Evolutionary Machine Learning: An Essay on Benchmarking

Filipe Assunção
fga@dei.uc.pt
Nuno Lourenço
naml@dei.uc.pt
Bernardete Ribeiro
bribeiro@dei.uc.pt
Penousal Machado
machado@dei.uc.pt

CISUC
Department of Informatics Engineering
University of Coimbra
Coimbra, Portugal

Abstract

The search for adequate structures and parameters for Machine Learning (ML) models is problem specific and time consuming. Often, researchers follow an iterative trial-and-error process, where suitable values for multiple parameters are tested. One way to address this issue is the application of Evolutionary Computation (EC) to search, optimise and tune the ML models. Selecting appropriate benchmarks for comparing different approaches is not always trivial and is a common problem of both EC and ML. However, when combining both fields it is possible to use the evolutionary process to our advantage, speeding up the evaluation stage. In this paper we discuss what can be done to mitigate some of the issues of benchmarking in Evolutionary Machine Learning (EML). The positions herein presented denote the point of view of the authors and should not be seen as a strict methodology, but rather as a set of guidelines.

1 Introduction

Evolutionary Machine Learning (EML) is a sub-field of Artificial Intelligence that applies Evolutionary Computation (EC) to automatically search for the structure and/or parameterisation of Machine Learning (ML) models. Examples of EML works are the evolution of the topology and weights of Artificial Neural Networks (ANNs) [3], or the evolution of multiple Support Vector Machine (SVM) parameters [4]. By adopting the principles behind EC, a population of solutions scattered in the domain space of the ML model is evolved, making it less likely to become trapped in a local optima, which would likely happen if the model was tuned by manually trying different parameter configurations in an attempt to reach near-optimal performances.

When working with EML one of the main challenges concerns benchmarking. In brief words, benchmarking can be defined as the act of performing experiments with datasets to compare the performance of different approaches. From this definition it is clear that the selection of datasets, experimental design and analysis are principles that are linked. In the current essay we will focus primarily on tackling the issues related with benchmarking from the perspective of the datasets, i.e., we will discuss the questions that must be faced when planning the experiments to test new / existing approaches regarding the choice that has to be made regarding the appropriate benchmarks for assessing the quality of the methods, and how many datasets should be used.

If we decide on simple benchmarks, with a low number of features and instances, the methodology will likely find solutions that perform well. But, these results have questionable importance, since solutions for such problems might be easily hand-crafted. On the other hand, when dealing with real world complex problems, mapping the candidate solutions to a comprehensible model, and assessing their quality on such huge datasets can be time consuming, making it impossible to timely measure the quality of the models. In addition, to tackle such problems, we often require large amounts of computational power, making the use of the benchmark unfeasible. Discussion around the problematic of benchmarking is not new. McDermott et al. [7] have already pointed the selection of benchmarks for evaluating Genetic Programming (GP) methods to be one of the open issues in the field.

In this position paper, more than proposing a methodology that should be strictly followed, we aim at discussing good practises. In Section 2 we focus on dataset selection. Next, in Section 3, we investigate methods that try to cope with the challenges posed by big data. To end, in Section 4, open questions and future discussion are raised.

2 Datasets

In EML the datasets are commonly grouped according to the ML task that the models being evolved aim to solve: (i) clustering; (ii) regression; or (iii) classification. In addition, and following the structure proposed by Prechelt [9] these benchmarks belong to one of the following categories:

Artificial – data is artificially generated following a given equation (logic or arithmetic);

Realistic – although the data is also artificially generated (as in the above category) it simulates the rules and specifications of real world systems (e.g., physical models);

Real problems – data gathered directly from observing the real world.

More and more contributions to the field have focused on the use of real world problems, which is motivated by the desire to search for true Artificial Intelligence, i.e., systems capable of outperforming the human performance and automate common everyday tasks.

With the increase of computational power and performance of the evolved models, popularised by Graphics Processing Units (GPUs) and consequent emergence of deep learning techniques, the problems that practitioners try to solve are becoming increasingly more challenging. However, there are no well-established methodologies specifying how to select which problems to test on. Although there are works in the literature describing how to measure the complexity of datasets (e.g. [11]), they tend to be difficult, and time consuming to use. It is impractical to apply such methods to a wide range of benchmarks, and thus, authors often base their decision on the complexity in terms of number of instances and dimensionality of the input space / number of features, and on the benchmarks used by the methodologies to which comparisons are going to be established. Another criteria that is often analysed is the available progression margin, which defines the problem complexity based on the difficulty that previous approaches had to solve it.

There are several platforms that work as repositories for benchmarks. The most popular is the UCI ML repository [6], which at the time of writing is composed by 394 datasets. The UCI platform stores information on each dataset, which includes the number of instances, features, types of the data and the task to be performed. In addition, a brief description of each benchmark and corresponding reference papers are provided. Although it provides a good platform in the sense that it allows users to scan a large list of benchmarks, showing their main characteristics, it does not provide a list of the results obtained by previous methodologies in a clean and accessible way. Furthermore, despite the large number of available benchmarks, a large percentage (approximately 40%) has less than 1000 instances, and about 23% have no more than 10 features. A platform that solves one of the issues found in the UCI repository is Kaggle (check `http://www.kaggle.com`): a web-platform for the organisation of contests often tackling real world problems. By using leaderboards the performance of different approaches in each benchmark becomes clear. The benefits of the the two previous platforms are combined in OpenAI [2]; the main disadvantage of OpenAI is that it is focused on reinforcement learning problems, more particularly, game environments.

From the above discussion on the available platforms we suggest that the ideal platform should at least follow the following principles:

- Provide a detailed description of each dataset, including properties such as the number of features and instances, task to be performed, and type of the dataset. Complexity of the dataset according to established metrics should also be provided;

- The performances obtained on each benchmark by different approaches should be shown and detailed in the form of a list. Each result entry should be accompanied by the article describing the method and whenever possible the implementation. Authors should be allowed to submit this information so that the platform is self-maintaining;

- Ideally, the platform should provide means to confirm the accuracy of the results by automatically running new experiments with the provided code on different partitions of the benchmark;

- It should be possible to order and filter the benchmarks available in the platform according to any of its properties and results, so that one can easily explore them and choose the ones to tackle.

So far we have discussed the main challenges in the analysis and decision of which benchmarks to use for testing purposes. But, the question of how many benchmarks should be used has not yet been addressed. An obvious answer would be that the more datasets are used the better, so that it is easier to characterise the behaviour of the tested method on benchmarks with different properties. However, this might not be feasible: papers have limited sizes, and the time needed for conducting such an amount of experiments makes authors inclined to select a small amount of benchmarks. Specially if we are dealing with real world problems, where the available amounts of information often comprise Big Data (further discussed in the next section). We recommend that experiments should be conducted in at least four benchmarks (the more the better). Testing on fewer than that does not allow for any strong conclusions about the quality of the approach rather than the one that it performs better or worse in a couple datasets, but an extrapolation and generalisation assumption to other benchmarks can hardly be made. Moreover, it is our opinion that the benchmarks should be selected with an increasing complexity degree, so that it is possible to test if the approach despite performing good in difficult problems also leads to good results in simple and easier tasks, and vice-versa.

3 Dealing with Big Data

By combining the principles of EC with ML, a population of candidate solutions encoding the model's structure and/or parameters is evolved through time. Although evolution is paralallelisable, assessing the quality of each candidate solution in real world problems can be time consuming.

Several tools that take the advantages provided by GPU computing have been proposed recently (e.g., Caffe [5] or Tensorflow [1]). Using these frameworks has two main advantages: in the one hand, they provide stable implementations for evaluating the performance of the evolved models; on the other hand, by providing GPU interfaces they speed up the evaluation time.

Nevertheless, in some circumstances the speed up introduced by the use of GPU processing is not enough. Imagine evaluating a deep neural network that takes about 1 hour to train; if a population of 100 candidate solutions is evolved then each generation would take about 4 days, which makes evolution unfeasible. Thus, when this happens authors normally resort to sampling techniques, and smaller training sessions that give some insight on the expected performance of the model on the long term. We believe that random sampling approaches are not the most appropriate form to reduce the dimension of datasets, as they may fail to retain some of the properties of the benchmark, possibly leading to deceiving results. We defend that we should use the evolutionary process in our favour, and sample a percentage of the instances of the benchmark every given number of generations, taking the results on the samples into account when generating new ones. Examples of these type of approaches have been proposed by Stanovov et al. [10] and Morse and Stanley [8].

4 Road Ahead

In this short essay we have pointed out various aspects of current benchmarking practices in EML. We have discussed the following issues:

- Limitations imposed by the difficulty of selecting a set of benchmarks for testing a developed (or existing) methodology. It is not clear what makes a dataset complex; most practitioners make such decisions based on benchmark properties, such as number of instances and dimensionality of the input space, or based on the performance of other approaches;

- Similar to the selection of benchmarks, the same rationale can be applied to deciding how many datasets should be used for conducting experiments. We defend that at least four different datasets, with different complexities should be used.

- Dealing with Big Data is challenging, specially when multiple candidate solutions are being evolved in simultaneous, and need to be evaluated for assessing their performance. To deal with this limitation the dataset can be sampled, but taking advantage of the iterative nature of EC.

Nonetheless, there are many more questions that have to be addressed by further research in the area, which directly impact how the evolved models are selected and compared. One of the most important ones comprises the definition of metrics that can objectively measure the difficulty of benchmarks. We are well aware that this is not an extensive review, and that there are several works that already follow some of the guidelines discussed here. Our main goal with this essay is to set off a discussion about good practices that can improve the field, leading to better and sound research.

Acknowledgements

This work is partially funded by: Fundação para a Ciência e Tecnologia (FCT), Portugal, under the grant SFRH/BD/114865/2016.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

[3] Dario Floreano, Peter Dürr, and Claudio Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

[4] Frauke Friedrichs and Christian Igel. Evolutionary tuning of multiple SVM parameters. *Neurocomputing*, 64:107–117, 2005.

[5] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[6] M. Lichman. UCI ml repository, 2013. URL `http://archive.ics.uci.edu/ml`.

[7] James McDermott, David R White, Sean Luke, Luca Manzoni, Mauro Castelli, Leonardo Vanneschi, Wojciech Jaskowski, Krzysztof Krawiec, Robin Harper, Kenneth De Jong, and Una-May O'Reilly. Genetic programming needs better benchmarks. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 791–798. ACM, 2012.

[8] Gregory Morse and Kenneth O. Stanley. Simple evolutionary optimization can rival stochastic gradient descent in neural networks. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, pages 477–484. ACM, 2016.

[9] Lutz Prechelt. A quantitative study of experimental evaluations of neural network learning algorithms: Current research practice. *Neural Networks*, 9(3):457–462, 1996.

[10] Vladimir Stanovov, Eugene Semenkin, and Olga Semenkina. Instance selection approach for self-configuring hybrid fuzzy evolutionary algorithm for imbalanced datasets. In *International Conference in Swarm Intelligence*, pages 451–459. Springer, 2015.

[11] Julian Zubek and Dariusz M Plewczynski. Complexity curve: a graphical measure of data complexity and classifier performance. *PeerJ Computer Science*, 2:76, 2016.