{h})), \quad (10)$$

where Z is the normalization constant $Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E_{RBM}(\mathbf{v}, \mathbf{h}))$. Hidden units are not accessible. Compute the energy function in Eq.9 requires to sample the hidden units. The free energy $F_{RBM}(\mathbf{v})$ is an energy function where hidden state are marginalized [58]. The *free energy* is defined as:

$$P_{RBM}(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp(-E_{RBM}(\mathbf{v}, \mathbf{h})) = \frac{1}{Z} \exp(-F_{RBM}(\mathbf{v})), \quad (11)$$

where

$$F_{RBM}(\mathbf{v}) = -\mathbf{v}^T \boldsymbol{\xi}^v - \sum_{j=1}^m \log \left(1 + \exp(\mathbf{v}^T \mathbf{W}(:, j) + \xi_j^h) \right), \quad (12)$$

and with $\mathbf{W}(:, j)$ is the j th column vector of $\mathbf{W}$. Training a RBM can be performed with the energy function in Eq.9 or the free energy function in Eq.12. In the first case, hidden neurons are involved into the learning algorithm. In the second case, hidden neurons are marginalized. Different learning procedures of BMs has been explored in the literature.

2.3. Learning procedures for Boltzmann Machines

BM are energy models. A training set $\mathcal{D} = \{\mathbf{v}^k\}_{1 \leq k \leq N}$ composed of N observations is used to find the best set of parameters $\mathcal{P} = \{\mathbf{W}, \boldsymbol{\xi}\}$, $\boldsymbol{\xi}$ regrouping visible and hidden bias vectors. We also note $P^0(\mathbf{v})$ the probability distribution over the training set, $P^\infty(\mathbf{v})$ the probability distribution corresponding to the equilibrium state and $P^q(\mathbf{v})$ the probability distribution after q steps of reconstruction (a reconstruction step corresponding to a *two-way sampling* between the visible and the hidden layers: $\mathbf{h}$ sampled from $\mathbf{v}$ and $\mathbf{v}$ sampled from $\mathbf{h}$). Finally, we denote $\mathbf{s} = (\mathbf{v}, \mathbf{h})^T$ the set of all units of the model.

Different methods to train BMs have been proposed in the literature. In this paper, minimization of the Contrastive Divergence (CD) presented in [25, 3] is considered as well as the limits of this approach and of other methods.

2.3.1. Training Boltzmann Machines using the Contrastive Divergence

An intuitive idea to train the BM consists to maximize the joint log-likelihood:

$$\mathcal{L}^{ML}(\mathcal{P}|\mathcal{D}) = \sum_{k=1}^N \log P_{RBM}(\mathbf{v}^k). \quad (13)$$

Working directly on Eq.13 is difficult due to the presence of the constant Z . The minimization of the Kullback-Leibler (KL) divergence G [44] between $P^0(\mathbf{v})$ and $P^\infty(\mathbf{v})$ is equivalent to the maximization of the log-likelihood and bypass the need to calculate Z . The derivation of G with respect to the parameters leads to the following update rule for each the parameters λ_i [32]:

$$\frac{\partial G}{\partial \lambda_i} = \left\langle \frac{\partial E_{BM}(\mathbf{s})}{\partial \lambda_i} \right\rangle_0 - \left\langle \frac{\partial E_{BM}(\mathbf{s})}{\partial \lambda_i} \right\rangle_\infty, \quad (14)$$

where $\langle \cdot \rangle_0$ is the expectation over the training set at the initial time and $\langle \cdot \rangle_\infty$ is the expectation value at the equilibrium state. Updating the parameters requires to compute at each iteration the equilibrium distribution which means

a large number of updates. To reduce the training cost, the minimization of the KL-divergence is replaced by the Minimization of the Contrastive Divergence (MCD) [25] which consists to minimize the contrast D between two KL-divergences:

$$D = KL(P^0(\mathbf{v}), P^\infty(\mathbf{v})) - KL(P^q(\mathbf{v}), P^\infty(\mathbf{v})). \quad (15)$$

The only case where D is zero is the case when $P^0(\mathbf{v}) = P^q(\mathbf{v})$, *i.e.* when after q steps of reconstruction the probability distribution does not change, then the RBM is already stable and $P^0(\mathbf{v}) = P^\infty(\mathbf{v})$. Update rules become:

$$\frac{\partial D}{\partial \lambda_i} = \left\langle \frac{\partial E_{BM}(s)}{\partial \lambda_i} \right\rangle_0 - \left\langle \frac{\partial E_{BM}(s)}{\partial \lambda_i} \right\rangle_q, \quad (16)$$

with $\langle \cdot \rangle_q$ the expectation after q reconstruction steps. In the case of RBMs with binary values, note that $\langle s_i \rangle_q$ refers to the probability of $s_i = 1$ after q iterations (even, if, in practice, $q = 1$). **Markov Chain Monte Carlo (MCMC) method is used with the sampling rule (Eqs.5-6), to estimate $\langle s_i \rangle_q$.** The update equations for the transfer matrix coefficient W_{ij} and the bias ξ_i at each iteration can be deduced from Eq.16:

$$\begin{cases} \Delta W_{ij} \sim \langle s_i s_j \rangle_0 - \langle s_i s_j \rangle_1, \\ \Delta \xi_i \sim \langle s_i \rangle_0 - \langle s_i \rangle_1. \end{cases} \quad (17)$$

To conclude, minimizing the KL-divergence can be replaced by the MCD rule to speed up the training. Note that if we use the energy function with explicit values of neurons like in Eq.17, the learning algorithm requires to sample the hidden units to update the parameters of the model.

Carreira-Perpinan [5], MacKay [57], and Yuille [106] studied properties of the CD and demonstrated MCD rule converges to the optimal solution with a small bias.

2.3.2. Training based on free energy

To avoid the approximation of the CD by sampling hidden units, different training methods using the free energy (Eq.12) have been proposed to get a better approximation than the log-likelihood in Eq.13. The main advantage is the free energy does not require to be minimized with respect to the hidden variables. Marlin *et al.* in [58] review various scores based on free energy to train BMs.

Younes [105] and Tieleman [95] proposed to simulate the Markov chain with only one step to estimate the probability distribution $P_{BM}(s)$ and to update the parameters with a small learning rate to maximize $\mathcal{L}^{ML}(\mathcal{P}|\mathcal{D})$ given Eq.13. The Contrastive Divergence [25] can also be used to learn the parameters of a BM by replacing the energy function in Eq.16 with the free energy function. The Ratio Matching proposed by Hyvärinen [40] consists to minimize the Euclidean distance between the conditional probability of each component of $v_i^k = 1$ and $v_i^k = 0$ given $\mathbf{v}^{k \setminus i}$: the visible layer without the i -th component.

$$\mathcal{L}^{RM}(\mathcal{P}|\mathcal{D}) = \sum_{k=1}^N \sum_{i=1}^n \sum_{\epsilon \in \{0,1\}} P^\infty(\mathbf{v}^k) (P^0(v_i^k = \epsilon | \mathbf{v}^{k \setminus i}) - P^\infty(v_i^k = \epsilon | \mathbf{v}^{k \setminus i}))^2 \quad (18)$$

The Generalized Score Matching proposed by Lyu in [56] and improved by Marlin in [58] consists in maximizing between the inverse of the conditional probabilities of a visible unit v_i^k with the other $\mathbf{v}^{k \setminus i}$ known over the training set. The function to maximize is given by:

$$\mathcal{L}^{GSM}(\mathcal{P}|\mathcal{D}) = \sum_{k=1}^N \sum_{i=1}^n P^\infty(\mathbf{v}^k) \left(\frac{1}{P^0(v_i^k | \mathbf{v}^{k \setminus i})} - \frac{1}{P^\infty(v_i^k | \mathbf{v}^{k \setminus i})} \right)^2. \quad (19)$$

Marlin [58] concludes in his paper on the *inductive principles* for RBM that the stochastic maximum likelihood and the CD are well suited in many situations (applications, computation time, ...).

Research on training algorithm for RBM remains active today. We can cite for instance Montavon's paper [66] where the Wasserstein distance [97] is used instead of the KL divergence to train Wasserstein Restricted Boltzmann Machine (WRBM). Kuleshov proposed to use variational inference techniques to train RBM [43]. Finally, Fisher introduced in [19] the Boltzmann Encoded Adversarial Machine (BEAM) which consists in adding an adversarial term in the cost function.

3. Extensions of Boltzmann Machines

One of the major strength of the RBM lies in its flexible architecture which makes it suitable for many different applications. There exists a lot of versions of the RBM, for instance some papers propose to modify the original BM with a Rectified Linear Unit (ReLU) activation function [69] or to fuzzified the parameters of the model as in the Fuzzy Restricted Boltzmann Machine (FRBM) [6]. In this section we focus on some well-known extensions of the RBM in the neural network community.

3.1. Multi-layers Boltzmann Machines

In practice, RBM are preferred to fully connected BM to model complex systems because of RBM update rule. However, the absence of inter-layer links in the RBM reduces the capacity of the model to capture complex features. A simple solution to overcome this limit is to add additional hidden layers to capture high-order information. Two RBMs with multiple hidden layers have been proposed in the literature: the DBN and the DBM.

3.1.1. Deep Belief Network

In Multilayer Perceptron (MLP)s, the information is propagated from the input layer to the output layer and the error of predicted output is back-propagated [23] to correct parameters of the network by minimizing the cross-entropy error. Back-propagation algorithm proposed by Sejnowski and Lecun [49] has however some well-known limitations: the training requires labeled data, the learning can be very slow and the network may converge into a local minima instead of the desired global minimum. The DBN introduced in 2006 by Hinton [29, 26] has been proposed to pretrain deep neural network. The DBN can be seen as stacked RBMs (see Fig.5a), each RBMs being trained independently. We note $\mathbf{h}^{(i)}$ the i -th hidden layer. The first step consists in training of the RBM between the visible and the first hidden layer $\mathbf{h}^{(1)}$. The observations from the training set are sampled into the first hidden layer and will constitute a new training set for the next RBM. In Fig.5a for instance, once the RBM between $(\mathbf{v}, \mathbf{h}^{(1)})$ is trained, the second RBM $(\mathbf{h}^{(1)}, \mathbf{h}^{(2)})$ is trained but $\mathbf{h}^{(1)}$ is updated by ignoring $\mathbf{v}$ and is computed only with respect to $\mathbf{h}^{(2)}$. A large number of applications used for DBN or its extension *e.g.* for face recognition [38], for audio classification task [51], for machine health monitoring systems [109, 42], for schizophrenia prediction [78], for detecting faults in axial coupling systems [100] or for time series forecasting [45].

3.1.2. Deep Boltzmann Machines

Unlike DBN, the DBM [83] is an undirected graph. In the Fig.5b, $\mathbf{v}$ and $\mathbf{h}^{(2)}$ are updated in function of $\mathbf{h}^{(1)}$. $\mathbf{v}$ and $\mathbf{h}^{(2)}$ allow to update $\mathbf{h}^{(1)}$ according to Eq.20.

$$\begin{cases} \Pr(\mathbf{v} = 1 | \mathbf{h}^{(1)}) = \phi(W^{(1)}\mathbf{h}^{(1)} + \xi^{\mathbf{v}}), \\ \Pr(\mathbf{h}^{(2)} = 1 | \mathbf{h}^{(1)}) = \phi(W^{(2)T}\mathbf{h}^{(1)} + \xi^{\mathbf{h}^{(2)}}), \\ \Pr(\mathbf{h}^{(1)} = 1 | \mathbf{v}, \mathbf{h}^{(2)}) = \phi(W^{(1)T}\mathbf{v} + W^{(2)}\mathbf{h}^{(2)} + \xi^{\mathbf{h}^{(1)}}). \end{cases} \quad (20)$$

The structure of the DBM in Fig.5b can be seen as a RBM whose visible and second hidden layers are concatenated into one layer. The visible and hidden layers $\mathbf{h}^{(i)}$ with i an even number are updated at once and all hidden layer $\mathbf{h}^{(j)}$ with j an odd number are updated at once. The feedback information from deepest layers when updating lower layers allow the DBM to be more robust than the DBN [85]. For a given visible layer, the *mean field inference* allows to update hidden layers [84, sect. 4.2]. This method consists in estimating the probability of activation of each hidden neuron given the visible layer only. In the case of two hidden layers $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$, the mean field inference allows us to compute the probability of $\mathbf{h}^{(2)}$ given the visible layer and to update $\mathbf{h}^{(1)}$ according to $\mathbf{v}$ and $\Pr(\mathbf{h}^{(2)} | \mathbf{v})$ (see Fig.6a). The DBM training algorithm presented in [21, Chap. 20] for a 3-layer model requires to apply mean field inference

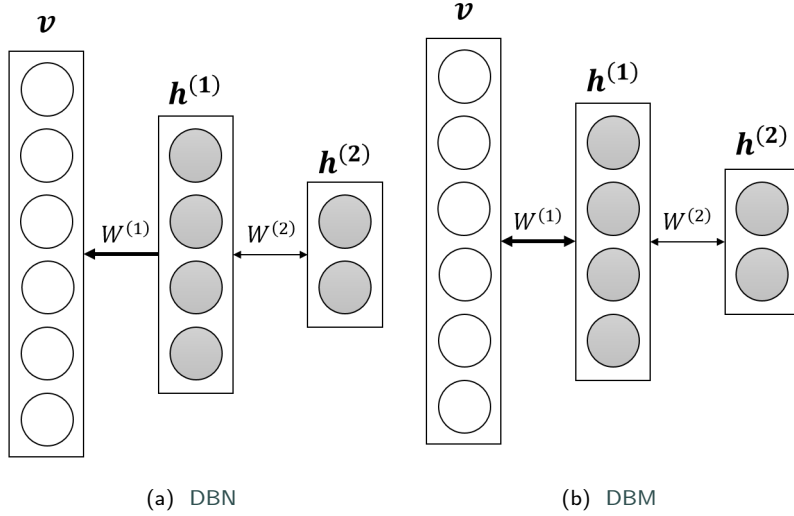


Figure 5: Graphs of Deep Belief Network and Deep Boltzmann Machines. In both figures, all links between neurons are represented with one arrow for the visibility. In Fig.5a, links between the two last layers are undirected and structures between other layers are directed graph.

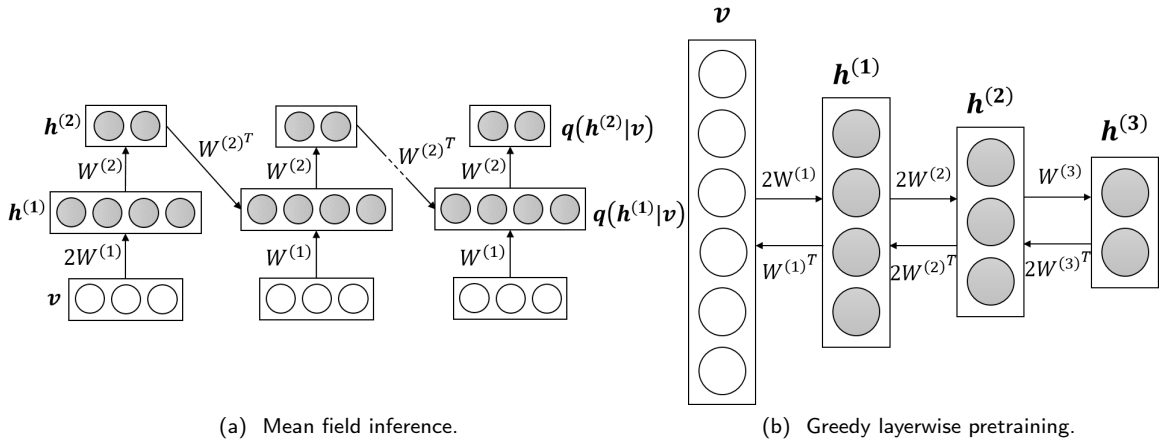


Figure 6: Methods to train DBM. In Fig.6a, each iteration consists in updating $h^{(1)}$ with v and $h^{(2)}$ from the previous iteration. $h^{(2)}$ is then updated. For the first iteration, $h^{(2)}$ value is 0 but the transfer matrix W_1 is double. The number of iterations has to be sufficiently large to stabilize the estimation probability $q(h^{(2)}|v)$. Note that $q(h^{(2)}|v)$ is not binary but represents the expectation of the state $h^{(2)}$. In Fig.6b, each RBM is trained independently. For the first RBM, $h^{(1)}$ is updated with twice the weight matrix and the bias. For the other RBM each layer has been updated with twice the weight matrix except in the last RBM where the last layer is updated normally. Once a RBM is trained, the training set is converted into the hidden layer with twice the weight matrix for the next RBM.

for each iteration due to the change of weight matrices. A greedy layerwise pretraining of DBM has been proposed by Salakhudinov and Larochelle in [85, 84] to simplify the training of the DBM. Like DBN, each RBM is trained independently but weight values are doubled to compute layers which is connected with two layers. Fig.6b illustrates one step of training.

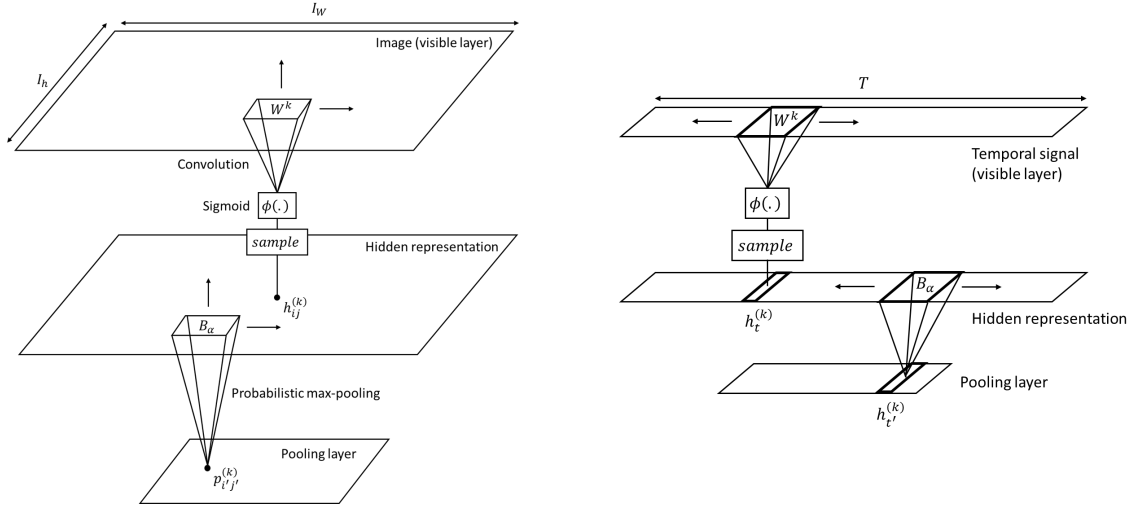


Figure 7: Max-pooling convolutional RBM. The figure on the left is the original model designed for image processing. The figure on the right is a 1-D convolutional RBM adapted for temporal signal processing.

3.2. Convolutional Boltzmann Machine

Deep RBM architectures ignore the geometric data structure and connect all visible units to each hidden unit. RBMs can become quickly huge and hard to train efficiently in some application like image processing. Lee *et al.* [50] proposed the convolutional Restricted Boltzmann Machine (**conv-RBM**) using as depicted in Fig.7 a set of K convolutional filters for sampling K hidden representations of the input image v (here, we use the operator "*" for the convolution operation):

$$\Pr(h_{ij}^{(k)} = 1 | v) = \phi \left(\xi_k^h + (W^{(k)})^T * v \right)_{i,j}, \quad (21)$$

$$\Pr(v_{ij} = 1 | \{h^{(k)}\}_{1 \leq k \leq K}) = \phi \left(\xi^v + \sum_{k=1}^K (W^{(k)} * h^{(k)})_{i,j} \right). \quad (22)$$

In order to stack **conv-RBM** to form a convolutional **DBN**, the dimensions of the hidden representation are reduced using a *probabilistic max-pooling* operator. The probabilistic max-pooling consists in converting blocks B_α in the hidden representation $h^{(k)}$ into a single pixel $p_\alpha^{(k)}$. The pixel $p_\alpha^{(k)}$ is equal to one if one hidden units in the block B_α is equal to one. A constraint on hidden units in B_α is enforced: at most one hidden unit can be equal to one in B_α . The structure of a **conv-RBM** is given in Fig.7 and the energy function of the max-pooling **conv-RBM** is:

$$E_{\text{conv-RBM}}(I, h) = - \sum_{1 \leq i,j \leq N_H} \sum_{k=1}^K \left(h_{ij}^{(k)} (W^{(k)})^T * v \right)_{i,j} + \xi_k^h h_{ij}^{(k)} - \xi^v \sum_{1 \leq i,j \leq N_V} v_{ij}, \quad (23)$$

subj. to $\sum_{(i,j) \in B_\alpha} h_{ij}^{(k)} \leq 1, \forall k, \alpha$

In practice, deep architectures are preferred to capture more complex features. **conv-RBMs** are stacked to form a convolutional Deep Belief Network (**cDBN**). **conv-RBM** has been introduced to detect pedestrian [73, 99], but this model has also been used to model dynamic data like in [82] for environmental sound classification and in [87] for rolling bearing fault analysis. A sound signal being seen as a 1D image (see Fig 7), the model learns dynamic dependencies between sampled values.

3.3. Boltzmann Machine with real value visible units

Neurons in **BMs** are binary. However, variables of real systems generally have a continuous or discrete behaviour. This is especially the case in image processing [94] where the pixels of an image vary between 0 and 256 in 8 bits

coding. To keep the continuous behaviour of the visible units, several extensions of the BM have been proposed to tolerate continuous visible units.

3.3.1. Gaussian-Bernoulli Restricted Boltzmann Machines

In 2006, Hinton proposed the GBRBM [30] to reduce the dimensionality of the RBM using continuous visible units. Hidden units keep on the binary behaviour and visible units are random variables following a conditioned Gaussian distribution (hence the name) with a variance σ_{ii}^2 (see Fig.8a), *i.e.*:

$$\Pr(v_i|\mathbf{h}) \sim \mathcal{N}(W(i, :)\mathbf{h} + \xi_i^v, \sigma_{ii}^2). \quad (24)$$

The energy is now defined as:

$$E_{GBRBM}(\mathbf{v}, \mathbf{h}) = \sum_{i=1}^n \frac{(v_i - \xi_i^v)^2}{2\sigma_{ii}^2} - \sum_{i=1}^n \sum_{j=1}^m \frac{v_i}{\sigma_{ii}^2} h_j W_{ij} - \sum_{j=1}^m h_j \xi_j^h. \quad (25)$$

An improved training procedure of the GBRBM is proposed in [9]. The GBRBM is now systematically combined with new extensions of the RBM to handle continuous data. For example, [10] proposed a DBM with Gaussian visible units for facial reconstruction. The main limitation of the GBRBM is that the model does not learn the covariance between visible units, and so GBRBM gives unsatisfactory results for modeling natural images as in [28] where neighboring pixels may be strongly correlated. A recent paper reconsiders the GBRBM as a mixture of Gaussian model for modeling natural image statistics [59].

3.3.2. Mean and covariance Restricted Boltzmann Machines

The mean and covariance Restricted Boltzmann Machine (mcRBM) proposed by Hinton and Ranzato [28] has two groups of hidden units: mean units and covariance units. Mean units capture information about the mean intensity of each visible neurons and covariance units model dependencies between visible units. The BM between visible and mean units forms the GBRBM and the BM with the visible and covariance units forms the covariance Restricted Boltzmann Machine (covRBM). The energy function of a mcRBM is defined as the sum of the energy of a GBRBM between the visible units $\mathbf{v}$ and the mean units $\boldsymbol{\mu}$ (see Eq.25) and the energy of a covRBM between visible units $\mathbf{v}$ and covariance units $\mathbf{c}$:

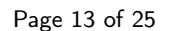
$$E_{mcRBM}(\mathbf{v}, \boldsymbol{\mu}, \mathbf{c}) = E_{GBRBM}(\mathbf{v}, \boldsymbol{\mu}) + E_{covRBM}(\mathbf{v}, \mathbf{c}). \quad (26)$$

The covariance term aims to capture correlations between visible neurons. An intuitive idea is to define a tensor W where each coefficient W_{ijk} associates two visible neurons, v_i and v_j and one hidden neurons h_k . In the case of natural images whose dimensions are those of the visible layer, the number of parameters shall increase in an exponential way with the number of hidden neurons.

To avoid a large number of parameters, the tensor is replaced by two factor matrices R and P . The first matrix R computes a multiple projection of visible neurons. The product $\mathbf{v}^T R$ will give a row vector where each component is a projection of $\mathbf{v}$ filtered by a column of R . Each coefficients of the row vector are squared to avoid divergence of parameters during the learning [13]. The second factor matrix P gives the projection to sample covariance hidden units. The energy of the covariance hidden units is given by:

$$E_{covRBM}(\mathbf{v}, \mathbf{c}) = -\mathbf{c}^T \boldsymbol{\xi}^c - (\mathbf{v}^T R)^2 P \mathbf{c}. \quad (27)$$

$\boldsymbol{\xi}^c$ is the bias vector of the covariance layer. The structure of a mcRBM is given in Fig. 8b. In practice, the Hybrid Monte Carlo (HMC) sampling algorithm [72, sect.5] is used to teach mcRBM to prevent sampling of visible units which required to inverse a square matrix of dimension $\dim(\mathbf{v})^2$. Like the conv-RBM, mcRBM has been proposed for image processing but it has also been used to model dynamic data. In [13], Dahl used mcRBM for the speech recognition task. The difference between the conv-RBM and the mcRBM lies in the modelling approach of those models. On one hand the conv-RBM models local correlations – we have an assumption on the geometry (spatial and/or temporal) of data –, on the other hand the mcRBM, considers only global correlation.



$W^{(i)}$ is the i -th weight matrix ($n \times k$) between $\mathbf{v}$ and $\mathbf{s}^{(i)}$. $\alpha^{(i)}$ and Λ are both diagonal matrix which prevent $\mathbf{s}^{(i)}$ and $\mathbf{v}$ from having large values. Courville *et al.* showed that *ssRBM* provides better results than *mcRBM* in image classification tasks [11]. However, according to [21, chap. 20], the risk with the *ssRBM* is to obtain a non-positive definite covariance matrix which can be avoided with heuristic rules. Courville also proposed some extensions to the *ssRBM* [12] to provide results in classification task. Goodfellow [22] added an additional term to the *ssRBM* to make the partition function Z tractable at the cost of losing the generative property. In [104], the *ssRBM* has been mixed with the *DBM* and the *conv-RBM* to form a Contrastive spike and slab Convolutional Deep Boltzmann Machine (*CssCDBM*) for image classification.

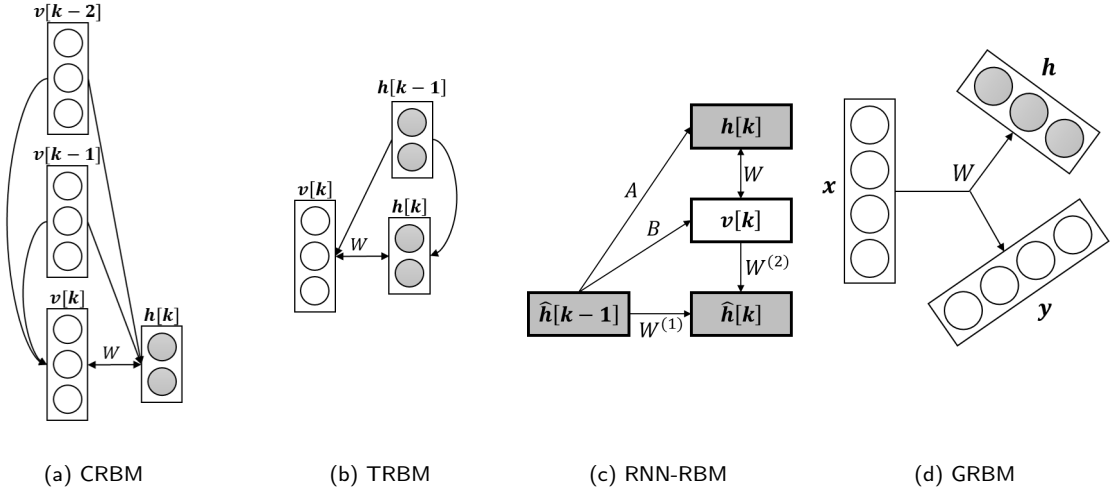


Figure 9: Architectures of RBMs for sequential dynamic systems. One direction arrows materialize the dependencies. The probability of interest is the conditional probability of the new state given past states. In 9c, the prediction of $\hat{h}[k]$ is a RNN.

3.4. Boltzmann Machine for dynamic systems

Modelling causal relationships is essential for time series forecasting. Prediction models are usually discriminative. Different approaches using modified MLP have been proposed to model dynamic data like the Elman Networks [77], autoencoders [46], Recurrent Neural Network (RNN) [61] or Long-Short Term Memory (LSTM) [33].

In BMs, the state of the network is updated from the visible layer of the training set which in this particular case is $\mathcal{D} = \{v[k]\}_{1 \leq k \leq N}$, until the equilibrium state is reached. All $v[k]$ are treated independently regardless the previous state. Models previously presented in this paper can be used to handle dynamic data by using a short-time window to the visible units as illustrated in [45, 89] which make use of extensions of BMs but alternative structures have also been proposed mixing generative and discriminative properties of the RBM to take into account past observations, *e.g.* the Discriminative Restricted Boltzmann Machine (DRBM) and the Hybrid Discriminative Restricted Boltzmann Machine (HDRBM) used for classification tasks [48, 47].

3.4.1. Conditional Restricted Boltzmann Machine

The conditional Restricted Boltzmann Machine (CRBM) [64] is the extension of the vector autoregressive (VAR) model [55] using RBM structure (see Fig.9a). A linear combination of the past observations are added to the bias values of the visible and the hidden layers (see Eq.31). The energy of a CRBM and a classical RBM are similar:

$$\begin{cases} \xi^v[k] = \xi^v[0] + \sum_{i=1}^p B^{(i)} v[k-i], \\ \xi^h[k] = \xi^h[0] + \sum_{i=1}^p C^{(i)} v[k-i]. \end{cases} \quad (31)$$

$\xi^v[0]$ and $\xi^h[0]$ are fixed biases of the visible and hidden units. $B^{(i)}$ and $C^{(i)}$ are transfer matrices between the observation $v[k-i]$ and, respectively, the visible and the hidden layers. p is of past observation order. Unlike VAR model, there is no proposed criterion for choosing p . The CRBM can also be used in other applications such as modelling the dependencies of a Multiple Input Multiple Output (MIMO) system as in [102], intra/inter-gender voice conversion [103] or missing label classification [53].

3.4.2. Temporal Restricted Boltzmann Machines

A similar structure of CRBM has been proposed in [91] named Temporal Restricted Boltzmann Machine (TRBM) which models the current state of the network conditionally on past hidden units. This structure reminds the HMM.

Eq.32 of the TRBM is slightly modified with respect to Eq.31 to describe CRBM:

$$\xi^h[k] = \xi^h[0] + \sum_{i=1}^p A^i h[k-i]. \quad (32)$$

Past hidden units are required to update present hidden units. Because hidden units are not available, training a TRBM using CD algorithm requires to sample hidden units from a sequence of observations. A large number of applications has been proposed using TRBM and its extensions like in [16] with an Input-Output TRBM used for 2D facial expression transfer. An widely used extension of TRBM is the Recurrent Temporal Restricted Boltzmann Machine (RTRBM) [92] (see Fig.9b) where visible and hidden units at time k ($v[k]$, $h[k]$) depends conditionally on the previous hidden state $h[k-1]$. The hidden units in a TRBM memorize past observations. In the literature, we can find a large number of TRBM-based architectures for different applications. For instance Nakashika *et al.* used the RTRBM for voice conversion in [71, 70]. Mittelman introduced the Structured Recurrent Temporal Restricted Boltzmann Machines (SRTRBM) and the Spike-and-slab SRTRBM to learn time series signals [62] for different examples (motion capture video or weather modeling). Introduced by Boulanger-Lewandowski, the Recurrent Neural Network - Restricted Boltzmann Machine (RNN-RBM) [4] (see Fig.9c) is a combination of a RTRBM and a RNN. The expectation of hidden units (or *mean field*) $\hat{h}[k]$ is propagated through the RNN structure:

$$\hat{h}[k] = \phi \left(W^{(1)} \hat{h}[k-1] + W^{(2)} \hat{v}[k] + \xi^h \right), \quad (33)$$

and the biases in the RTRBM are defined as:

$$\begin{cases} \xi^v[k] = \xi^v[0] + A \hat{h}[k], \\ \xi^h[k] = \xi^h[0] + B \hat{h}[k]. \end{cases} \quad (34)$$

The RNN-RBM is a popular extension of the TRBM. It provides better results than RTRBM for human motion modelling or polyphonic music modelling [4]. A similar approach replacing the RNN by a LSTM has been proposed in [98] to model long term dependencies in music generation.

3.4.3. Gated Restricted Boltzmann Machine

Initially proposed for video compression and denoising using the temporal structure by Memisevic [60], the Gated Restricted Boltzmann Machine (GRBM) is a discriminative extension of the RBM (see Fig.9d for a graphical representation). It is composed of two visible layers and a hidden layer: the input layer $\mathbf{x}$, the output layer $\mathbf{y}$ and the hidden layer $\mathbf{h}$. The three layers are connected with a 3D tensor W of dimensions $(n_x \times n_y \times n_h)$. The energy function is given by:

$$E_{GRBM}(\mathbf{y}, \mathbf{h}; \mathbf{x}) = - \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} \sum_{k=1}^{n_h} W_{ijk} x_i y_j h_k - \sum_{k=1}^{n_h} \xi_k^h h_k - \sum_{j=1}^{n_y} \xi_j^y y_j. \quad (35)$$

Here, the function to maximize during the training stage is the conditional probability $\Pr(\mathbf{y}|\mathbf{x})$:

$$\Pr(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{h}} \Pr(\mathbf{y}, \mathbf{h}|\mathbf{x}) = \sum_{\mathbf{h}} \frac{1}{Z(\mathbf{x})} \exp(-E_{GRBM}(\mathbf{y}, \mathbf{h}; \mathbf{x})), \quad (36)$$

with $Z(\mathbf{x})$ the marginalized function. Inference test allows to estimate $\mathbf{h}$ and $\mathbf{y}$ as functions of $\mathbf{x}$. The tensor W in Eq.35 captures the correlations between the input and the output layers. Setting $\mathbf{y} = \mathbf{x}$ allow the GRBM to capture dependencies between the input components of $\mathbf{x}$ [80] as for an autoencoder. For large inputs, the number of parameters may quickly become too large to use this model for real time applications. Models using factors like mcRBM (see Fig.8b) are preferred to GRBM to model the correlations between the visible neurons. Thereby the GRBM can be used to estimate observation $\hat{\mathbf{x}}[k+1]$ as function of past $\hat{\mathbf{x}}[k]$ [46]. Fig.9d gives the graph representation of the GRBM.

These RBM-based approaches for modelling dynamic systems are used with sequential data. The next section introduces the Diffusion Network for the modelling of dynamic continuous systems.

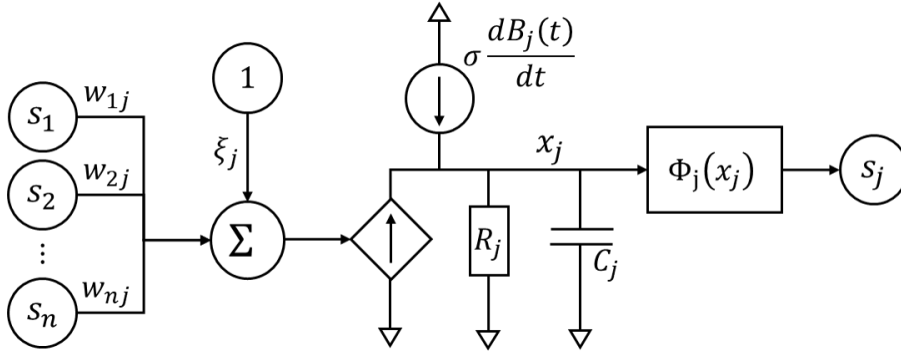


Figure 10: Electronic structure of the neuron j from a DN. An important difference with the Boltzmann Machine and its DN extension is a DN, the signal to model $x_j(t)$ is the input of the activation function and not the output $s_j(t)$.

4. Diffusion Network

4.1. Description

Between 1982 and 1984, Hopfield introduced two learning models based on ANN to mimic the behaviour of biological neurons. First, the bHN [34], is a deterministic neural network with binary units (neurons). This model leads to the popular BM proposed by Hinton [25] with many extension proposed in previous the section. Second, the cHN [35] which is the continuous version of the bHN. Movellan proposed as well a stochastic extension of the cHN to model Diffusion Processes (DP): the DN [67]. The DP is a continuous time Markov process. The aim of the DP is to model the dynamic dependencies between real valued signals. Let $\mathbf{X}(t)$ be the n -dimensional signal vector at time t . Learning a DP consists to find the best Stochastic Differential Equation (SDE) [54] describing the evolution of $\mathbf{X}(t)$:

$$d\mathbf{X}(t) = \mu(\mathbf{X}(t))dt + \sigma d\mathbf{B}(t). \quad (37)$$

Eq.37 has two parts: first, the *drift* function $\mu(\mathbf{X}(t))$ which is the deterministic part of the model and the *diffusion* term $\sigma d\mathbf{B}(t)$ which corresponds to the stochastic part of the model. $d\mathbf{B}(t)$ is a simple Brownian motion vector. We can also write the SDE for each neuron j :

$$dx_j(t) = \mu_j(\mathbf{X}(t))dt + \sigma dB_j(t), \quad (38)$$

The DN is a fully connected neural network (see Fig.2), the structure of each neuron being represented in Fig.10. The drift component is computed from the inputs. An additive Gaussian noise models the stochastic part of the model. The drift of the neuron j is written:

$$\mu_j(\mathbf{X}(t)) = \kappa_j \left(-\rho_j x_j(t) + \xi_j + \sum_{i=1}^n W_{ij} \phi_i(x_i(t)) \right). \quad (39)$$

The drift can also be represented by an electronic scheme as given in Fig.10. The sum of weighted neuron state inputs is converted into a current. The filter $R_j C_j$ converts the input into a voltage $x_j(t)$ which drives the dynamic of the system. This structure differs from the cHN's structure of neuron with the presence of the noise generator. The noise helps to avoid bad solutions. A major difference between the DN and the cHN is the DN focuses on the modelling of the dynamics of $\mathbf{X}(t) = (x_i(t))_{1 \leq i \leq n}^T$ (the vector input of the activation function) whereas the cHN models static data by converging to an equilibrium (stable) vector: $\langle \mathbf{S} \rangle_\infty = (\langle s_i \rangle_\infty)_{1 \leq i \leq n}^T$ (the vector output of the activation function).

The activation function of a DN is a sigmoid function:

$$s_j = \phi_j(x_j) = \theta_L + (\theta_H - \theta_L) \frac{1}{1 + \exp(-a_j x_j)} \quad (40)$$

where θ_L and θ_H are respectively the lower and the upper bounds of the function. a_j is a slope parameter of $\phi_j(\cdot)$. The influences of the parameter a_j are in the behavior of neurons and in the noise regularization. An illustration of the role of a_j is given in Fig.11.

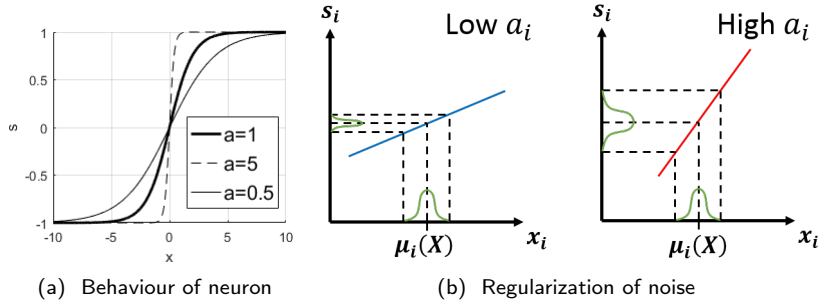


Figure 11: Influence of the activation parameters a_i . Fig.11a displays the activation between -1 and 1 for three different values of a . The more the parameter a_i is big, the more the neuron will have a binary behaviour. The more a_i tends to zero, the more the neuron will have a linear behaviour. In Fig.11b, the two schemes describe how the noise influences the state of the neuron. At a given time, $x_i(t)$ is a Gaussian random variable centered on $\mu_i(X(t))$ with a variance σ^2 (see Eq.38). In the left figure, a_i has a low value, the slope of the function is almost horizontal. The dispersion of the noise is "compressed" and the state of the neuron is almost deterministic. If a_i is high like in the second schema, the slope of the curve is more vertical and the dispersion of the state increases.

For implementation, we can write the **sequential** evolution of each neurons:

$$x_j(t + \Delta t) - x_j(t) = \kappa_j \left(-\rho_j x_j(t) + \xi_j + \sum_{i=1}^n W_{ij} \phi_i(x_i(t)) \right) \Delta t + \sigma z_j(t) \sqrt{\Delta t}, \quad (41)$$

where $z_j(t)$ is a standard Gaussian noise. The expression of the next sample is given by:

$$x_j(t + \Delta t) = (1 - \kappa_j \rho_j \Delta t) x_j(t) + \kappa_j \xi_j \Delta t + \kappa_j \sum_{i=1}^n W_{ij} \phi_i(x_i(t)) \Delta t + \sigma z_j(t) \sqrt{\Delta t}. \quad (42)$$

Parameters ρ_j and κ_j weight input of the network and the previous state but κ_j controls the linear combination of the neuron inputs.

4.2. Learning procedure

Training a DN consists in finding the **SDE** (see Eq.37) which gives the best description of the evolution of the data. Movellan *et al.* have proposed a Monte Carlo Expectation Maximization (**MCEM**) approach to train the network [68] which consists in computing the log likelihood of the signal in a time window $[0, T]$. The path (the signal in a window) is seen as a random multivariate variable in the probabilistic universe Ω : the set of functions from $[0, T]$ to $\mathbb{R}^n$. Using the Girsanov's theorem, we can write the density of the path $\mathbf{X}$ in $[0, T]$ as:

$$\begin{aligned} \mathcal{L}(\mathbf{X}; \lambda) &= \exp \left(\frac{1}{\sigma^2} \int_0^T \mu(\mathbf{X}(t)) d\mathbf{X}(t) - \frac{1}{2\sigma^2} \int_0^T \mu(\mathbf{X}(t))^2 dt \right) \\ &= \exp \left[\sum_{j=1}^n \left(\frac{1}{\sigma^2} \int_0^T \mu_j(\mathbf{X}(t)) dx_j(t) - \frac{1}{2\sigma^2} \int_0^T \mu_j(\mathbf{X}(t))^2 dt \right) \right] \end{aligned} \quad (43)$$

where $\lambda = \left\{ (W_{ij})_{1 \leq i, j \leq n}, (\xi_i)_{1 \leq i \leq n}, (\rho_i)_{1 \leq i \leq n}, (\kappa_i)_{1 \leq i \leq n}, (a_i)_{1 \leq i \leq n} \right\}$ is the set of parameters. The integral $\int_0^T \mu_j(\mathbf{X}(t)) dx_j(t)$ is an Itô stochastic integral [74]. For a given training set $\mathcal{D} = \mathbf{X}$, learning the DN consists in finding λ which maximizes the log-likelihood of the density function $\mathcal{L}(\mathbf{X}; \lambda)$. In the literature, Movellan *et al.* proposed to maximize the log-likelihood using Expectation Maximization (**EM**), *i.e.*, each parameter λ_i is obtained by

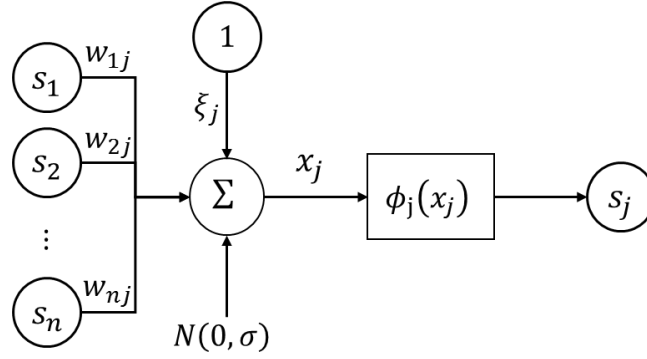


Figure 12: Structure of the neuron j of a DN-RBM. The expression of the activation function $\phi_j(x_j)$ is given in Eq.40.

deriving the log-likelihood and solving Eq.44 for all λ_i :

$$\frac{\partial \log \mathcal{L}(\mathbf{X}; \lambda)}{\partial \lambda_i} = 0 \quad (44)$$

Movellan *et al.* give the solutions of Eq.44 for the bias, the inverse resistor, the inverse capacitance and the transfer matrix. The presence of hidden units allows to extract useful information but the hidden path (the evolution of hidden units in time) is not available and must be sampled before solving Eq.44. Movellan proposed to apply the Monte Carlo's method to generate m paths of hidden units [81]. We note $\mathbf{H}^l$ the l -th sample of the hidden path. $\mathbf{X}^l = [\mathbf{V}^0, \mathbf{H}^l]$ is the l -th path regrouping visible and hidden units. Each path $\mathbf{X}^l$ is different due to the presence of hidden units, a weight $\pi(l)$ is apply to each observation:

$$\mathcal{L}^{MCEM}(\lambda|\mathcal{D}) = \sum_{l=1}^m \pi(l) \log \mathcal{L}(\mathbf{X}^l; \lambda) \quad (45)$$

where $\pi(l)$ is the weight of the path l :

$$\pi(l) = \frac{\mathcal{L}(\mathbf{X}^l; \lambda)}{\sum_{k=1}^m \mathcal{L}(\mathbf{X}^k; \lambda)} \quad (46)$$

4.3. Continuous Restricted Boltzmann Machine

Until now, there is no real-world application of DNs in the literature. However this model inspired Chen and Murray to propose the continuous Restricted Boltzmann Machine (DN-RBM) in [8], a RBM using the neuron structure of a DN. Chen and Murray proposed the abbreviation CRBM for continuous RBM. We already mentioned different models based on RBM in sect.3, using the prefix 'c' (for conditional, convolutional, covariance [63] and now continuous). To avoid any confusion with the previous models, we will call this model DN-RBM.

The DN-RBM can be seen as a particular case of the DN or the cHN. Here, we focus on the stochastic equilibrium state of the network. For all neurons, capacities C_i are the same and we have the following equality: $\rho_i \kappa_i \Delta t = 1$. Links are symmetric like in the cHN (i.e. $W_{ij} = W_{ji}$) but contrary to DN, the variable of interest are the neuron output states $\mathbf{s}$ and not the inputs $\mathbf{x}$ of the activation function. In the discrete time case, the structure of neurons is given in Fig.12. The main difference with the cHN is that the activation function is unique for each neuron like in DN. The neuron activation function is the sigmoid function defined in Eq.40.

The energy function of the DN-RBM is very similar to the energy of the cHN [36]:

$$E_{DN-RBM}(\mathbf{s} = \{\mathbf{v}, \mathbf{h}\}) = -\mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{v}^T \boldsymbol{\xi}^v - \mathbf{h}^T \boldsymbol{\xi}^h + \sum_i \frac{1}{a_i} \int_0^{s_i} \phi^{-1}(s') ds', \quad (47)$$

with $\phi^{-1}(\cdot)$ the inverse of the activation for a coefficient slope $a_i = 1$. As for BM, we can use the MCD rule to train parameters of the DN-RBM. Eq.17 does not change for the DN-RBM and the update law for the coefficient a_i reads :

$$\Delta a_i \propto \frac{1}{a_i^2} \int_{\langle s_i \rangle_1}^{\langle s_i \rangle_0} \phi^{-1}(s') ds'. \quad (48)$$

Like BMs, the DN-RBM is used to model the stochastic equilibrium of the network. This is not a dynamic model since there is no time recurrence between the observations. However, the neuron structure can be used in a large number of RBM extension (previously presented) and the continuous behavior for the hidden units allows us to capture more information than with binary units. Like for binary RBMs, it is possible to associate extension of the RBM to the DN-RBM in many applications such as wind speed forecasting in [37] or evaluation of sound quality in [39] where author train a DBN using the structure of DN-RBM.

5. Discussion

COMMENT: the entire section were added (in version 2.0).

COMMENT: Subsections have been added.

The success of the Boltzmann machines lies in the flexibility of the RBM and its adaptability to handle many problems in different fields of study. Boltzmann machines can be used for the prediction and detection of anomalies in time series. They compete with other learning approaches such as RNN, LSTM, HMM. But unlike these, BM are generative models that can learn probabilistic representations from data in an unsupervised way. A natural advantage of generative models is that, if they are fed with incomplete data, they are anyway able to recover the missing data.

5.1. Limitation and improving paths

Unfortunately, Boltzmann machines still suffer from limitations. Many papers highlight the difficulty of the RBM to model efficiently the probability distribution $P_{data}(\mathbf{x})$ and propose solutions to overcome those limitations. In addition, the family of Boltzmann machines got recently a serious challenger: the family Generative Adversarial Network. The good results of these models could diminish the interest in Boltzmann machines in particular in image processing (see [110] for example). Three paths of development were identifying to improve the performance of BMs:

- **The graph structure:** how the neurons interact each other? Researchers have proposed different structures to take account priors of the data. For example mcRBM and GRBM encourage the network to capture correlations between variables. Some researchers proposed to adapt an existing model like Lee *et al.* in [50] with the conv-RBM. A mixture of models has also been studied, like the CssCDBM or the RNN-RBM.
- **The neuron structure:** how the information is processed inside a neuron? Some modifications of the neuron structure have been proposed to better fit with the data behavior. For example the GBRBM and the DN-RBM have been proposed to work with continuous data. This is the case of the DN for which data are continuously time varying. Neuron structure has also been studied to improve the performance of the network (see for example [69]).
- **The cost function:** how the model is trained? The motivation of this path of improvement is to reduce the bias between the $P_{data}(\mathbf{x})$ and the learned probability $P_{model}(\mathbf{x})$ by modifying the cost function. This question has been addressed by changing the metrics between $P_{data}(\mathbf{x})$ and $P_{model}(\mathbf{x})$ as in [40, 56, 66] or by adding a penalization term [19].

The more neurons, the more computational efforts are needed: massive networks should not be the only way to reduce the modelization error. The choice of the dimension remains today an unsolved issue. There is no theorem nor criterion (equivalent to the Akaike Information Criterion (AIC) for auto-regressive model or Extrem Learning Machine (ELM) methods for feedforward neural networks [14]) to help decide what is the optimal number of hidden neurons and hidden layers. In addition to being energy consuming, a large dimension network is also time consuming. In many papers, authors focus on the comparison of the performance between models but do not compare computational efforts between models. The issue of computational efforts can have a major impact in particular in real-time system for example but its strongly depends on the data, the application and the used hardware.

5.2. Dynamic in generative models

Two approaches have been proposed to add dynamic prior in BMs. The first solution consists in learning correlation between temporal observations. Some researcher propose to use BM developed for image analysis to model temporal dependencies between observations (cf. [13] for example). This strategy is not recommended in the case of high dimension data: a larger window means more parameters to learn and updating the network would be time consuming. The second option consists in using RBM's extension with a temporal memory based on the hidden neurons (cf. TRBM, RTRBM or RNN-RBM for example).

The DN, a stochastic versions of cHN, generalizes the Markov model family among which HMM, Kalman-Bucy filters, BM or RNN are special cases. *E.g.* a HMM can be seen as a DN with discrete-valued hidden states and discrete-time dynamics; a Kalman filter is a DN driven by linear dynamics; a RNN is a zero-noise DN. Results on realistic sequence modeling and sequence recognition tasks are encouraging [68] but more works are needed with larger databases to produce a definitive conclusion. The reason for this good performances lies in that DN can cope with issues like bifurcations or dynamic time warping albeit combining continuous state-representation well suited to problems with unobservable quantities or with sparseness constraints as explained by Movellan in [68]. Contrary to BMs DNs did not known the same interest in the literature despite many promising properties of the DN for dynamic problem. But Theoretical issues concerning the approximating power of the stochastic differential equations need to be explored.

5.3. Summary

Table 1 lists the different models mentioned in this review and offers a comparison between them. To quantify the impact of those models we propose a mark related to the number of documents (all type) we got on SCOPUS. Marks are based on the number of documents we got when we enter the name of the model (all field). Some specific rules have been added to avoid bias for the DN and the DN-RBM. The exact number of documents is not a relevant indicator due to the introduced bias. Another bias lies in the fact that each author does not call a model with the same name (for example, the Gaussian-Bernoulli RBM has also been called the Gaussian RBM or the Continuous RBM). Also some different models can have the same name or acronym (example: CRBM with 'c' for continuous, covariance, conditional or convolutional). Finally, the DN is a very popular term used in many different fields. The mark in table 1 is put for each model before and after 2010 to provide an idea of a global evolution.

6. Conclusion

In dynamic systems, time provides a significant information to solve many problems. Ignore time dependencies between each observation can mar results models in comparison with dynamical model. Among the existing solutions to extract time information, GBN have been applied with success in many application. The basic idea behind the use of graph model is each link models dependencies between variables. Those dependencies can be time dependencies but also geometric dependencies in image processing or many others in function of the application. Different approaches and strategy have been proposed in the literature to incorporate temporal coherence between variables. First the use of a time window to dispose of last states of the system as input of the model. This solution can be quickly inefficient in the case of high dimensional input. Learning a hidden representation of time dependencies is an option to reduce the size of the model. In those model, the temporal information is stored in hidden variables. Using the geometry knowledge (conv-RBM) of variables is also a solution to model local dependencies and avoid high dimensional model issues. Finally, the cost function can be modify to take account of the dynamic like in DN with the use of the SDE.

The emergence of GBN come from results of Hopfield's works on deterministic binary network [34] and continuous network [35] which lead to many works on stochastic discrete networks and stochastic continuous and dynamic networks. The RBM proposed by Hinton has been a shockwave in the neural network community like its illustrated in Fig.1. Research on new generative architectures is most often guided by the encountered applications *e.g.* image-video processing, speech processing, bio-medical, economic or financial. The major advantage of the RBM is its flexibility. New extension of RBM are regularly proposed in the literature. Researcher often got inspired by other existing model and proposed a mixed version of the RBM by modifying the graph, the neuron's structure or the cost function.

Model Name	Date	Visible units behavior	Hidden units behavior	(G) / (D)	Mark < 2010	Mark ≥ 2010	Sources
Hidden Markov Model	1966	Discrete	Discrete	D	★ ★ ★	★ ★	[79]
binary Hopfield Network	1982	Binary	Binary	G	★ ★ ★ ★	★ ★ ★ ★	[34]
continuous Hopfield Network	1984	Continuous	Continuous	G	★ ★ ★ ★ ★	★ ★ ★ ★ ★	[35]
Boltzmann Machine	1983	Binary	Binary	G	★ ★ ★ ★ ★	★ ★ ★ ★ ★	[1]
Restricted Boltzmann Machine	1986	Binary	Binary	G	★ ★	★ ★ ★ ★ ★	[27]
Deep Belief Network	2006	Binary	Binary	G	★ ★	★ ★ ★ ★ ★	[29]
Deep Boltzmann Machine	2009	Binary	Binary	G	★ ★ ★	★ ★ ★ ★ ★	[83]
convolutional RBM	2009	Binary	Binary	G	★	★ ★ ★ ★ ★	[50]
convolutional DBN	2009	Binary	Binary	G	★ ★	★ ★ ★ ★ ★	[50]
Gaussian-Bernoulli RBM	2006	Continuous	Binary	G	★ ★ ★	★ ★ ★ ★	[10]
covariance RBM	2010	Continuous	Binary	G	★	★ ★ ★	[80]
mean and covariance RBM	2010	Continuous	Binary	G	★	★ ★	[28]
Product of Student t-distributions	2003	Continuous	Continuous	G	★ ★	★ ★	[101]
mean Product of Student t-distributions	2010	Continuous	B and C	G		★ ★	[63]
spike and slab RBM	2011	Continuous	B and C	G		★ ★	[11]
Contrastive spike and slab Convolutional DBM	2018	Continuous	B and C	G		★	[104]
Conditional RBM	2007	Binary	Binary	G	★	★ ★ ★	[86]
Gated RBM	2007	Binary	Binary	G	★	★ ★ ★	[60]
Discriminative RBM	2008	Binary	Binary	D	★	★ ★ ★ ★	[47]
Hybrid Discriminative RBM	2008	Binary	Binary	D and G	★	★ ★ ★ ★	[47]
Temporal RBM	2007	Binary	Binary	D	★	★ ★ ★ ★	[91]
Recurrent Temporal RBM	2009	Binary	Binary	D	★	★ ★ ★	[92]
Structured Recurrent Temporal RBM	2014	Binary	Binary	D	★	★ ★	[62]
Recurrent Neural Network - RBM	2012	Binary	Binary	D	★	★ ★ ★ ★	[4]
Diffusion Network	1993	Continuous	Continuous	G	★	★ ★ ★ ★	[68]
Diffusion Network - RBM	2002	Continuous	Continuous	G	★	★	[7]

Table 1

Table of comparison of the different models.

Column (G) / (D) refers to Generative model of Discriminative model.

Marks are based on the number of document on SCOPUS when we enter the name of the model (some specific rule has been added to avoid bias). The exact number of documents is not a relevant indicator due to the introduced bias. We choose to give a mark in function of the order of magnitude of the number of documents.

WARNING: the column of visible units behaviour is not significantly because Gaussian visible units of the GBRBM are almost always used in every BM extension.

References

- [1] E. HL Aarts and J. HM Korst. Boltzmann machines as a model for parallel annealing. *Algorithmica*, 6(1-6):437–465, 1991.
- [2] C. Assis, A. C.M. Pereira, E.G Carrano, R. Ramos, and W. Dias. Restricted Boltzmann machines for the prediction of trends in financial time series. In *IEEE International Joint Conference on Neural Networks*, pages 1–8, 2018.
- [3] Y. Bengio and O. Delalleau. Justifying and generalizing contrastive divergence. *Neural computation*, 21(6):1601–1621, 2009.
- [4] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392*, 2012.
- [5] M. Carreira-Perpinan and G. Hinton. On contrastive divergence learning. In *Conference of Artificial Intelligence and Statistics*, volume 10, pages 33–40, 2005.
- [6] C.P. Chen, C.Y. Zhang, L. Chen, and M. Gan. Fuzzy restricted Boltzmann machine for the enhancement of deep learning. *IEEE Transactions on Fuzzy Systems*, 23(6):2163–2173, 2015.
- [7] H. Chen, P. Fleury, and A.F. Murray. Continuous-valued probabilistic behavior in a vlsi generative model. *IEEE Transactions on Neural Networks*, 17(3):755–770, 2006.
- [8] H. Chen and A.F. Murray. Continuous restricted Boltzmann machine with an implementable training algorithm. *IEE Proceedings-Vision, Image and Signal Processing*, 150(3):153–158, 2003.
- [9] K. Cho, A. Ilin, and T. Raiko. Improved learning of gaussian-bernoulli restricted Boltzmann machines. In *International conference on artificial neural networks*, pages 10–17, 2011.
- [10] K.H. Cho, T. Raiko, and A. Ilin. Gaussian-bernoulli deep Boltzmann machine. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, Aug 2013.
- [11] A. Courville, J. Bergstra, and Y. Bengio. A spike and slab restricted Boltzmann machine. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 233–241, 2011.
- [12] A. Courville, G. Desjardins, J. Bergstra, and Y. Bengio. The spike-and-slab rbm and extensions to discrete and sparse data distributions. *IEEE transactions on pattern analysis and machine intelligence*, 36(9):1874–1887, 2014.
- [13] G. Dahl, A.R. Mohamed, G. Hinton, and Ranzato M.A. Phone recognition with the mean-covariance restricted Boltzmann machine. In *Advances in neural information processing systems*, pages 469–477, 2010.
- [14] S. Ding, X. Xu, and R. Nie. Extreme learning machine and its applications. *Neural Computing and Applications*, 25(3-4):549–556, 2014.
- [15] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.
- [16] Matthew D.Z., Graham W.T., L. Sigal, Iain M., and R. Fergus. Facial expression transfer with input-output temporal restricted Boltzmann machines. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1629–1637. Curran Associates, Inc., 2011.
- [17] S.E. Fahlman, G. Hinton, and T. Sejnowski. Massively parallel architectures for AI: NETL, thistle, and Boltzmann machines. In *National Conference on Artificial Intelligence*, pages 109–113, Jan 1983.
- [18] F. Feng, R. Li, and X. Wang. Deep correspondence restricted Boltzmann machine for cross-modal retrieval. *Neurocomputing*, 154:50–60, 2015.
- [19] C.K. Fisher, A.M. Smith, and J.R. Walsh. Boltzmann encoded adversarial machines, 2018.
- [20] C. Giraud. *Introduction to High-Dimensional Statistics*. Chapman and Hall/CR, 2014.
- [21] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- [22] I. Goodfellow, A. Courville, and Y. Bengio. Spike-and-slab sparse coding for unsupervised feature discovery. *arXiv preprint arXiv:1201.3382*, 2012.
- [23] R. Hecht-Nielsen. Theory of the backpropagation neural network. In *International Joint Conference on Neural Networks*, volume 1, pages 593–605, Washington, DC, USA, 1989.
- [24] J.A. Hertz, A.S. Krogh, and R.G. Palmer. *Introduction To The Theory Of Neural Computation*, volume 1 of *Santa Fe Institute Series*. Westview Press, 1991.
- [25] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.
- [26] G. Hinton. Deep belief networks. *Scholarpedia*, 4(5):5947, 2009.
- [27] G. Hinton. A practical guide to training restricted Boltzmann machines. In Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade (2nd ed.)*, volume 7700, pages 599–619. Springer, 2012.
- [28] G. Hinton and Ranzato M.A. Modeling pixel means and covariances using factorized third-order Boltzmann machines. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2551–2558. IEEE, 2010.
- [29] G. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.
- [30] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.
- [31] G. Hinton and R. Salakhutdinov. Replicated softmax: an undirected topic model. In *Advances in neural information processing systems*, pages 1607–1614, 2009.
- [32] G. Hinton and T. Sejnowski. Learning and relearning in Boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(282-317):2, 1986.
- [33] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [34] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.
- [35] J.J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, 81(10):3088–3092, 1984.
- [36] J.J. Hopfield and D.W. Tank. Computing with neural circuits: A model. *Science*, 233(4764):625–633, 1986.
- [37] Y. Hu, J. Liu, J. You, and P.W. Chan. Continuous rbm based deep neural network for wind speed forecasting in hong kong. In *Proceedings of*

- the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 368. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015.
- [38] G.B. Huang, H. Lee, and E. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2518–2525, 2012.
- [39] H.B. Huang, R.X. Li, M.L. Yang, T.C. Lim, and W.P. Ding. Evaluation of vehicle interior sound quality using a continuous restricted Boltzmann machine-based DBN. *Mechanical Systems and Signal Processing*, 84:245–267, 2017.
- [40] A. Hyvärinen. Some extensions of score matching. *Computational statistics & data analysis*, 51(5):2499–2512, 2007.
- [41] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.
- [42] S. Khan and T. Yairi. A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing*, 107:241–265, 2018.
- [43] V. Kuleshov and S. Ermon. Neural variational inference and learning in undirected graphical models. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6734–6743. Curran Associates, Inc., 2017.
- [44] S. Kullback and R.A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.
- [45] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi. Time series forecasting using a deep belief network with restricted Boltzmann machines. *Neurocomputing*, 137:47–56, 2014.
- [46] M. Längkvist, L. Karlsson, and A. Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24, 2014.
- [47] H. Larochelle and Y. Bengio. Classification using discriminative restricted Boltzmann machines. In *Proceedings of the 25th international conference on Machine learning*, pages 536–543. ACM, 2008.
- [48] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio. Learning algorithms for the classification restricted Boltzmann machine. *Journal of Machine Learning Research*, 13:643–669, Mar 2012.
- [49] Y. LeCun, D. Touresky, G. Hinton, and T. Sejnowski. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pages 21–28. CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.
- [50] H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 609–616, New York, NY, USA, 2009. ACM.
- [51] H. Lee, P. Pham, Y. Largman, and A.Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in neural information processing systems*, pages 1096–1104, 2009.
- [52] X. Li. A spatial–temporal hopfield neural network approach for super-resolution land cover mapping with multi-temporal different resolution remotely sensed images. *ISPRS Journal of photogrammetry and remote sensing*, 93:76–87, Jan 2014.
- [53] X. Li, F. Zhao, and Y. Guo. Conditional restricted Boltzmann machines for multi-label learning with incomplete labels. In *Artificial Intelligence and Statistics*, pages 635–643, 2015.
- [54] A Ludwig. *Stochastic Differential Equations as Dynamical Systems*, pages 489–495. Birkhäuser Boston, Boston, MA, 1990.
- [55] H. Lütkepohl. *Vector Autoregressive Models*, pages 1645–1647. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [56] S. Lyu. Interpretation and generalization of score matching. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 359–366. AUAI Press, 2009.
- [57] D. MacKay. Failures of the one-step learning algorithm. In *Available electronically at <http://www.inference.phy.cam.ac.uk/mackay/abstracts/gbm.html>*, Sep 2001.
- [58] B. Marlin, K. Swersky, B. Chen, and N. Freitas. Inductive principles for restricted Boltzmann machine learning. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 509–516, Chia Laguna Resort, Sardinia, Italy, May 2010. PMLR.
- [59] J. Melchior, N. Wang, and L. Wiskott. Gaussian-binary restricted Boltzmann machines for modeling natural image statistics. *PloS one*, 12(2):e0171015, 2017.
- [60] R. Memisevic and G. Hinton. Unsupervised learning of image transformations. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.
- [61] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048. ISCA, 2010.
- [62] R. Mittelman, B. Kuipers, S. Savarese, and H. Lee. Structured recurrent temporal restricted Boltzmann machines. In *International Conference on Machine Learning*, pages 1647–1655, 2014.
- [63] V. Mnih, G. Hinton, and M.A. Ranzato. Generating more realistic images using gated MRF's. In *Advances in Neural Information Processing Systems*, pages 2002–2010, 2010.
- [64] V. Mnih, H. Larochelle, and G. Hinton. Conditional restricted Boltzmann machines for structured output prediction. *arXiv preprint arXiv:1202.3748*, 2012.
- [65] A.R. Mohamed and G. Hinton. Phone recognition using restricted Boltzmann machines. In *Acoustics Speech and Signal Processing*, pages 4354–4357. IEEE, 2010.
- [66] G. Montavon, K.-R. Müller, and M. Cuturi. Wasserstein training of restricted Boltzmann machines. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3718–3726. Curran Associates, Inc., 2016.
- [67] J. Movellan and J. McClelland. Learning continuous probability distributions with symmetric diffusion networks. *Cognitive Science*, 17(4):463–496, 1993.
- [68] J. Movellan, P. Mineiro, and R. Williams. A Monte Carlo EM approach for partially observable diffusion processes: Theory and applications to neural networks. *Neural computation*, 14(7):1507–1544, 2002.

- [69] V. Nair and G. Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th international conference on machine learning*, pages 807–814, 2010.
- [70] T. Nakashika, T. Takiguchi, and Y. Ariki. High-order sequence modeling using speaker-dependent recurrent temporal restricted Boltzmann machines for voice conversion. *Proceedings of the Annual Conference of the International Speech Communication Association, INTER-SPEECH*, pages 2278–2282, Jan 2014.
- [71] T. Nakashika, T. Takiguchi, and Y. Ariki. Voice conversion using RNN pre-trained by recurrent temporal restricted Boltzmann machines. *Transactions on Audio, Speech and Language Processing*, 23(3):580–587, 2015.
- [72] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Canada, Sep 1993.
- [73] M. Norouzi, M. Ranjbar, and G. Mori. Stacks of convolutional restricted Boltzmann machines for shift-invariant feature learning. In *Conference on Computer Vision and Pattern Recognition*, pages 2735–2742. IEEE, 2009.
- [74] B. Øksendal. *Stochastic differential equations*. Springer, 2003.
- [75] J.K. Paik and A.K. Katsaggelos. Image restoration using a modified hopfield network. *IEEE Transactions on image processing*, 1(1):49–63, 1992.
- [76] J.H. Park, Y.S. Kim, I.K. Eom, and K.Y. Lee. Economic load dispatch for piecewise quadratic cost function using hopfield neural network. *IEEE transactions on power systems*, 8(3):1030–1038, 1993.
- [77] D. T. Pham and X. Liu. Training of elman networks and dynamic system modelling. *International Journal of Systems Science*, 27(2):221–226, 1996.
- [78] W. HL Pinaya, A. Gadelha, O.M. Doyle, C. Noto, A. Zugman, Q. Cordeiro, A.P. Jackowski, R.A. Bressan, and J.R. Sato. Using deep belief network modelling to characterize differences in brain morphometry in schizophrenia. *Scientific reports*, 6:38897, 2016.
- [79] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [80] M.A Ranzato, A. Krizhevsky, and G. Hinton. Factored 3-way restricted Boltzmann machines for modeling natural images. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 621–628, Chia Laguna Resort, Sardinia, Italy, May 2010. PMLR.
- [81] C. Robert and G. Casella. *Monte Carlo statistical methods*. Springer-Verlag, 2013.
- [82] H.B. Sailor, D.M. Agrawal, and H.A Patil. Unsupervised filterbank learning using convolutional restricted Boltzmann machine for environmental sound classification. In *INTERSPEECH*, pages 3107–3111, 2017.
- [83] R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 448–455, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, Apr 2009. PMLR.
- [84] R. Salakhutdinov and G. Hinton. An efficient learning procedure for deep Boltzmann machines. *Neural Computation*, 24(8):1967–2006, 2012.
- [85] R. Salakhutdinov and H. Larochelle. Efficient learning of deep Boltzmann machines. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 693–700, 2010.
- [86] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.
- [87] H. Shao, H. Jiang, H. Zhang, W. Duan, T. Liang, and S. Wu. Rolling bearing fault feature learning using improved convolutional deep belief network with compressed sensing. *Mechanical Systems and Signal Processing*, 100:743–765, 2018.
- [88] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical Report CU-CS-321-86, University of Colorado, Department of Computer Science, Boulder, US, 1986.
- [89] R. Souriau, V. Vigneron, J. Lerbet, and H. Chen. Boltzmann machines for signals decomposition. Application to Parkinson’s disease control. In *XXVIIème Colloque francophone de traitement du signal et des images (GRETSI 2019)*, Lille, France, Aug 2019.
- [90] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep Boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230, 2012.
- [91] I. Sutskever and G. Hinton. Learning multilevel distributed representations for high-dimensional sequences. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 548–555, San Juan, Puerto Rico, Mar 2007. PMLR.
- [92] I. Sutskever, G. Hinton, and G. Taylor. The recurrent temporal restricted Boltzmann machine. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1601–1608. Curran Associates, Inc., 2009.
- [93] G.W. Taylor, G. Hinton, and S.T. Roweis. Modeling human motion using binary latent variables. In *Advances in neural information processing systems*, pages 1345–1352, 2007.
- [94] Y.W. Teh and G. Hinton. Rate-coded restricted Boltzmann machines for face recognition. In *Advances in neural information processing systems*, pages 908–914, 2001.
- [95] T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071. ACM, 2008.
- [96] V. Upadhyay and PS. Sastry. An overview of Restricted Boltzmann Machines. *Journal of the Indian Institute of Science*, pages 1–12, 2019.
- [97] C. Villani. *Optimal transport: old and new*, volume 338. Springer, 2008.
- [98] R. Vohra, K. Goel, and J.K. Sahoo. Modeling temporal dependencies in data using a DBN-LSTM. In *International Conference on Data Science and Advanced Analytics*, pages 1–4. IEEE, 2015.
- [99] L. Wang. Three-dimensional convolutional restricted Boltzmann machine for human behavior recognition from RGB-D video. *EURASIP Journal on Image and Video Processing*, 2018(1):120, Nov 2018.
- [100] S. Wang, J. Xiang, Y. Zhong, and H. Tang. A data indicator-based deep belief networks to detect multiple faults in axial piston pumps.

Mechanical Systems and Signal Processing, 112:154–170, 2018.

- [101] M. Welling, S. Osindero, and G. Hinton. Learning sparse topographic representations with products of Student-t distributions. In *Advances in neural information processing systems*, pages 1383–1390, 2003.
- [102] M. Wong, B. Farooq, and G.A. Bilodeau. Discriminative conditional restricted Boltzmann machine for discrete choice and latent variable modelling. *Journal of choice modelling*, 29:152–168, 2018.
- [103] Z. Wu, E.S. Chng, and H. Li. Conditional restricted Boltzmann machine for voice conversion. In *China Summit and International Conference on Signal and Information Processing*, pages 104–108. IEEE, 2013.
- [104] B. Xiaojun and W. Haibo. Contractive slab and spike convolutional deep Boltzmann machine. *Neurocomputing*, 290:208–228, 2018.
- [105] L. Younes. Parametric inference for imperfectly observed gibbsian fields. *Probability theory and related fields*, 82(4):625–645, 1989.
- [106] A. Yuille. The convergence of contrastive divergences. In *Advances in neural information processing systems*, pages 1593–1600, 2005.
- [107] C.X. Zhang, J.S. Zhang, N.N. Ji, and G. Guo. Learning ensemble classifiers via restricted Boltzmann machines. *Pattern Recognition Letters*, 36:161–170, 2014.
- [108] N. Zhang, S. Ding, J. Zhang, and Y. Xue. An overview on restricted Boltzmann machines. *Neurocomputing*, 275:1186–1199, 2018.
- [109] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R.X. Gao. Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115:213–237, 2019.
- [110] J.-Y. Zhu, T. Park, P. Isola, and A. A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.



Rémi Souriau received his engineering and master diploma at the engineering school of Grenoble INP-ENSE3 (École Nationale Supérieure de l'énergie, l'eau et l'environnement) in signal processing, electronic and informatic. Now, Rémi Souriau is a PhD student specialized in computer science and applied mathematic at the university of Paris-Saclay. The title of his thesis is machine learning algorithms for stochastic dynamic models: application to control on deep-brain stimulation.



Before becoming a faculty member of université Paris-Saclay, Vincent Vigneron was working at the French Nuclear Center CEA as a computer science engineer. He received the Dipl.-Ing. degree from the ENSIMEV school, France in 1991, the PhD degree in applied mathematics and the Habilitation degree from université d'Evry, France, in 1997 and 2007 respectively. Actually, Vincent Vigneron is Associate Professor in the Electrical Engineering department. He is co-director of the IBISC/SIAM team. His main research topics concern the resolution of inverse problem with statistical learning, neural networks, blind source separation. Since 2014, he is dean of international affairs of université d'Evry.



Hsin Chen (陳新) received his B.S. and M.S. degrees in Electrical Engineering (EE) in 1996 and 1998, respectively, from the National Tsing Hua University (國立清華大學, NTHU) in Taiwan. During 2000-2004, he passed his PhD in Electronics at the Edinburgh University, UK. Afterwards, he joined the EE Dept. of the NTHU where he is now appointed as a full professor.



Jean Lerbet is a former student of École Nationale des Ponts et Chaussées (1984). He passed his PhD in theoretical mechanics in 1987, the Agrégation of mathematics in 1988, and habilitation in 2001. He is now a full Professor at UEVE since 2006. His scientific topics are Lie groups and differential geometry of vector bundles applied to mechanics (kinematics, dynamics, stability, ...)

Graphical Abstract

A Review on Generative Boltzmann Networks applied to Dynamic Systems

Rémi Souriau, Jean Lerbet, Hsin Chen, Vincent Vigneron

