Proposal for M2 Research Internshipfor ENS Lyon for 2025

Complexity Theory for Models of Deep learning. Complexity Theory for Models of Very Deep learning.

Olivier Bournez

Keywords: Complexity Theory. Computations over the reals. Deep Learning Models. Very Deep Learning Models. Neural Ordinary Differential Equations. Continuous time models of computation. Analog machines and models. Ordinary Differential Equations. Computability

Superviser: Olivier Bournez.

Administrative Position: Professor (of Computer Science) at Ecole Polytechnique.

Location: Laboratoire d'Informatique de l'X, LIX, Ecole Polytechnique , 91128 Palaiseau Cedex

Phone: +33 (0)1 77 57 80 78

Email: olivier.bournez@lix.polytechnique.fr.

Www: http://www.lix.polytechnique.fr/~bournez.

General context

With no contests, models, and approaches from deep learning have revolutionized machine learning.

At this date, a nice and robust associated complexity theory is missing. One first reason why such a theory is not existing is because the involved models are not classical models of computation over the discrete, but using real numbers. The **complexity and computability theory for models over the continuum** is far from being as clear as the classical one over the discrete.

About computation theory over the continuum:

Indeed, the very basics of classical computation theory are based on the (possibly effective) Church-Turing thesis [25, 23, 7]. It states that all *discrete* models that all (sufficiently powerful) reasonable computable models, or even reasonable machines, are equivalent to Turing machines. This means that we can abstract from the model of computation, both when talking about computability (what can be solved by algorithms), and when talking about complexity (what can be solved efficiently by algorithms). This leads to very famous questions such as $P = NP$ asking whether non-determinism helps, independently of the programming language, model of computation, or machine...

However, it is important to realize that the above statements restrict to discrete-time and discrete-space models: it is assumed that we restrict to computational models and machines working in steps (i.e. clock, or a discrete time) over discrete data, such as words, or integers.

When discussing computations over the reals, the situation is far from clear. Models of computations or machines, such as systems built using analog electronics, do not fall under the previous hypothesis [26, 27]. Various models of computation have been introduced to discuss computability and complexity, with various motivations. They are sometimes provably not equivalent.

This includes:

- **Computable analysis** (see e.g. [28, 10]) considers that a real is computable if there is an algorithm that produces its decimal representation. A function $f : \mathbb{R} \to \mathbb{R}$ is considered to be computable if there is an algorithm that maps a (suitable) encoding of x to the encoding of $f(x)$. Computable functions in this model must be continuous. It also permits us to discuss complexity: one assumes to produce digit number n in a time polynomial in n. Many statements from the classical analysis have been analyzed using this approach. Many questions from computational complexity can be related to questions about the complexity of continuous operators in the model of computable analysis [28, 19].

- **Algebraic models** such as Blum-Shub-Smale model [5, 4] or Valiant's model consider models of computation operating directly over real numbers: some given operations are assumed to be computable in unit time. For examples, it can be the operations $+, -, *, =$ and the relations $<$ and $=$: the ordered field $\mathbb{R}$ with its natural operations. It is then possible to discuss the class $\mathrm{P}_{\mathbb{R}}$, and the class NP_R where non-determinism is allowed, and this leads to the question whether $\mathrm{P}_{\mathbb{R}}$=$\mathrm{NP}_{\mathbb{R}}$. This question possibly differs from the classical question $\mathrm{P} = \mathrm{NP}$. This concept of complexity permits for example to discuss naturally the complexity of problems in computer algebra about polynomials. For example, telling whether a degree 4 polynomial has a real root is a NP_R-complete problem. While this model focuses on decision problems, Valiant's model focuses more on evaluation problems. It considers families of multivariate polynomials, and the complexity of such families can be measured by the size of the arithmetic circuits that compute them. This leads to the question $\mathrm{VP} = \mathrm{VNP}$.
- **Numerical analysis and Computer algebra** [6] has its own way of measuring the complexity of problems, which still differs. It is sometimes possible to relate the approaches as in [2], relating numerical analysis to questions about arithmetic circuits.
- **Analog computing**: An analogue computer is a type of computer that uses continuous quantities to compute [26, 27]. This includes for example machines built using analogue electronics, using operational amplifiers: they act on voltage and act according to some programmable Ordinary Differential Equation (ODE). A famous model is the General Purpose Analog Computer model from Claude Shannon [24], which has been proved to correspond to (vectorial) polynomial ordinary differential equations [15]. A natural question is whether we can compute faster using such models [9].

Of course, these are sometimes rather old models. But recent developments have been obtained, and some recently surprising relations have been obtained, and can possibly be used in the context of this proposal.

Deep learning and computation theory:

A second deep reason is that many of the natural questions involved in deep learning applications are **not decision problems** or approximation or counting problems and that classical complexity has been developed mainly focusing only on these aspects. Classical complexity does not seem directly relevant to discuss the complexity of involved problems. In addition, to previous issues, in particular, many of the problems considered are not well captured by decision problems, but rather (local) optimisation problems, and the way the complexity is measured seems often non-relevant to discuss the efficiency of considered methods and algorithms.

We can of course mention many recent results. For example, the question of the learning of deciding whether we can learn a neural network is ETR-complete [1]. The class ETR, sometimes denoted, $\exists\mathbb{R}$, is between NP and PSPACE . It corresponds to the class NP of BSS model mentioned above but without irrational constants. We are here in the context of decision problems. Some complexity classes have been introduced to discuss search problems, not covered by decision problems. This includes classes such as PPAD, PLS or CLS. Using this approach, the complexity of performing gradient descent has been characterized recently [13] by proving that this corresponds to complexity class $\mathrm{CLS} = \mathrm{PPAD} \cap \mathrm{PLS}$.

However, all of this is obtained at the price of assuming **very specific activation function**, such as the $ReLU(x) = \max(0, x)$ function, or piecewise algebraic/polynomial functions, in order to fall in the framework of existing complexity theory. The questions about what happens for the **actual activation functions** that are used in practice (see e.g. https://en.wikipedia.org/wiki/Activation_function), for example based on $\tanh$ is not covered. We believe that, using our expertise on models over the continuum, this is possible to extend some of these results, and furthermore to develop a robust and more suitable complexity theory.

Very deep learning vs deep learning:

We can also mention very deep learning models, by opposition to deep learning models: It is well known that when the number of layers increases (so-called **very deep models**, with sometimes more that 100 or 1000 layers), the models become very hard to train. Among a plethora of options that have been considered, Residual Neural Networks (*ResNets*) [16] have very clearly emerged as an important subclass of models. They mitigate the gradient issues [3] arising when training the deep neural networks. The idea in these particular models is to add skip connections between the successive layers, an idea partially bio-inspired. Since residual neural network was used and won the ImageNet 2015 competition, this particular architecture became the most cited neural network of the 21st century according to some studies (see references in wikipedia). Up to this date, winners of this competition are variations of such models.

Some authors, such as [29], proved that **there is a mathematical explanation** for their performance in practice, as the discrete-time process used in these models can be proved to be the Euler discretization of

some continuous time Ordinary Differential Equation (ODE). The observed obtained robustness and training properties, come then from the well-known robustness of ODEs with respect to perturbation and with respect to perturbation of their initial conditions.

It was later realized and proved mathematically that **various efficient models are actually nothing but reformulations of discretization schemes for ODEs**. For example, following [21], the architecture of *PolyNet* [31] can be viewed as an approximation to the backward Euler scheme solving the $\mathrm{ODE}\, u_t = f(u)$. *Fractalnet* [20] can be read as a well-known Runge-Kutta scheme in numerical analysis. *RevNet* [14] can be interpreted as a simple forward Euler approximation of some simple continuous dynamical system. All these models are very deep models, but this remains true for simpler models. For example, following [18], it transpires that the key features of well-known *GRU* [12] or an *LSTM* [17], over generic recurrent networks, are updates rules that look suspiciously like discretized differential equations.

This leaded to consider some models such as *neural ODE* [11], which can be seen as continuous versions of *ResNet*. While Neural ODEs do not necessarily improve upon the sheer predictive performance of ResNets, they offer the vast knowledge of ODE theory to be applied to deep learning research. For instance, the authors in [30] discovered that Neural ODEs are more robust for specific perturbations than convolutional neural networks. Moreover, inspired by the theoretical properties of the solution curves, they proposed a regularizer that improved the robustness of Neural ODE models even further. We do not intend to be exhaustive on the various applications of this new point of view on deep learning models.

Description of the work

We are experts of computability and complexity issues related to models of computation over the reals, covering discrete time or continuous time-models.

According to the taste of the candidate, we will **either focus on models from deep learning, or very deep learning**. For the first, we believe this is possible to adapt some of the constructions about the continumum to the framework of neural networks. For example, while it is well known that neural networks can approximate any function over a compact domain, we believe t is possible to actually discuss the complexity of the question from a computable analysis point of view, and get a way to analyse the complexity of the process or of functions. As an other example, we believe that several of the above mentioned results can be extended to deal with other functions than $Relu$ or piecewise polynomial functions, using some constructions from computable analysis.

For the latter (very deep learning models), continuous time models includes models based on ordinary differential equations. In particular, we know how to program with ordinary differential equations, and how to measure complexity for such models: see e.g. [9, 8, 22] for surveys. We used this knowledge in various contexts to solve some open problems in bioinformatics, applied mathematics, and other contexts. We propose here to develop this approach to the above models of very deep learning.

At the end, the purpose of the internship is to discuss complexity and computability issues for models of deep learning and very deep learning. While most of the approaches in the context of deep learning try to learn models, without clear understanding of what is feasible and what is not, the fact that we can actually build on purpose particular ordinary differential solving a given problem do provide some lower and upper bounds on the hardness of the learning process.

The objective will be to develop such results, and provide the basis for a theory for models of (very)deep learning.

Notice that this is the fact that these very deep models are very close to models based on ordinary differential equations that make this analysis feasible, while complexity theory is not well adapted to discuss classical models from (not very deep) deep learning.

Comments

The actual topic of the work is related to computability and complexity theory. The very deep learning part requires only common and basic knowledge in ordinary differential equations. Most of the intuitions of today's constructions come from classical computability and complexity.

There is no specific prerequisite for this internship, except some knowledge about computability theory. This subject can be extended to a PhD. Possibilities of funding according to the administrative situation of candidates.

The subject can also be adapted according to the requests, knowledge, and skills of candidates. Please contact me if interested or in case of questions.

References

[1] Mikkel Abrahamsen, Linda Kleist, and Tillmann Miltzow. Training neural networks is ER-complete. *Advances in Neural Information Processing Systems*, 34:18293–18306, 2021.

[2] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009.

[3] David Balduzzi, Marcus Frean, Lennox Leary, J. P. Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 342–350. PMLR, PMLR, 2017.

[4] Lenore Blum, Felipe Cucker, Mike Shub, and Steve Smale. *Complexity and Real Computation*. Springer, 1998.

[5] Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers; NP completeness, recursive functions and universal machines. *Bulletin of the American Mathematical Society*, 21(1):1–46, jul 1989.

[6] Alin Bostan, Frédéric Chyzak, Marc Giusti, Romain Lebreton, Grégoire Lecerf, Bruno Salvy, and Éric Schost. *Algorithmes efficaces en calcul formel*. published by the Authors, 2017.

[7] Olivier Bournez, Florent Becker, Jean-Loup Carré, Mathieu Liedloff, and Gérard Rozsavolgyi. *Informatique tout-en-un MP2I-MPI*. Dunod, 2024. à paraître.

[8] Olivier Bournez and Manuel L. Campagnolo. *New Computational Paradigms. Changing Conceptions of What is Computable*, chapter A Survey on Continuous Time Computations, pages 383–423. Springer-Verlag, New York, 2008.

[9] Olivier Bournez and Amaury Pouly. A survey on analog models of computation. In Vasco Brattka and Peter Hertling, editors, *Handbook of Computability and Complexity in Analysis*. Springer, 2021.

[10] Vasco Brattka, Peter Hertling, and Klaus Weihrauch. A tutorial on computable analysis. In *New computational paradigms*, pages 425–491. Springer, 2008.

[11] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pages 6571–6583, 2018.

[12] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[13] John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: $CLS = PPAD \cap PLS$. *Journal of the ACM*, 70(1):1–74, 2022.

[14] Aidan N. Gomez, Mengye Ren, Raquel Urtasun, and Roger B. Grosse. The reversible residual network: Backpropagation without storing activations. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, volume 30, pages 2214–2224, 2017.

[15] Daniel S. Graça and José Félix Costa. Analog computers and recursive functions over the reals. *Journal of Complexity*, 19(5):644–664, 2003.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] Patrick Kidger. On neural differential equations. *CoRR*, abs/2202.02435, 2022.

[19] Ker-I Ko. *Complexity theory of real functions*, volume 3 of *Progress in theoretical computer science*. Birkhäuser, Boston, 1991.

[20] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *ICLR*, 2016.

[21] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*, pages 3276–3285. PMLR, OpenReview.net, 2018.

[22] Pekka Orponen. Computational complexity of neural networks: a survey. *Nordic Journal of Computing*, 1(1):94–110, Spring 1994.

[23] S. Perifel. *Complexité algorithmique*. Références sciences. Ellipses, 2014.

[24] Claude E. Shannon. Mathematical theory of the differential analyser. *Journal of Mathematics and Physics MIT*, 20:337–354, 1941.

[25] Michael Sipser. *Introduction to the Theory of Computation*. PWS Publishing Company, 1997.

[26] Bernd Ulmann. *Analog and hybrid computer programming*. De Gruyter Oldenbourg, 2020.

[27] Veritasum. Future computers will be radically different (analog computing). Youtube video, 2022.

[28] Klaus Weihrauch. *Computable Analysis - An Introduction*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2000.

[29] E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 1(5):1–11, 2017.

[30] Hanshu Yan, Jiawei Du, Vincent Y. F. Tan, and Jiashi Feng. On robustness of neural ordinary differential equations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[31] Xingcheng Zhang, Zhizhong Li, Chen Change Loy, and Dahua Lin. Polynet: A pursuit of structural diversity in very deep networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3900–3908. IEEE Computer Society, 2017.