From Statistical Physics to Algorithms in Deep Neural Systems

Abstract

Nowadays it is pretty common for people to talk about Artificial Intelligence. Contrarily from what it is imaginable, AI research is not focused on the realization of "self-thinking" machines: instead, it involves the study of any algorithm, machine or more general, artificial agent being able to perceive the environment and to react in a "smart" way, i.e. maximizing the chance of successfully achieving its own goals (like, for example, correctly classifying an object). In particular, great interest is shown towards Artificial Neural Networks (ANNs) modeling, which are biological brain-inspired. The research around ANNs started about 80 years ago, but they received huge consideration by the research community just in the last few decades. Empirically, their learning capability for non-trivial tasks has been acknowledged since 1990; however there was the lack of powerful simulation tools. In the last years, particularly thanks to the use of Graphical Processing Units (GPUs) to exploit most of the computation (and recently the use of TPUs), almost everyone owns a device able to simulate an ANN. For this reason, in the last few years artificial neural networks were challenged to solve more and more complex tasks, being able, for example, to correctly classify images in 1000 classes, to win over the world champion of Go (with the machine named AlphaGo), to understand the human speech etc. In order to accomplish all of these tasks, their size and complexity scaled-up: nowadays it is common to train ANNs having more than hundred millions of parameters. As long as the learning techniques are able to "somewhat" train the network, the community is in general not very interested in understanding all the learning dynamics inside the network.

I lived my PhD work like a journey, starting from the simplest possible model, the perceptron. Such an ANN architecture shows similarities to the Ising model; hence, analytical tools from the mechanical statistics can be borrowed to analyze the version (solution) space of the problem aimed to be solved. From the theory, we know that local research tools are typically destined to fail in the learning dynamics

for the binary perceptron (a model with binary parameters). However, recently it has been shown the existence of dense sub-dominant clusters of solutions for the learning problem. Exploiting this property of the version space, a model to solve the binary perceptron problem has been designed, in which the synaptic couplings (the parameters to be learned) are considered being stochastic, according to a given distribution. It was observed that this learning dynamics, even though relying on a local-research algorithm, still finds solutions lying in the dense cluster. Such a model was extended to more complex ANN architectures. Sadly, without some extra heuristics borrowed from the common knowledge of deep learning, it seems not to achieve state-of-the-art performances and it is still matter of studies. The analysis of the version space for the perceptron model showed some geometrical characteristics, the same for more complex architectures is however not duable because of the complexity of the network itself. For this reason, an empirical exploration algorithm has been designed, aiming to investigate the properties of the version space for the Tree Committee Machine (TCM). Even though from the theory the version space should not be connected in the general case, it was empirically observed that the algorithmically-accessible version subspace is connected. Other works showed the same behavior for more complex architectures: this could be a hint for understanding why the current learning algorithms still work on larger networks. Finally, the problem of parameter reduction for deep networks (deep refers to the great number of layers) has been explored: do we really need huge architectures to solve the learning tasks? We have showed that a massive parameter reduction without performance loss is possible, designing a proper regularization term which allows us to prune parameters from the network which are not relevant to solve the learning problem. This is done by designing an additional regularization term, which is a biologically-inspired penalty for non-relevant synaptic couplings in our model.