Neural Architecture Growth for Frugal Learning
Starting tiny, growing where needed

Keywords: neural networks, low computational complexity, differential calculus, optimization, expressive power

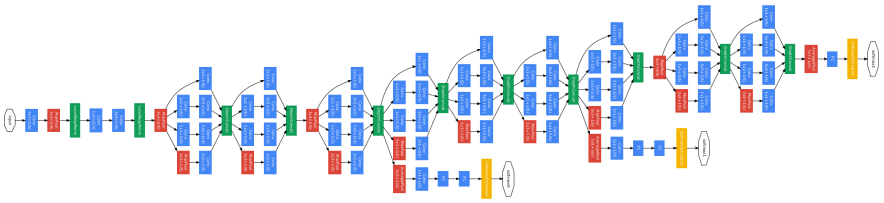
Research lab: INRIA TAU team
Location: LISN (building 660 "Digiteo", at Université Paris-Saclay)
Supervision team: Guillaume Charpiat (webpage), (guillaume.charpiat@inria.fr) & Sylvain Chevallier & Stella Douka & Manon Verbockhaven
Funding: European project MANOLO

Punchline: We aim at training tiny networks with a flexible architecture that adapts and grows on the fly while training.

Thematic context: Deep learning has shown impressive, highly-mediatized results on various applications (Go game, StarCraft, translation, object detection in images, high-resolution image generation, text generation...), obtained at the cost of training huge neural network architectures, which therefore also takes time and money (for instance, GPT-3 has 10^{11} parameters and might cost millions of dollars to be trained), both at training and exploitation times.

Frugal learning, on the opposite, consists in training with as few samples or as little computational power (*Green AI*) as possible. We will focus on the latter here.

Large vs. small: One advantage of having many neurons per layer is that it is known (experimentally and theoretically [2]) to facilitate optimization during training, thus yielding better results. However, trained neural networks show high internal redundancy, and various techniques have been developped to squeeze them into smaller networks with comparable accuracy. For instance [3] manages to divide by 100 the number of neurons in order to run online object recognition in videos on a smartphone. On the other extreme, training and applying "tiny" models (with "only" 10^5 parameters) will be much faster (as each test of the network is of much smaller complexity) but they might suffer from a lack of expressivity, preventing them from fitting data accurately.



Where is a supplementary block most needed to increase the performance?

Change of paradigm: We propose to start with the simplest possible neural network (i.e., one neuron, a.k.a., linear regression) and make the network grow according to the information brought by the backpropagation pass. Such information can indeed be used to go beyond usual limitations in small network training, to explicitely tackle potential optimization and expressivity issues. To do this, we locate precisely, while training, the learning bottlenecks of a neural network, i.e., the layers which lack expressive power, in order to boost them by adding neurons or new

layers appropriately where and when it is needed. With such an iterative architecture refinement scheme, important gains in architecture search (auto-DL) are expected. Indeed current Neural Architecture Search methods consist in trying many different architectures, while with our approach a single training progressively grows the architecture.

Work done on this topic in our team so far:

A PhD student (Manon Verbockhaven) has already formalized mathematically the concepts required to spot and fix expressivity bottlenecks. She has also implemented layer growth of fixed convolutional and fully-connected architectures. More details concerning this methodology can be found in [1].

A master 2 intern (Barbara Hajdarevic) has been extending this work to layer graph growth, i.e. adding new layers to the computational graph. This is a necessary step to unleash the power of the approach and to check how it performs.

Possible research tracks can cover many different domains, depending on the candidate's skills and preferences, ranging from functional analysis, statistical tests, and information theory, to optimization, hyper-optimization and coding. For instance:

- **Computing optimal neurons to be added to a given layer.** Find better optimization schemes to better deal with non-linearity of activation functions.
- **Strategy for neuron/layer addition.** Currently, to decide whether to add the best possible neuron (or layer), we consider a trade-off betwen performance gain and additional computational complexity, inspired from Information Theory (Minimum Description Length paradigm, Kolmogorov complexity), as well as a statistical significance criterion. Complementary approaches from, e.g., Reinforcement Learning, could be considered.
- **Algorithmics for expressivity bottlenecks localization.** Backpropagation properties might be better exploited to design new algorithms to identify expressivity bottlenecks faster.
- **Training stability.** Optimizing neural networks with both new parameters (neurons recently added) and old ones (old neurons that had more time to converge) can be challenging, as different learning rates might be needed at the same time.
- **Extension to attentation mechanisms** (Transformers, etc.)

Possible applications:

- with industrial and academic partners of the European project MANOLO: search for small, efficient neural networks that can run in low-ressource environnements
- learning physics (dynamical systems such as fluid mechanics): automatically adapting the architecture and the mesh resolution (ANR SPEED)
- more generally, AI for Green (agro-ecology, etc.)

Requirements:

- Machine Learning.
- Mathematics: functional analysis, differential calculus, statistics.
- Coding skills. Programming will be done mostly in python with the PyTorch deep learning platform.

State of the art and other methods:

The main approaches to full neural network architecture optimization are based on automatic hyper-parameter tuning (auto-DL), but they are computationally extremely demanding, as they run many tries on many architecture variations.
Some approaches incorporate architecture flexibility in their design [4]; they can however only suppress connexions between existing blocks, or suppress blocks, but not add new ones.

Only few approaches try to grow architectures (such as [6]); unfortunately they are most often based on ad-hoc criteria which are not mathematically justified.

References:

[1] Verbockhaven, M. and Charpiat, G. Growing Tiny Networks: Spotting Expressivity Bottlenecks and Fixing Them Optimally.

[2] Scaling description of generalization with number of parameters in deep learning; Mario Geiger, Arthur Jacot, Stefano Spigler, Franck Gabriel, Levent Sagun, Stéphane d'Ascoli, Giulio Biroli, Clément Hongler, Matthieu Wyart, Journal of Statistical Mechanics: Theory and Experiment 2020

[3] Bayesian Compression for Deep Learning; Christos Louizos, Karen Ullrich, Max Welling, NIPS 2017

[4] DARTS: Differentiable Architecture Search; Hanxiao Liu, Karen Simonyan, Yiming Yang, ICLR 2019

[5] Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. On the expressive power of deep neural networks. In Proceedings of the 34th International Conference on Machine Learning, pp. 2847-2854, 2017.

[6] Maile, K., Rachelson, E., Luga, H., and Wilson, D. G. When, where, and how to add new neurons to ANNs. In First Conference on Automated Machine Learning (Main Track), 2022.