Evolutionary optimization techniques to enhance deep learning

Gabriele Di Bari

Ph.D. student in Mathematics, Computer Science, Statistics at University of Perugia
Tutors: Marco Baioletti, Valentina Poggioni

Abstract. The target of my research program is to find new approaches to train Neural Networks (NN), in particular using Evolutionary Algorithms (EA).
This area of research, called Neuroevolution, studies the optimization of topology and weights of neural networks. The EAs are meta-algorithms, which have many advantages with respect to classical optimization techniques based on gradient descent (GD): (i) the *loss function* does not need to be differentiable, (ii) they do less likely get stuck in local minima, (iii) they are easy to parallelize.
The first approach was the application of Differential Evolution (DE) to NNs for supervised problems, and the results were promising. Moreover, the advantages abovementioned are the reasons why the EAs are largely used in other machine learning task, like Reinforcement Learning. Consequently, the application of EAs as NNs optimizer for Reinforcement Learning will be my future research subject.

Keywords: Neuroevolution · Machine Learning · Neural Networks · Evolutionary Algorithms · Reinforcement Learning.

Introduction

Supervised Learning is a Machine Learning task where a function is inferred from examples composed of a pair of input and corresponding output. To tackle this task many models were proposed. Among the most important model is the *Artificial Neural Network* (ANN), which tries to emulate the human brain behaviour through a weighted graph.

The usual method used to train the weights of Neural Network is the *Back-propagation* (BP) with *stochastic gradient descent* (SGD), which minimizes the *loss function* that evaluates the error between the NN outputs and the expected outputs. The minimization is achieved through the modification of the Neural Network weights. BD with SGD is an iterative local-convergence method which needs a *loss function* differentiable, and it is hard to parallelize.

There are other ways to train a Neural Network; one of them is the use of meta-algorithms coming from the Neuroevolution field. There are many advantages in using EAs. First of all, they can optimize functions which are not differentiable, since no derivatives are calculated by these algorithms. Then, EAs

are less likely to get stuck in local minima. They use a pool of solutions, combined with greedy methods, in order to create new solutions which explore most of the search space. Finally, EAs are easy to parallelize. In fact, they usually use pointwise operators, which can be concurrently applied to different solutions belonging to the same pool. However, there are some disadvantages, in fact, EAs suffers from stagnation problem and they are slower to converge in a local minimum.

With the aim to exploit the advantages of EAs, my research program has focused on the studying of the Differential Evolution (DE). DE is an Evolutionary Algorithm, which is used to optimize real-valued functions without the gradient. The DE is a composition of four meta-operations: Initialization, Mutation, Crossover, Selection. For each operator, there are many variants developed in order to improve DE performances. Moreover, the DE has two hyperparameters, the first one is F used by Mutation operator, and the other one is the used by Crossover operator. These parameters are essential for reaching the convergence. For this reason, there exist many self-adaptive methods of the DE hyperparameters.

The idea is to use DE as an optimizer of Deep Neural Networks. So, a batching system was proposed in order to reduce the computation time. Moreover, in order to find the best combination of the operators (and their parameters), they have been tuned through a Quade test analysis on several datasets.

Another application of NNs is Algorithm Learning, i.e the Machine Learning task where an algorithm is inferred from examples. The first model proposed by Alex Graves at. al. is called Neural Turing Machine (NTM) [7]. That model uses an NN as transition function of Turing machine. The NTM, in order to be fully differentiable, uses a fuzzy representation of the Turing machine states. Then, given the advantages of DE compared to classical methods, it was proposed in order to infer the transition function.

State of the Art

The common way for the training a NN in Supervised Learning approach is the Backpropagation with Stochastic Gradient [15]. This method is widely used both in the business application and in research fields.

On the other hand, Neuroevolution is not very common for business application. This is the reason why using EAs as NNs optimizer is a problem which lacks a properly defined state-of-the-art.

Anyway, the dominating approach used in Neuroevolution is the genetic one [5, 14, 18], which is generally used to optimize the connections of a NN. That approach is also used to optimize weights but, being a discrete approach, it needs an encoding phase. The encoding phase is crucial and heavily domain-dependent.

Another way is the use of Evolutionary Algorithms which work on real values such as DE. There are many papers on the application of DE as an optimizer for small NNs. In [12] the Differential Evolution is used, with the most known

variants, as the NN optimizer without a batching system (entire dataset is used during the training phase) for problems with the small domain. Another paper is [11], where the Limited Evaluation Evolutionary Algorithm (LEEA), quite similar to the DE, is used with a batching system (but without fitness inheritance) in order to train different NNs on small problems. After the publication of my preliminary results Prellberg and Kramer, in [13], approached the MNIST problem with a small Convolutional Neural Network (CNN). Such network, optimized first through Backpropagation and then with their own EA, achieved the same accuracy, which is 97% and it is not too far from the state of the art.

Another Machine learning field, where Neuroevolution is growing up, is the Reinforcement Learning [16, 17, 6]. Regarding Reinforcement learning, EA algorithms are common both in the business application and in the research field [16, 17, 6]. In particular, there are some cases where the use of the EAs gives better results than temporal difference methods (e.g. Q-Learning) [10].

Contributions

In my master thesis, I proposed the design and the development of a machine learning method based on DE for neural network optimization: DENN (Differential Evolution for Neural Networks). For DENN, I developed a batching system with an effective fitness inheritance method, which permitted the use of DE as a Feedforward Neural Networks optimizer for real Supervised classification problems. The algorithm and the related tests are presented in [1]. The results of the tests are comparable to the ones obtained using BP with SGD.

After the thesis, I continued to study and develop variants of DE in order to improve DENN. During my first Ph.D. year, I focused on trying all the common variants of DE and its operators in order to determine which of them was the best for NNs training. Moreover, during this phase, I developed a new Crossover operator, called *interm*, which has proven to work well. The operators (and their parameters) have been tuned through a Quade test analysis performed on several datasets. The results of the analysis and the *interm* implementation are shown in [2].

Subsequently, I worked on the application of DENN on Machine Learning tasks where the classic methods do not perform well; I developed a variant of DENN which uses a model based on Neural Turing Machines (NTM) [7] called Neural Random-Access Machines (NRAM) [8]. The NRAM is a model inspired by CPUs which features a controller, registers, a memory, and many operators. The controller is a trainable Feedforward Neural Network called Neural Controller, which represents the program to infer and which resolves a problem through explicit manipulation and referencing of pointers. The results are close to the ones in the original paper where the authors used Backpropagation with ADAM [9] and curriculum learning [3].

Research Methodology and Approach

Some of the objectives of my proposal were already achieved: first, I designed and developed a DE variant which is able to train a Feedforward NN; then, I tuned the DENN operations and their parameters in order to find the best combination to be applied to Supervised Learning problems. Finally, I used DENN as optimizer of Feedforward Neural Networks for Algorithm learning. For the future, the long-term objectives of my proposal can be divided into three steps: the first one is to use the knowledge I acquired on DE in order to design Evolutionary Algorithms able to train Recurrent Neural Networks (RNN). The next step will be to apply Neuroevolution techniques to train a Recurrent Neural Controller for a NTM-like model. Moreover, I am planning to design and develop EAs variants to be applied to Reinforcement learning problems. In the beginning, I would like to use EAs in virtual worlds, such as the *Starcraft II Learning Environment*, the environment used by Google DeepMind to improve Starcraft 2's AI. Such environments are good to train and test models because of their controlled structure.

Preliminary Results

I applied evolutionary algorithms to the Feedforward Neural Networks optimization, with good results presented in [1] and in [2].

About the results, on the small dataset DENN reaches better results then Backpropagation.

For the known MNIST dataset, the DENN algorithm applied to a network without the hidden layers, reaches more than 90% of accuracy, the same result obtained by Backpropagation.

The Quade test analysis identified the best combination of DE operators, which are *current_to_pbest* as Mutation, *interm* as Crossover, and *SHADE* as Self-Adaptive method.

Regarding NRAM and the problem of algorithms learning, the preliminary results will be presented at AI*IA 2018 [4].

Conclusions

During the last years, Machine learning became popular not only for the research world, but also in the business field.

Looking at both preliminary results obtained during first year and others recent researches, the study of Evolutionary Algorithms on deep learning problems appears to be promising in order to solve real world problems.

In the future, more attention will be given to the research area of Reinforcement learning, which can guarantee a more general approach with respect to Supervised learning.

References

1. M. Baioletti, G. Di Bari, V. Poggioni, and M. Tracolli. Can differential evolution be an efficient engine to optimize neural networks? In *International Workshop on Machine Learning, Optimization, and Big Data*, pages 401–413. Springer, 2017.
2. M. Baioletti, G. Di Bari, V. Poggioni, and M. Tracolli. Differential evolution for learning large neural networks. Tech. Rep., available at https://github.com/Gabriele91/DENN-RESULTS-2018, 2018.
3. Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML*, 2009.
4. G. Di Bari, V. Poggioni, M. Baioletti, and V. Belli. Neural random access machines optimized by differential evolution. AI*IA,The 17th International Conference of the Italian Association for Artificial Intelligence, Available at https://github.com/Gabriele91/DENN-NRAM-RESULTS-2018, 2018.
5. D. Floreano, P. Dürr, and C. Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.
6. P. Goyal, H. Malik, and R. Sharma. Application of evolutionary reinforcement learning (erl) approach in control domain: A review. In *Smart Innovations in Communication and Computational Sciences*, pages 273–288. Springer, 2019.
7. A. Graves, Greg Wayne, and I. Danihelka. Neural turing machines, 2014.
8. K. Karol, A. Marcin, and S. Ilya. Neural random-access machines. *CoRR*, abs/1511.06392, 2015.
9. D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.
10. D. E. Moriarty, A. C. Schultz, and J. J. Grefenstette. Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research*, 11:241–276, 1999.
11. G. Morse and K. O. Stanley. Simple evolutionary optimization can rival stochastic gradient descent in neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2016*, pages 477–484, New York, NY, USA, 2016. ACM.
12. A. P. Piotrowski. Differential evolution algorithms applied to neural network training suffer from stagnation. *Applied Soft Computing*, 21:382 – 406, 2014.
13. J. Prellberg and O. Kramer. Limited evaluation evolutionary optimization of large neural networks. 06 2018.
14. J. D. Schaffer, D. Whitley, and L. J. Eshelman. Combinations of genetic algorithms and neural networks: a survey of the state of the art. In *[Proceedings] COGANN-92: International Workshop on Combinations of Genetic Algorithms and Neural Networks*, pages 1–37, 1992.
15. J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. Published online 2014; based on TR arXiv:1404.7828 [cs.NE].
16. B. Wang, Y. Sun, B. Xue, and M. Zhang. A hybrid differential evolution approach to designing deep convolutional neural networks for image classification. *arXiv preprint arXiv:1808.06661*, 2018.
17. R. Xiong, J. Cao, and Q. Yu. Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle. *Applied Energy*, 211:538–548, 2018.
18. X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.