ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

Randomized Machine Learning Approaches: Recent Developments and Challenges

Claudio Gallicchio[1], José D. Martín-Guerrero[2], Alessio Micheli[1], Emilio Soria-Olivas[2]

1 - Department of Computer Science, University of Pisa
Largo Bruno Pontecorvo 3 - 56127 Pisa, Italy
2 - Electronic Engineering Department, University of Valencia
Avgda. Universitat s/n, 46100 Burjassot,Valencia, Spain

Abstract.
Randomness has always been present in one or other form in Machine Learning (ML) models. The last few years have seen a change of role in the use of randomness, which is no longer a specific and accessory improvement in very particular aspects of a model, but the main theoretical basis that supports some ML methods, e.g., the well-known random forests.

In the Neural Network (NN) area, since its origins, randomness gave rise to a rich set of models, which have been recently exploited especially for efficiency aims. However, the bias induced by the use NN with random weights deserves further analysis, especially in the novel advances in the fields of deep NNs, dynamical systems (Recurrent NN), and NNs for learning in structured domains.

1 Introduction

A randomized approach employs a degree of randomness as part of its constructive strategy. As the introduction of randomness into computations can be an effective tool to design better methods in many fields of computer science (e.g. randomized algorithms), the interest of randomization in Machine Learning (ML) is wide as well. In this paper we present a brief overview of the recent advances and major challenges emerging in the field of randomized approaches for ML in general, with a focus on the impact on the area of NN modeling, in which we are recently witnessing an unprecedentedly seen increase in popularity of randomized methodologies.

The rest of this paper is organized as follows. Section 2 presents an overview of the use of randonness in ML methods. Section 3 surveys the literature on randomized NNs, presenting recent developments and open challenges. Finally, Section 4 introduces the papers accepted for the ESANN special session.

2 Randomness in Machine Learning

Randomness is intrinsic to ML, as the probabilistic approach is one of its most common interpretations. As a way of example, given the dependent variables $\mathbf{x}$ and the independent variable y, supervised learning can be understood as modeling the marginal probability distribution $P(y) = \sum_x P(y|\mathbf{x})$. The other main paradigm of ML, unsupervised learning, can also be approached proabalistically,

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

thus being the objective modeling the probability distribution $P(\mathbf{x})$ [1, 2, 3]. This kind of approaches have attracted special interest in the last few years; in particular, generative models [4] by means of the so-called Generative Adversarial Networks (GAN) [5]. Randomness can be present in ML at many different levels, usually enhancing performance or alleviating problems and difficulties of classical methods. In fact, many historic breakthroughs in the field of ML have been linked to randomness; mutation facts in genetic algorithms [6] or temperature in simulated annealing [7] are two representative examples. The current trend suggests that it will become an even more relevant issue in the years to come.

Section 2.1 briefly shows how random approaches may enhance different aspects of ML methods while Section 2.2 shows key examples of models that are intrinsically random; NNs are not included as they will be described in more detail in Section 3.

2.1 Randomness in specific aspects of Machine Learning

Data splitting. Overfitting is one of the most common problems that a ML model must deal with and eventually overcome. To assess this issue, different techniques of random data splitting have been proposed, such as hold-out, k-fold cross-validation or leave-one out.

Data generation. Randomness may also appear not only in splitting data sets already available but also in the generation of the data sets. Two relevant scenarios are those given by bootstrap and highly unbalanced classification problems:

1. Bootstrap is a technique that randomly samples (with replacement) a given data set in order to produce new sets based on the original one [8]. Although bootstrap is often used with small data sets, it can also improve bigger sets in terms of a better performance and stability, and a lower variance and overfitting.
2. Highly unbalaced classes is a common scenario in many real applications, e.g., fraud detection, rare diseases, screening of chemical compunds, etc. Besides, a problem of outlier detection could also be considered as a particular case of an unbalanced classification problem. Some procedures based on including randomness are usually considered in order to balance the representativeness of each class; the most simple approaches consist in randomly undersampling the majority class or randomly replicate patterns of the minority class; there are, nevertheless, more sophisticated approaches, like Synthetic Minority Over-sampling Technique (SMOTE) that also make use of randomness to produce new patterns [9].

Observation order. During on-line training, a random presentation of the data to the ML models usually enhances their performance. For instance, in a binary classification problem, it is inconvenient to present all the patterns belonging to one class and then, the patterns corresponding with the other class. Some models are more robust to the order of presentation than others, but in general

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

a random presentation is beneficial, especially in some kinds of models, e.g., NNs [10].

Hyperparameters. Most ML models are based on a search and eventual optimization of parameters. That search might be global or local; the latter is a paradigmatic example of a strong dependence with the initial values of the parameters, and hence, a smart intialiation would be helpful. When no a priori knowledge is available, a smart random initialization is the most common and sensible choice [10]. Once initialized, hyperparameters must be tuned; traditionally, that was accomplished by selecting a range defined by (random) maximum and minimum values for each hyperparameter, and then carrying out a grid search within the predefined range [11]. Although it already involved a slight randomness, [12] shows empirically and theoretically that fully-randomly chosen trials are far more efficient for hyperparameter optimization than trials on a grid.

Key elements of the models. The inclusion of randomness at the structure level is also possible. Two of the most remarkable examples are given by the random neurons of Restricted Boltzmann Machines (RBMs) [5], and the trees that made up a Random Forest (RF) [13], a model that will be briefly explained in Section 2.2. Other approaches incorporate randomness in the model design, as in the case of NNs with random weights (see Section 3). Moreover, tree-based models, such as RF and Extremely Randomized Trees (ERTs) [14] make use of randomness in their learning process.

2.2 Intrinsically random models

Apart from neural approaches that will be explained in Section 3, there are a number of models that are intrinsically random, i.e., randomness is no longer used to enhance a particular part of the model, as most of the examples provided in Section 2.1, but it is the basis of the model itself. For their particular relevance, three models have been selected to be briefly explained, namely RFs, ERTs and Conditional Random Fields (CRFs).

A RF is a tree-based ensemble learning method, which builds several Decision Tree (DT) models independently and then averages their individual predictions to compute a final prediction [13]. In a RF model, each single tree belonging to the ensemble is built from samples drawn randomly with replacement from the training set. Furthermore, when splitting a tree node during the building of a tree, the split that is chosen is the best split among a random subset of the input variables, selected without replacement, instead of picking the best split among all the input variables as that done in a single DT model. As a direct consequence of the randomization of the tree growing method combined with ensemble averaging, the variance of a RF model is reduced compared to that associated with a single non-random DT. Thus, several advantages are achieved, such as the improvement of the prediction performance, the reduction in the sensitivity to small changes in the training data and the control of overfitting, at the expense of a small increase in the bias, the loss of interpretability and a

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

higher computational cost. The main hyperparameters that need to be specified and optimized when building a RF model are the minimum number of samples required to split a tree node, the number of trees in the forest, and the number of input variables randomly selected to consider when looking for the best split.

ERTs are another tree-based ensemble learning method [14]. In ERT models, one further step of randomization is added when splitting a tree node by randomizing the choice of both input variable and cut-point. As in RF models, ERT models use a random subset of the input variables, but instead of searching for the best split-points, for each candidate input variable, its split-point is chosen fully at random (i.e. independently of the target variable), and then the best split among all the randomly-generated splits is picked. Thus, the variance and the computational complexity of an ERT model are reduced even more compared to those of a RF model, at the expense of a slightly greater bias. Another key difference between RF and ERT is that ERT models use the entire original training set to build each tree, instead of using bootstrap sampling, thus trying to minimize the bias. The main hyperparameters to adjust when building an ERT model are the same as in the case of RF.

CRFs are a class of model that has the appealing ability of processing neighboring samples. This peculiar characteristic can be used to extract information from the context, and hence, has a paramount relevance in the field of Natural Language Processing (NLP). In particular, CRFs can be considered as discriminative probabilistic graphical models [15]. They are used to encode known relationships between observations and construct consistent interpretations, as an alternative to Hidden Markov Models.

3 Randomized Neural Networks Approaches

The use of NN with random weights has its roots in the pioneering works on the perceptron [16, 17], originally conceived as composed of units organized in successive areas/layers, where the connections between the first two are random and those between the last two are adapted by the popular perceptron error-correction rule. The idea of leaving untrained some of the connections (or all of them) in a NN architecture has been successively revisited and analyzed in a number of early works (see e.g. [18, 19, 20]). However, this class of models, hereafter named *randomized NNs*, hit the literature only in recent years, raising a growing interest in the research community and taking the shape of a methodology of consolidated use [21, 22, 23].

Under an architectural point of view, a randomized NN can be typically thought as composed of two parts. The first part is an untrained hidden layer that serves to non-linearly embed the input into a high dimensional feature space (i.e. a basis expansion) in which the original problem is more likely to be linearly solved [24]. The second part is a trained readout component that combines the features in the hidden space (typically linearly) for the output computation.

In an attempt to give an abstract uniform formulation, in the basic case of static/vectorial data processing, a *feed-forward randomized NN* implements a

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

function that maps an input pattern $\mathbf{u} \in \mathbb{R}^{N_U}$ into the output $\mathbf{y} \in \mathbb{R}^{N_Y}$ by means of a basis expansion of the form[1]:

$$\mathbf{y} = \begin{bmatrix} \sum_{j=1}^{N_X} w_{1,j}^{out} f(\mathbf{w}_j \mathbf{u}) \\ \dots \\ \sum_{j=1}^{N_X} w_{N_Y,j}^{out} f(\mathbf{w}_j \mathbf{u}) \end{bmatrix} = \mathbf{W}^{out} f(\mathbf{W}\ \mathbf{u}). \tag{1}$$

where N_X is the dimension of the hidden space, f is the activation function of the hidden units (usually of sigmoid-type such as *tanh*), $\mathbf{W} = [\mathbf{w}_1; \dots; \mathbf{w}_{N_X}]$ is the weight matrix[2] of the hidden units whose elements are set randomly, and $\mathbf{W}^{out} = \{w_{i,j}^{out}\}_{j=1,\dots,N_X}^{i=1,\dots,N_Y}$ is the learned readout matrix.

The main reference models in this context are the Random Vector Functional-Link (RVFL) network [25] (in which often direct connections between the input and the output layer are considered) and the Extreme Learning Machine (ELM) [26, 27], which greatly contributed to the diffusion of the approach in recent years. Moreover, if the basis functions in eq. 1 are implemented as radial basis functions (RBFs), than RBF networks with random centers are obtained [28, 29].

In the case of temporal/sequential data processing, the untrained hidden layer is made up of recurrent units and provides a pool of random filters that contextualize each new input, acting as a state-based memory of the past input history. In this case, under the umbrella of Recurrent Neural Network (RNN) models [30], we speak of *recurrent randomized NNs* and the reference framework is given by the Reservoir Computing (RC) paradigm [31], which has been proposed in literature under several equivalent forms. In particular, the Liquid State Machine (LSM) [32] uses spiking units and has been originated from a biologically plausible perspective rooted in neuroscience, while the Echo State Network (ESN) [33, 34] adopts sigmoid units and achieved an impressive diffusion in the neurocomputing community. Another approach that falls within the same framework of computation is given by the Fractal Prediction Machine [35]. Adopting an ESN-like notation, in the case of recurrent randomized NNs, the basis expansion of eq. 1 becomes:

$$\begin{aligned} \mathbf{y}(t) = \begin{bmatrix} \sum_{j=1}^{N_X} w_{1,j}^{out} f(\mathbf{w}_j^{in} \mathbf{u}(t) + \hat{\mathbf{w}}_j \mathbf{x}(t-1)) \\ \dots \\ \sum_{j=1}^{N_X} w_{N_Y,j}^{out} f(\mathbf{w}_j^{in} \mathbf{u}(t) + \hat{\mathbf{w}}_j \mathbf{x}(t-1)) \end{bmatrix} &= \mathbf{W}^{out} f(\mathbf{W}^{in} \mathbf{u}(t) + \hat{\mathbf{W}} \mathbf{x}(t-1)) \\ &= \mathbf{W}^{out} \mathbf{x}(t) \end{aligned} \tag{2}$$

where, at each sequence step t, $\mathbf{u}(t)$ and $\mathbf{y}(t)$ respectively denote the input and the output, while $\mathbf{x}(t) \in \mathbb{R}^{N_X}$ is the *state* of the network, computed by the recurrent hidden *reservoir* layer. In this case, $\mathbf{W}^{in} = [\mathbf{w}_1^{in}; \dots; \mathbf{w}_{N_X}^{in}]$ is

[1] Bias terms are omitted for the ease of notation.
[2] The notation [;] is used to denote row-wise concatenation.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

the input-to-reservoir weight matrix and $\hat{\mathbf{W}} = [\hat{\mathbf{w}}_1; \ldots; \hat{\mathbf{w}}_{N_x}]$ is the recurrent weight matrix, both left untrained after random initialization. As can be seen in eq. 2, the reservoir implements a (discrete-time) dynamical system ruled by the relation $\mathbf{x}(t) = f(\mathbf{W}^{in}\mathbf{u}(t) + \hat{\mathbf{W}}\mathbf{x}(t-1))$, thus a primary role in the initialization of a recurrent randomized NN consists in ensuring stability of its state dynamics. This aspect is traditionally related to the spectral properties of $\hat{\mathbf{W}}$, according to the characterizations expressed by the Echo State Property (ESP) [34].

A common characterization of feed-forward and recurrent randomized NN is the extreme training efficiency, as learning is limited to the readout's parameters, i.e. to the weights in $\mathbf{W}^{out}$ in eq. 1 and 2. This involves a least-squares problem in the feature space of the hidden units' activations, typically approached by means of direct methods such as Moore-Penrose pseudoinverse or ridge-regression [31]. Fast training, ease of implementation (compared to standard fully trained architectures) and a vast number of successful applications reported in literature represent the striking points that make randomized NNs appealing, and a *de facto* state-of-the-art for efficiently learning in vectorial and temporal domains.

However, in the research community the topic of feed-forward randomized NNs (and especially of ELMs) has raised a debate, naturally related to the elegance of a full learning versus the introduction of randomness, contributing to motivate further research studies. On the other hand, in RNN modeling the use of randomness has been subject of more in-depth studies, with hyper-optimistic views mixed to different extents with constructive dialogues [36]. In this case, the analysis of randomized recurrent approaches has been especially motivated by the strict need for training efficiency and by aspects related to the characterization and control of the system dynamics, including stability, learnability [37], etc.

More in general, new analyses and developments in this field are demanded as a way of enhancing the understanding, awareness of use and performance of randomized NNs. Among the others, in particular, we identify the following major research lines.

Analysis of limitations. Characterizing the bias due to the use of fixed random weights is of primary importance in order to grasp the essential nature of randomized NNs and to identify on a solid basis their range of applicability, i.e. the class of learning tasks for which the training of the whole network's weights is effectively non necessary. Vice versa, a deeper understanding of the operation of NNs with random weights, through comparisons with fully trained ones, allows us to push forward our comprehension about the true merits of learning. While for feed-forward architectures a clear characterization of the inherent limitations due to the random weights setting is still needed, a considerable amount of works have addressed this problem for the case of recurrent models. Remarkably, it has been shown that initializing RNNs with small random weights results in a Markovian organization of their state dynamics [37, 38], which allows to discriminate among different input sequences on a suffix-based fashion even prior to learning. Such Markovian bias of RNNs [39, 40] has been directly linked to

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

the properties of ESNs in [41], and has shown to be of great help in characterizing their successful and unsuccessful applications. Other related works aim to investigate the way in which the random components are initialized, with promising insights obtained e.g. by imposing structural patterns to the recurrent weight matrix $\hat{\mathbf{W}}$ [42], by studying the network's dynamics [43, 44], or by using methodologies from the random matrix theory [45].

Implementations. Developments in software implementation of randomized NNs are recently gaining a growing interest, especially in relation to the possibility of realizing a synergy between the already extremely efficient training algorithms and hardware advances in high performance computing, e.g. GPUs. Such interplay is of particular relevance in the case of Big Data [46, 47]. Another recent trend is to explore physical realizations or RC networks via unconventional computation approaches, leading to photonic implementations of reservoirs that promise to open the way to ultra-fast learning (see e.g. [48, 49]).

Learning in Structured Domains. Along with the increased intrinsic complexity and expressive potential of data representations, learning in domains of complex structures such as trees, graphs and network data, often implies exploding computational costs, thereby the advantages of an efficient training methodology are in this case even more evident. Seminal contributions in this concern are represented by the works on TreeESN [50] and GraphESN [51], which generalize the RC paradigm respectively to trees and graphs, and that have opened the way to theoretical and experimental investigations that are still demanded.

Deep Learning. The ability of effectively learn feature representations at higher levels of abstraction and the impressive accuracy achieved in several complex real-world problems, are making deep NN more and more popular. Deep NN architectures generally consist of a hierarchy of many layers of non-linear units, where one of the main shortcomings is typically related to the fact that training algorithms are particularly time-demanding in this case. With the major goal of alleviating such computational burden, randomized approaches to deep NN design have been explored in recent literature. In the context of feed-forward models, networks based on stacking RVFL/ELM modules have been investigated e.g. in [52, 53, 54], showing competitive results in tasks related to image processing, handwritten character recognition, object detection and gesture recognition.

As concerns temporal data processing, a growing research interest is being devoted to deep RNN methodologies, enabling the potentiality of learning dynamical feature representations at multiple time-scales. In this context, networks composed of hierarchies of multiple ESNs have shown a great potentiality in applications to temporal benchmarks [55] and speech processing tasks [56]. Moreover, besides the applicative viewpoint, the analysis of stacked reservoir architectures proposed for the deepESN model [57, 58] allows to investigate the inherent role of layering in RNNs, also in terms of Lyapunov stability [59]. This type of analysis also opens the way to grounded approaches aiming at extending the advantages of deep learning to temporal data processing in an efficient way.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1.
Available from http://www.i6doc.com/en/.

4 Special Session Contributions

This Section briefly describes the main ideas of the papers that have been accepted in the ESANN special session, showing further significant instances of advancements in the area of randomized ML approaches.

In [60] Tiňo proposes a theoretical analysis of the asymptotic properties of Fisher memory in linear ESNs. The paper addresses the case in which the random recurrent weight matrix $\hat{\mathbf{W}}$ is constrained to the class of Wigner matrices. Interesting insights are found e.g. on the distribution of the weights in $\hat{\mathbf{W}}$ and on the pattern of input-to-reservoir coupling that maximizes the system memory.

In [61], Oneto et al. present a sound theoretical paper about a randomized algorithm based on data generating dependent prior and data dependent posterior Boltzmann distributions. The authors prove that the algorithm is Differentially Private (DP) and shows better generalization properties than the Gibbs (randomized) classifier associated to the same distributions. This allows, in turn, to produce enhanced DP-based generalization bounds.

Bacciu et al. focus on different approaches to realize preference learning when dealing with ELMs [62]. The main conclusion is that using weight sharing techniques to exploit symmetries in the task induces a smoother total preference relation, learned from different personal pairwise confrontations. The method generalizes very well also to external data taken under different experimental conditions.

Targeting the problem of reducing the efforts in data acquisition and labeling, in [63] Akusok et al. propose a comparison among three active learning methodologies based on ELMs. Experimental results on three benchmark datasets show promising results in comparison with literature approaches.

In [64], Wójcik and Kurdziel analyze the initialization of deep networks with random projection matrices. In particular, the analysis is limited to fully-connected Convolutional Neural Networks (CNNs) with pre-training. The results achieved in the paper suggest that random projection can be a feasible approach for CNNs, outperforming other common initialization methods.

References

[1] D. Barber. *Bayesian Reasoning and Machine Learning.* Cambridge UP, 2012.

[2] K.P. Murphy. *Machine Learning: A Probabilistic Perspective.* MIT Press, 2012.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[4] T. Jebara. *Machine Learning: Discriminative and Generative.* Springer, 2004.

[5] I. Goofellow, Y. Bengio, and A.Courville. *Deep Learning.* MIT Press, 2016.

[6] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[7] P.J. van Laarhoven and E.H. Aarts. *Simulated Annealing: Theory and Applications.* Springer, 1987.

[8] R.E. Schapire and Y. Freund. *Boosting: Foundations and Algorithms.* MIT Press, 2012.

[9] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[10] S. Haykin. *Neural Networks and Learning Machines.* Prentice Hall, 2009.

ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

[11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2008.

[12] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

[13] L. Breiman. Random forest. *Machine Learning*, 45(1):5–32, 2001.

[14] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[15] J. Lafferty, A. McCallum, and C.F.N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, 2001.

[16] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[17] M. Minsky and S. Papert. *Perceptrons*. MIT press, 1969.

[18] S.I. Gallant and D. Smith. Random cells: an idea whose time has come and gone... and come again. In *Proceedings of the IJCNN 1987*.

[19] W.F. Schmidt, M.A. Kraaijveld, and R.P.WDuin. Feedforward neural networks with random weights. In *Proceedings of the 11th IAPR International Conference on Pattern Recognition, 1992. Vol. II. Conference B: Pattern Recognition Methodology and Systems*, pages 1–4. IEEE, 1992.

[20] D.J. Albers, J.C. Sprott, and W.D. Dechert. Dynamical behavior of artificial neural networks with random weights. *Intelligent Engineering Systems Through Artificial Neural Networks*, 6:17–22, 1996.

[21] Y. Miche, B. Schrauwen, and A. Lendasse. Machine learning techniques based on random projections. In *Proceedings of ESANN*, pages 295–302, 2010.

[22] L. Zhang and P.N. Suganthan. A survey of randomized algorithms for training neural networks. *Information Sciences*, 364:146–155, 2016.

[23] S. Scardapane and D. Wang. Randomness in neural networks: An overview. *WIREs Data Mining and Knowledge Discovery*, 7(e1200), 2017.

[24] T.M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3):326–334, 1965.

[25] Y.-H. Pao and Y. Takefuji. Functional-link net computing: theory, system architecture, and functionalities. *Computer*, 25(5):76–79, 1992.

[26] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

[27] G. Huang, G.-B. Huang, S. Song, and K. You. Trends in extreme learning machines: A review. *Neural Networks*, 61:32–48, 2015.

[28] D.S. Broomhead and D. Lowe. Multi-variable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[29] D. Stosic, D. Stosic, C. Zanchettin, T. Ludermir, and B. Stosic. Qrnn: q -generalized random neural network. *IEEE Transactions on Neural Networks and Learning Systems*, 28(2):383–390, 2017.

[30] J. F. Kolen and S. C. Kremer. *A field guide to dynamical recurrent networks*. IEEE Press, 2001.

[31] M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[32] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[33] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[34] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks - with an erratum note. Technical report, GMD - German National Research Institute for Computer Science, Tech. Rep., 2001.

[35] P. Tino and G. Dorffner. Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning*, 45(2):187–217, 2001.

[36] D. Prokhorov. Echo state networks: appeal and challenges. In *Proceedings of IJCNN*, volume 3, pages 1463–1466. IEEE, 2005.

[37] B. Hammer and P. Tiňo. Recurrent neural networks with small weights implement definite memory machines. *Neural Computation*, 15(8):1897–1929, 2003.

[38] P. Tiňo and B. Hammer. Architectural bias in recurrent neural networks: Fractal analysis. *Neural Computation*, 15(8):1931–1957, 2003.

[39] P. Tiňo, M. Cernansky, and L. Benuskova. Markovian architectural bias of recurrent neural networks. *IEEE Transactions on Neural Networks*, 15(1):6–15, 2004.

[40] P. Tiňo, B. Hammer, and M. Bodén. Markovian bias of neural-based architectures with feedback connections. In *Perspectives of neural-symbolic integration*, pages 95–133. Springer, 2007.

[41] C. Gallicchio and A. Micheli. Architectural and markovian factors of echo state networks. *Neural Networks*, 24(5):440–456, 2011.

[42] I. Farkaš, R. Bosák, and P. Gergel'. Computational analysis of memory capacity in echo state networks. *Neural Networks*, 83:109–120, 2016.

[43] F.M. Bianchi, L. Livi, and C. Alippi. Investigating echo-state networks dynamics by means of recurrence analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 2016. In Press.

[44] L. Livi, F.M. Bianchi, and C. Alippi. Determination of the edge of criticality in echo state networks through fisher information maximization. *arXiv:1603.03685*, 2016.

[45] R. Couillet, G. Wainrib, H. Sevi, and H. T. Ali. The asymptotic performance of linear echo state neural networks. *Journal of Machine Learning Research*, 17(178):1–35, 2016.

[46] A. Akusok, K.-M. Björk, Y. Miche, and A. Lendasse. High-performance extreme learning machines: a complete toolbox for big data applications. *IEEE Access*, 3:1011–1025, 2015.

[47] L. Oneto, F. Bisio, E. Cambria, and D. Anguita. Statistical learning theory and elm for big social data analysis. *IEEE Computational Intelligence Magazine*, 11(3):45–55, 2016.

[48] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V.S. Udaltsov, Y.K. Chembo, and M. Jacquot. High-speed photonic reservoir computing using a time-delay-based architecture: Million words per second classification. *Physical Review X*, 7(1):011015, 2017.

[49] Z. Konkoli. *On Reservoir Computing: From Mathematical Foundations to Unconventional Applications*, pages 573–607. Springer International Publishing, 2017.

[50] C. Gallicchio and A. Micheli. Tree echo state networks. *Neurocomputing*, 101:319–337, 2013.

[51] C. Gallicchio and A. Micheli. Graph echo state networks. In *Proceedings of IJCNN*, pages 1–8, 2010.

[52] H. Cecotti. Deep random vector functional link network for handwritten character recognition. In *IJCNN 2016*, pages 3628–3633.

[53] W. Yu, F. Zhuang, Q. He, and Z. Shi. Learning deep representations via extreme learning machines. *Neurocomputing*, 149:308–315, 2015.

[54] J. Tang, C. Deng, and G.-B. Huang. Extreme learning machine for multilayer perceptron. *IEEE transactions on neural networks and learning systems*, 27(4):809–821, 2016.

[55] H. Jaeger. Discovering multiscale dynamical features with hierarchical echo state networks. Technical report, Jacobs University Bremen, 2007.

[56] F. Triefenbach, A. Jalalvand, K. Demuynck, and J.-P. Martens. Acoustic modeling with hierarchical reservoirs. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(11):2439–2450, 2013.

[57] C. Gallicchio, A. Micheli, and L. Pedrelli. Deep reservoir computing: A critical experimental analysis. *Neurocomputing*, 2016. (Accepted).

[58] C. Gallicchio and A. Micheli. Deep reservoir computing: A critical analysis. In *Proceedings of ESANN*, pages 497–502, 2016.

[59] C. Gallicchio, A. Micheli, and L. Silvestri. Local Lyapunov Exponents of Deep RNN. In *Proceedings of ESANN*, 2017.

[60] T. Tiňo. Fisher memory of linear Wigner Echo State Networks. In *Proceedings of ESANN*, 2017.

[61] L. Oneto, S. Ridella, and Anguita D. Generalization performances of randomized classifiers and algorithms built on data dependent distributions. In *Proceedings of ESANN*, 2017.

[62] D. Bacciu, M. Colombo, D. Morelli, and Plans D. Elm preference learning for physiological data. In *Proceedings of ESANN*, 2017.

[63] A. Akusok, E. Eirola, Y. Miche, A. Gritsenko, and A. Lendasse. Advanced query strategies for active learning with extreme learning machines. In *Proceedings of ESANN*, 2017.

[64] P. I. Wójcik and M. Kurdziel. Random projection initialization for deep neural networks. In *Proceedings of ESANN*, 2017.