Relations between Entropy and Accuracy trends in Complex Artificial Neural Networks

Lucia Cavallaro[1], Marco Grassia[2], Giacomo Fiumara[3], Giuseppe Mangioni[2], Pasquale De Meo[4], Vincenza Carchiolo[5], Ovidiu Bagdasar[1], and Antonio Liotta[6]

[1] University of Derby, Kedleston Road, Derby, DE22 1GB, UK
{l.cavallaro, o.bagdasar}@derby.ac.uk
[2] Università degli Studi di Catania, Dipartimento di Ingegneria Elettrica, Elettronica e Informatica, Italy
{marco.grassia, giuseppe.mangioni}@unict.it
[3] Università degli Studi di Messina, MIFT Department, Messina, 98166, Italy
g.fiumara@unime.it
[4] Università degli Studi di Messina, Polo Universitario Annunziata, Messina, 98122, Italy
p.demeo@unime.it
[5] Università degli Studi di Catania, Dipartimento di Matematica e Informatica, Italy
vincenza.carchiolo@unict.it
[6] Free University of Bozen-Bolzano, Faculty of Computer Science, Italy
antonio.liotta@unibz.it

Abstract. Training Artificial Neural Networks (ANNs) is a non-trivial task. In the last years, there has been a growing interest in the academic community in understanding how those structures work and what strategies can be adopted to improve the efficiency of the trained models. Thus, the novel approach proposed in this paper is the inclusion of the entropy metric to analyse the training process. Herein, indeed, an investigation on the accuracy computation process in relation to the entropy of the intra-layers' weights of multilayer perceptron (MLP) networks is proposed. From the analysis conducted on two well-known datasets with several configurations of the ANNs, we discovered that there is a connection between those two metrics (*i.e.*, accuracy and entropy). These promising results can be helpful in defining, in the future, new criteria to evaluate the training process goodness in real-time by optimising it and allow faster detection of its trend.

Keywords: Complex Artificial Neural Networks, Network Science, Graph Theory, Entropy, Accuracy

1 Introduction

Deep Learning [1], the sub-branch of Machine Learning that uses *deep* models, is now pervasive in many fields and has been successfully employed to approach many complex tasks [2–4]. The motivations behind its popularity are well known:

the large availability of data and computational power, and the capability of the trained models to generalise to unseen data, often with super-human performances. However, the training of Deep Learning models is far from trivial. In fact, various attempts are often required to find a combination of the hyperparameters (*e.g.*, the number of model layers, the number of neurons, the number of training epochs, etc.) such that the model is able to learn from the data and yet does not *overfit* (*i.e.*, the model performs a good generalisation to new data). Some works in the literature address this problem and aim to simplify the training phase, for instance by using Genetic Algorithms to select the model topology [5–7]. On the other hand, other works aim at accelerating the training by using sparse Artificial Neural Network models: the first work in this direction is by *Mocanu et al.* [8], who proposes an evolutionary algorithm that takes inspiration from biological neural networks, but others followed [9, 10].

In this preliminary work we aim at analysing the weights of fully connected Artificial Neural Network models to better understand their training using Network Science tools. In fact, whereas the application of Deep Learning algorithms to Complex Networks has quite a long history (from the seminal works of *Scarselli et al.* [11] to the rise of Geometric Deep Learning [12]), the vice-versa, (*i.e.*, the application of Network Science techniques to get insights from Deep Learning models) is still largely unexplored.

Herein, in particular, we borrow from Network Science the entropy metric, nowadays widely used to measure the complexity of complex networks such as in the works of *Gómez-Gardeñes et al.* [13] and *Bianconi et al.* [14,15], even though its main applications has been in information theory field [16, 17].

Our intuition has been to combine the analysis of the accuracy on the validation set during the training process with the intra-layer entropy weights to detect whether there is a relation between those two metrics. In order to do so, we tested two well-known datasets (*i.e.*, Lung cancer and COIL20) and we set up multilayer perceptron networks (MLP) in supervised training fashion in which we varied the number of hidden layers and neurons per each hidden layers from which we, then, discarded the underfitting and overfitting configurations. Herein, for the sake of brevity, only the slow and fast learning have been reported.

The results obtained confirmed our intuition. Indeed, what we discovered is that the entropy is growing during the transitory phase of the accuracy and starts to drop sharply once the accuracy became stable. Thus, there is a connection between those two metrics. This outcome paves the way in devising a new metrics that, combined with the classical Machine Learning techniques, may be proposed as a powerful tool to speed up training process of ANNs.

The paper is organised as follows. Sect. 2 describes all the tools required to understand the research herein presented. In particular, it is structured in theoretical background (Sect. 2.1), description of the datasets used (Sect. 2.2), and methodology followed (Sect. 2.3). Next, in Sect. 3, the results obtained are shown and commented jointly with the conclusions.

2 Materials and Methods

In this section all the basic notions and definitions used in this work are showed jointly with a concise description of the datasets used in our experiments along with the methodology followed to pursue the experiments.

2.1 Background

In this paper, we deal with *Artificial Neural Networks* (ANNs) in their classical configuration, which means that they are fully connected networks.

An ANN is a computational learning system that uses the structure and the functions of a network with the goal of understanding and converting a certain data input of one form into a desired output, usually in another form. This concept has been originated from several studies on human biology and the way neurons of the human brain work together. In particular, we focus our attention on a *multilayer perceptron* (MLP) supervised model.

MLP is a *feed-forward* ANN composed by several hidden layers, forming a Deep Network. The name of feed-forward derives from the structure of those kind of models because information flows through the function being evaluated from x, through the intermediate computations used to define a function f, and finally to the output y. In addition, there are no feedback connections in which outputs of the model are fed back into themselves. It is, however, possible to have feedback connections in a feed-forward neural network; in such a case, those networks called recurrent neural networks [18]. Feed-forward neural network are used for classification and regression, as well as for pattern encoding [19]. For the sake of brevity, herein, we consider only classification problems.

Supervised learning involves observing several samples of a given dataset, which will be divided into *training* and *test* samples. While the former is used to train the neural network, the latter works as a litmus test, as it is compared with the ANN predictions [1,9,18]. Indeed, thanks to the training set, the supervised learning models are able to teach models to obtain the desired output to the ANN. Thus, the training dataset is composed by inputs and correct outputs that allow the model to learn over time (*i.e.*, epochs). During the training process, hence, the algorithm can validate its accuracy through the loss function, adjusting until the error has been sufficiently minimised. Other types of learning strategies are unsupervised learning and semi-supervised learning.

Generally speaking, one of the most popular evaluation metrics to verify the goodness of the training process is performed by computing the accuracy, which measure how the predicted values are close to the expected value. It is computed as follows:

$$\text{Accuracy} = \frac{\text{pred}_{\text{correct}}}{\text{pred}_{\text{tot}}} \tag{1}$$

where $\text{pred}_{\text{correct}}$ is the number of correct predictions and pred_{tot} is the total number of predictions.

Due to the strict connection between the notion of ANN and networks, those systems can be seen as weighted graphs. A *weighted graph*, denoted as G, is a

triplet $G = \langle V, E, W \rangle$ in which V is the set of *vertices* (or nodes), $E = \{\langle i, j \rangle : i \in V \wedge j \in V\}$ is the set of *edges* (or links) and $W : E \to \mathbb{R}^+$ is a function that maps an edge $\langle i, j \rangle$ onto a non-negative real number w_{ij}. In addition, a *signed* graph is a weighted graph where if $w_{ij} \in \mathbb{R}$ (*i.e.*, weights can also be negative). Lastly, if a graph's weights are equal only to zero or one; *i.e.*, $w_{ij} = 1$, $w_{ij} = 0$, then it is called *unweighted graph*.

With those notions in mind, we employ the Shannon's entropy S to get an insight about the distributions. It is defined as follows:

$$S = -\sum_{i \in W} p_i \log p_i \tag{2}$$

where

$$p_i = \frac{w_i}{\sum_{j \in W} w_j}$$

and i represents the index of the i-th weight, whereas j is the index of all the elements in the set. Note that $i, j \in E_l$, then both are in the edges set of the specific layer l.

This formulation of *entropy* is a bit counter-intuitive: one would expect that the higher the entropy, the more uniform the distribution. The reason is that if weights are evenly distributed the p_i will all have different values, while if all weights are equal all the p_i are also equal. For instance, if there are only two weights $w_1 = 1$ and $w_2 = 2$, then $p_1 = 1/3$, $p_2 = 2/3$. On the other hand, if $w_1 = w_2 = 1$, then $p_1 = p_2 = 1/2$. After translating the weights into the p_is, this formulation should get more intuitive.

In addition, the maximum value (the upper bound) of entropy is equal to $\log N$, where N represents the number of elements in the set. In fact:

$$p_i = \frac{1}{N} \quad \forall i$$

and, thus,

$$S = -\sum p_i \log p_i = -N \cdot \left(\frac{1}{N}\right) \cdot \log\left(\frac{1}{N}\right) = \log N$$

An introductory discussion on the concept of information entropy is contained in [20], where entropy (and mutual information) are used to detect the key nodes of a complex network. More details on the use of entropy in complex networks can be found on the work of *Alves et al.* [21], in which the authors analysed the evolution over time of this metric in the context of the global value chain of the worldwide production networks. For a thorough description of the various definitions of the entropy metrics in complex network we refer the interested reader to [22].

2.2 Dataset

To conduct the experiments, two publicly available online[1] well-known datasets have been used.

The first one is the Lung Cancer[2] [23] that is a biological dataset composed by features on lung cancer to train the ANN to detect this specific kind of cancer, which contains 3312 gene data obtained from 17 people with normal lungs and 186 lung cancer patients that is classified into 5 classes: (i) Adenocarcinomas (139 patients), (ii) Squamous Cell Lung Carcinomas (21 patients), (iii) Pulmonary Carcinoids (20 patients), (iv) Small Cell Lung Carcinomas (6 patients), and (v) Normal Lung (17 people).

The second one is COIL20[3] [24, 25] that is an image dataset used to train ANNs to detect 20 different objects. The images of each object were taken five degrees apart as the object is rotated on a turntable and each object has 72 images. The size of each image is 32×32 pixels, with 256 grey levels per pixel. Thus, each one is represented by a 1024-dimensional vector [9].

The number of instances, input features and output classes of both datasets are summarised in Table 1.

Name	**Instances** (#)	**Input Features** (#)	**Output Classes** (#)	**Source**
Lung Cancer	203	3,312	5	[23]
COIL20	1440	1024	20	[24, 25]

Table 1: Dataset structures description. From left: dataset name; number of instances, of input features and of output classes.

2.3 Methodology

In this paper we want to address to what extent there are similarities between the behaviours of accuracy (on the validation set) and entropy in the artificial networks under scrutiny. In this respect, we tested different models with a variable number of hidden layers (from 2 to 3) and neurons for each layer (*i.e.*, 2, 5, 10, 50, 100, 250, 500) from which we detected the best models for the datasets under scrutiny and discarded the underfit and overfit scenarios. For the sake of brevity, the four most significant configurations obtained, which are summarised in Table 2, are herein reported. Lastly, 100 epochs for the training have been considered.

During the training, we have, then, computed the entropy of weights of the inter-layer links of the nodes (*i.e.*, neurons) of each layer. Note that the entropy

[1] http://featureselection.asu.edu/

[2] https://sites.google.com/site/feipingnie/file/https://jundongl.github.io/scikit-feature/datasets.html

[3] http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html

Dataset	Hidden Layers (#)	Neurons (#)	Learning Speed
Lung Cancer	L2	N10	Slow
Lung Cancer	L2	N50	Fast
Lung Cancer	L3	N10	Slow
Lung Cancer	L3	N50	Fast
COIL20	L2	N10	Slow
COIL20	L2	N50	Fast
COIL20	L3	N10	Slow
COIL20	L3	N50	Fast

Table 2: Configurations of hidden layer and neurons per each one. From left: dataset name, number of hidden layers, number of neurons per hidden layer and learning speed.

has been computed for the absolute values of the weights, since in an Artificial Neural Network weights can also be negative and entropy is defined only for non-negative values. Finally, the analysis for each layer to layers pair is reported aggregated (*i.e.*, the analysis of the whole model).

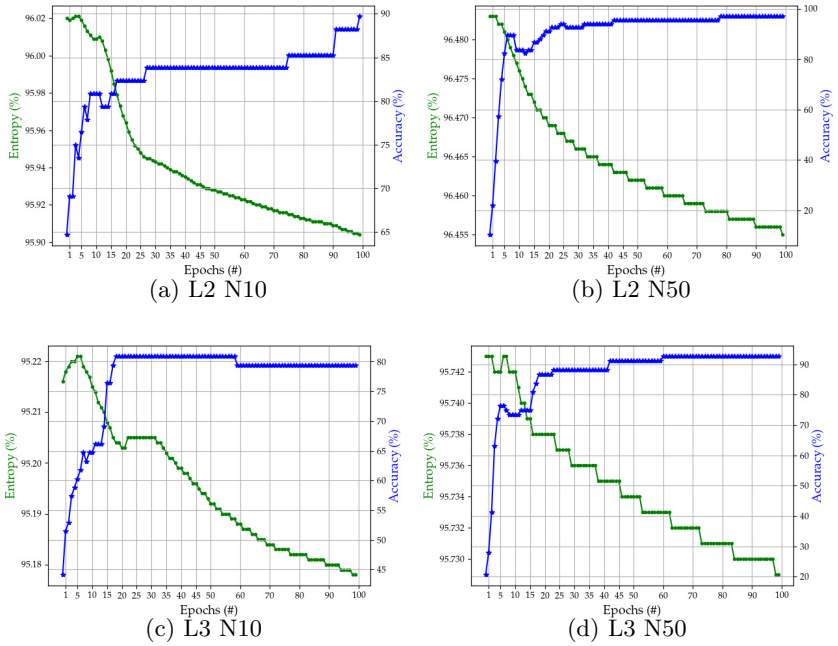
3 Discussion and Conclusions

In this section the results obtained from our analysis are commented. In Figures 1 and 2 are shown the comparative trend of accuracy and entropy for Lung and COIL20 datasets, respectively.

Note that to normalise the entropy outcomes, its values have been reported as the ratio between the current entropy value and its maximum theoretical value (*i.e.*, $\log N$, that is the upper bound).

First of all, as expected, in the slow learning scenarios (Figures 1a, 1c, 2a, and 2c) the highest accuracy attained does not reach a plateau greater than 90%. It is fascinating to notice that a higher number of hidden layers not necessarily leads to a more precise training. In Fig. 2d, whereas the configuration selected has three hidden layers and 50 neurons per layer, for instance, the accuracy reached at the last epoch (and all the previous values before that one as well) is lower than the one reported in Fig. 2b in which there is the same number of neurons, but only two hidden layers.

What emerges from our analysis is that the more stable the accuracy, the higher the entropy decrease. This aspect becomes clearer when Fig. 2a and Fig. 2c are compared with the other plots. Indeed, in those two figures, which represent specifically the slow learning trend in COIL20 dataset, the accuracy is still increasing without reaching its plateau; thus, the entropy is increasing as well. On the other hand, once the accuracy is stable (*i.e.*, no more improvements in terms of performances are expected at this step of the training process) the entropy starts to decrease. For instance, in Fig. 1d after the 15th epoch, the trend of the accuracy tends to stabilise and, at the same time, the entropy starts to sharply drop.



(a) L2 N10 (b) L2 N50

(c) L3 N10 (d) L3 N50

Fig. 1: Comparative analysis of Entropy (dotted markers in green) and Accuracy (star markers in blue) on Lung Dataset

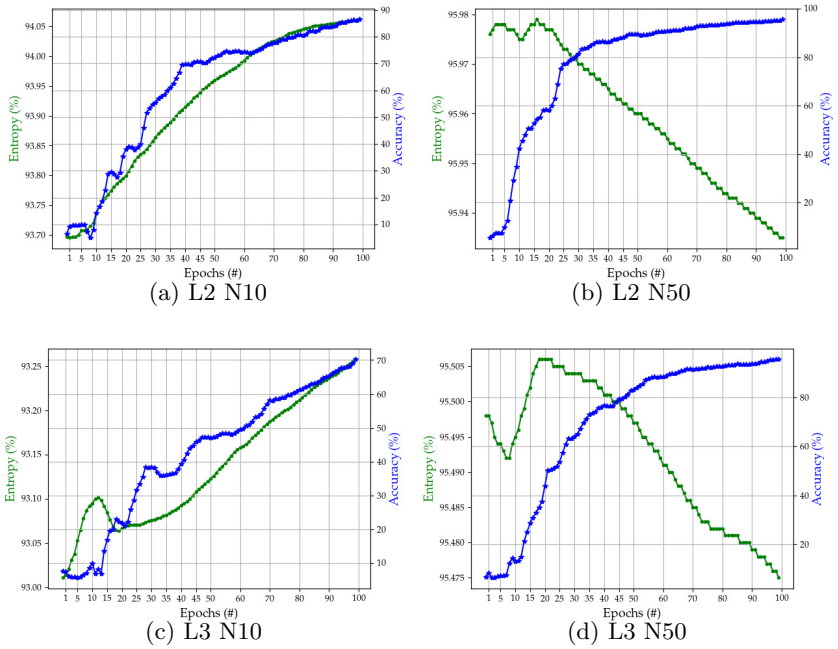
Another interesting aspect to pinpoint is that the fluctuations experienced by the accuracy are somehow followed by the entropy. Of course, all the variations in the entropy are significantly less pronounced respect to the accuracy trend; that is, the entropy ratio varies between 93% and 97% whereas the accuracy starts from 0% up to 95%. Nonetheless, the trend is still remarkable.

What we infer from those analyses is that the entropy variations are somehow related with the accuracy. This is an aspect that can pave the way to a branch of studies on this direction.

One of them can be to extend this work considering a wider range of datasets and configurations to detect whether there are specific datatets on which this approach is preferred. Thus, it would be also matter of study varying not only the number of hidden layers, but also the number of neurons per hidden layer that can be also unbalanced (instead of keeping it constant for all the hidden layers under scrutiny).

Other possibilities can be the use of different learning strategies (*i.e.*, unsupervised, semi-supervised).

Another application domain to involve the use of entropy for a finer tuning of the training process can be in Sparse ANNs. Indeed, we are also interested in understanding how does entropy behave as the network becomes sparser. Indeed, the advantage of combining the entropy with the accuracy could lead to



(a) L2 N10 (b) L2 N50

(c) L3 N10 (d) L3 N50

Fig. 2: Comparative analysis of Entropy (dotted markers in green) and Accuracy (star markers in blue) on COIL20 Dataset

more precise metric, which could be used, in turn, to develop a better and more powerful tool able to define a suitable stop condition for ANNs training. This could lead to a reduction in terms of the overall number of epochs needed to train the network with a satisfying accuracy and, thus, have a lower computational cost.

References

1. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.
2. W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.
3. J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, 2021. (Accelerated article preview).

4. M. Grassia, M. De Domenico, and G. Mangioni, "Machine learning dismantling and early-warning signals of disintegration in complex systems," *Nature Communications*, vol. 12, p. 5190, Aug 2021.
5. S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, "Optimizing deep learning hyper-parameters through an evolutionary algorithm," in *Proceedings of the workshop on machine learning in high-performance computing environments*, pp. 1–5, 2015.
6. F. Assunção, N. Lourenço, P. Machado, and B. Ribeiro, "Evolving the topology of large scale deep neural networks," in *European Conference on Genetic Programming*, pp. 19–34, Springer, 2018.
7. F. Mattioli, D. Caetano, A. Cardoso, E. Naves, and E. Lamounier, "An experiment on the use of genetic algorithms for topology selection in deep learning," *Journal of Electrical and Computer Engineering*, vol. 2019, 2019.
8. D. C. Mocanu, E. Mocanu, P. Stone, P. H. Nguyen, M. Gibescu, and A. Liotta, "Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science," *Nature communications*, vol. 9, no. 1, pp. 1–12, 2018.
9. L. Cavallaro, O. Bagdasar, P. De Meo, G. Fiumara, and A. Liotta, "Artificial neural networks training acceleration through network science strategies," *Soft Computing*, vol. 24, no. 23, pp. 17787–17795, 2020.
10. S. Liu, I. Ni'mah, V. Menkovski, D. C. Mocanu, and M. Pechenizkiy, "Efficient and effective training of sparse recurrent neural networks," *Neural Computing and Applications*, vol. 33, pp. 9625–9636, Aug 2021.
11. F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.
12. M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, p. 18–42, Jul 2017.
13. J. Gómez-Gardeñes and V. Latora, "Entropy rate of diffusion processes on complex networks," *Phys. Rev. E*, vol. 78, p. 065102, Dec 2008.
14. G. Bianconi, A. C. C. Coolen, and C. J. Perez Vicente, "Entropies of complex networks with hierarchically constrained topologies," *Phys. Rev. E*, vol. 78, p. 016114, Jul 2008.
15. G. Bianconi, "Entropy of network ensembles," *Phys. Rev. E*, vol. 79, p. 036114, Mar 2009.
16. C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.
17. T. Cover and J. Thomas, "Elements of information theory,(pp 33-36) john wiley and sons," *Inc, NY*, 1991.
18. I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. http://www.deeplearningbook.org.
19. M. Gori, "Chapter 5 - deep architectures," in *Machine Learning* (M. Gori, ed.), pp. 236–338, Morgan Kaufmann, 2018.
20. Y. Li, W. Cai, Y. Li, and X. Du, "Key node ranking in complex networks: A novel entropy and mutual information-based approach," *Entropy*, vol. 22, no. 1, 2020.
21. L. G. A. Alves, G. Mangioni, F. A. Rodrigues, P. Panzarasa, and Y. Moreno, "Unfolding the complexity of the global value chain: Strength and entropy in the single-layer, multiplex, and multi-layer international trade networks," *Entropy*, vol. 20, no. 12, 2018.
22. Y. M. Omar and P. Plapper, "A survey of information entropy metrics for complex networks," *Entropy*, vol. 22, no. 12, 2020.

23. A. Bhattacharjee, W. G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, *et al.*, "Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses," *Proceedings of the National Academy of Sciences*, vol. 98, no. 24, pp. 13790–13795, 2001.
24. D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1548–1560, 2010.
25. D. Cai, X. He, and J. Han, "Speed up kernel discriminant analysis," *The VLDB Journal*, vol. 20, no. 1, pp. 21–33, 2011.