Multiscale science of neural networks

Mark A. Peletier

July 19, 2023

Machine learning algorithms are impressive. We hear about incredible achievements (see e.g. this overview article[1]), and these achievements inspire both optimism and concern[2].

At the same time, anyone who has actually tried to train a neural network, for instance, will have noticed that this training feels like alchemy: lots of trial-and-error, and even when the final result is good, we never really understand what was the crucial ingredient.

The good news is that in the last two or three years we are seeing great progress towards a *mathematical* theory of machine learning. It's still early, but the first signs are promising. Our group is part of the growing group of mathematicians in developing such a mathematical theory of machine learning, and there are many possibilities for bachelor and master students to participate.

Examples of project topics

1. **A simple 'student-teacher setup'.** Chizat, Oyallon, and Bach [COB19] study a very simple neural network, where a 'student' network learns the parameters of a 'teacher' network. They show how the choice of the parameter at the start of training has far-reaching consequences for the final parameter point that is found. Put simply: 'Small initial parameters lead to good trained networks; large initial parameters lead to bad ones'. Or differently: 'if the student starts with small parameters, then the student manages to copy the teacher network; but starting with large parameters, the student doesn't manage to learn.'

   This result leads to many questions, each of which could be the topic of a project. For instance,

   (a) In any practical situation, what is 'small' and what is 'large' may not be easy to determine. How to do this in practice?

   (b) Can we rigorously prove some of the experimental observations of Chizat, Oyallon, and Bach? And thus understand when they apply and when not?

2. **Mean-field limit of dropout neural networks.** As many of the modern neural networks are significantly overparametrized, one of the interesting questions is: What happens when we increase number of neurons? Or, more mathematically, is there a proper limit of infinitely-wide neural networks? One of the well-behaved limits is the so-called mean-field limit [SS20b]. It turns out that when trained with SGD, in the limit the dynamics of such a model satisfies a measure-valued evolution equation.

   At the same time, it is often practical to use various extensions of SGD such as dropout GD (randomly turning off some neurons at every optimization step). In this project we want to check whether arguments similar to [SS20b] apply to the mean-field limit of a neural network trained with dropout SGD. Possible extensions and/or alternatives include

   (a) Studying fluctuations of the process [SS20a]

[1] `https://ai100.stanford.edu/2021-report/standing-questions-and-responses/sq2-what-are-most-important-advances-ai`

[2] `https://www.europarl.europa.eu/news/en/headlines/society/20200918ST087404/artificial-intelligence-threats-and-opportunities`

(b) Finding an optimal behaviour of the dropout rate p_n (we would need to define "optimal" first)

(c) Comparing convergence results to [GKK23] (which one is stronger?)

(d) Deriving large-deviation theory for the model (this may be difficult)

3. **The famous 'Wojtowytsch collapse'.** Wojtowytsch [Woj21a, Woj21b] showed that *stochastic gradient descent (SGD)*, the standard method to train networks, has different behaviour depending on the *dimension of the null set of the empirical loss*. Again, in one sentence: 'If the network is very overparametrized, and the step size is sufficiently small, then SGD converges almost surely to a global minimizer of the empirical loss'. This sounds like something that one would want to have, but it also raises several questions, such as

   (a) Good performance of the network is not the same as minimizing the empirical loss; in fact, in experiments this a.s.-convergence to a global minimizer seems to be associated with *bad* parameter points. How does this work?

   (b) How does *data augmentation* (creating artificial data points, for instance by shifting and flipping images) change the behaviour? It seems that Wojtowytsch' result gives handles to describe that.

4. **Approximation properties of neural optimal transport maps.** One of the key tasks of machine leaning is density estimation (this is how you sample cat pictures). One possible approach is to use optimal transport theory, namely approximate optimal transport maps with functions given by neural networks; see for example [CAL21]. In this project we will study how the approximation error of transport maps translates into the accuracy of the density estimation, and try to answer the question: How big a should a neural network be to reliably sample a pretty cat? Possible extensions are:

   (a) Explicit comparison of different normalizing flows.

   (b) Incorporating sample complexity into the error estimate [HR21].

What will you learn?

A project of this type will allow you to learn a number of topics and skills:

- What a neural network is, what it can do, and what it can not do
- How to describe such a network in mathematical terms, and how to use mathematics to understand its properties
- How training such a network works in practice
- How to code up such a network yourself in Python

We will obviously adapt the level to whether you are doing a Bachelor or a Master project.

What do you need to know and have to do such a project?

Neural networks operate on the boundary between finite-dimensional and infinite-dimensional mathematics. Networks have a finite number of parameters, but this number is inordinately large (hundreds of millions of parameters is no exception). In addition, they operate on objects such as images, videos, or time series, that also can be very high-dimensional.

In practice this means that there is an advantage in treating the number of parameters as *infinite*, by working in an infinite-dimensional setting; in other words, in the setting of Functional Analysis. Many types of evolution in infinite-dimensional systems, such as the stochastic gradient descent mentioned above, are described by partial differential equations, and the theory of PDEs therefore also plays a central role.

For a Bachelor project the requirements are

- Introduction to Functional Analysis (2WAF0)
- Good coding skills in Python, and an interest in coding
- Having taking the bachelor PDE course (2WA90) is a plus, but not strictly necessary

For a Master project:

- Applied Functional Analysis (2MMA10)
- Scientific Computing (2MMN10) and Scientific Programming (2MMS20)
- Probability and Stochastics I (2MMS10)
- Good coding skills in Python, and an interest in coding
- Having taken Mathematics of Neural Networks (2MMA80) and/or and the master course Partial Differential Equations (2MMA20) is definitely a plus, but they are not strictly necessary

References

[CAL21] S. Cohen, B. Amos, and Y. Lipman. Riemannian convex potential maps. In *International Conference on Machine Learning*, pages 2028–2038. PMLR, 2021.

[COB19] L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. *Advances in Neural Information Processing Systems*, 32, 2019.

[GKK23] B. Gess, S. Kassing, and V. Konarovskyi. Stochastic modified flows, mean-field limits and dynamics of stochastic gradient descent. *arXiv preprint arXiv:2302.07125*, 2023.

[HR21] J.-C. Hütter and P. Rigollet. Minimax estimation of smooth optimal transport maps. 2021.

[SS20a] J. Sirignano and K. Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 130(3):1820–1852, 2020.

[SS20b] J. Sirignano and K. Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

[Woj21a] S. Wojtowytsch. Stochastic gradient descent with noise of machine learning type. Part I: Discrete time analysis. *arXiv preprint arXiv:2105.01650*, 2021.

[Woj21b] S. Wojtowytsch. Stochastic gradient descent with noise of machine learning type. Part II: Continuous time analysis. *arXiv preprint arXiv:2106.02588*, 2021.