POLITECNICO
MILANO 1863

School of Industrial and Information Engineering
Department of Electronics, Information and Bioengineering

Master of Science Degree in Computer Science and Engineering

Pareto-Optimal Progressive Neural Architecture Search

Supervisor: Prof. Danilo Ardagna
Co-supervisor: Eng. Marco Lattuada

Master thesis by:
Matteo Vantadori, matricola 884021

Academic Year 2018-2019

Alla mia famiglia.

Abstract

Designing neural network architectures involves a series of criticisms, ranging from the unpredictable human efforts gone into fine-tuning them to a not granted intuition into how to design them to achieve a good accuracy. Neural Architecture Search (NAS) is the process of automating architecture engineering, searching for the best machine learning model. One of the main NAS approaches proposed in the literature relies on an already trained controller based on a recurrent neural network (RNN) to explore the neural networks search space by using a configuration string to encode the architectures, training the candidate models and evaluating their accuracy. At each step, the controller parameters are updated based on the trained networks accuracy, exploiting reinforcement learning. Progressive Neural Architecture Search (PNAS) seeks a more efficient method, searching for the architectures in order of increasing complexity with a sequential model-based optimization strategy: it defines a common recursive structure to generate the networks, whose number of building blocks rises through iterations. However, those algorithms are generally designed for an ideal setting, without taking into account the needs and the technical constraints of an ordinary user.
In this thesis, we propose a new architecture search called Pareto-Optimal Progressive Neural Architecture Search (POPNAS), that combines the benefits of PNAS to a time-accuracy optimization problem. POPNAS adds a new time predictor to the existing RNN controller, in order to carry out a joint prediction of time and accuracy for each candidate neural network, searching through the Pareto front. This approach allows us to reach a trade-off between accuracy and training time, identifying neural network architectures with good accuracy in the face of a drastically reduced training time.

Ringraziamenti

C'è un detto che ho imparato da una serie TV e che ripeto sempre nei momenti tristi: cogli il limone più aspro che la vita ha da offrirti e trasformalo in qualcosa di quanto più simile a una limonata.
Mi laureo in un momento difficile, lontano dalle persone che avrei voluto al mio fianco. Mi mancano le pause al Politecnico, mi mancano le serate al cinema, mi manca Milano. Ma, soprattutto, mi mancano i miei amici. Eppure, al mio fianco, vedo i volti di coloro che mi hanno accompagnato in questi anni. Chi c'è sempre stato, chi ho incontrato a metà strada, chi si è unito all'ultimo. Sappiate che avete reso la mia vita universitaria la più dolce limonata possibile, e per questo ve ne sarò sempre grato.
Valentina, con te ho condiviso molto più dell'università. Con te ho condiviso la mia vita. Quello che ora sono diventato lo devo anche a te, molto più di quanto non sarò mai pronto ad ammettere. In inglese, esiste un termine che i marinai utilizzano per indicare la stella scelta come unico punto di riferimento nel loro viaggio, per non perdere mai la rotta: *"lodestar"*. È quello che tu sei stata e che sarai sempre. Grazie per essere la parte migliore di me.
Mamma e papà, sapete più di tutti che la vita non è sempre stata clemente con me. Ho dovuto affrontare dure prove, da cui sono uscito scalfito e affranto. Ma oggi sono qui per dirvi: *"ce l'ho fatta"*. Oggi non c'è posto per la tristezza e l'autocommiserazione, perché questa è la mia piccola grande rivincita su tutto. Grazie per esserci sempre stati e per avermi concesso l'opportunità di diventare quello che sono ora. D'ora in avanti, vi prometto che darò tutto me stesso per dimostrare, ogni singolo giorno, che la vostra scommessa non è stata vana.
Ho avuto una famiglia che mi ha sempre incoraggiato, affiancato e sostenuto, anche più di quanto non meritassi. Per questo, voglio ringraziare personalmente anche i miei nonni Giordana, Stefano, Angela e Giansandro e i miei zii Lara e Giuseppe. Spero di essere un nipote di cui andare orgogliosi.
Ivan e Franklin, siete stati i comprimari del capitolo più bello della mia vita. Con voi ho condiviso i miei giorni, il mio studio e il mio svago. Da voi ho imparato a essere un amico e un compagno di vita migliore. Assieme abbiamo vissuto avventure incredibili, alcune probabilmente inenarrabili, ma tutte sicuramente indimenticabili. Siamo stati la miglior sitcom dell'ultimo decennio. Ho un solo rimpianto: che sia durata troppe poche stagioni.
Andrea, Filippo, Leandro e Giorgio, per me sarete sempre la vecchia guardia del Politecnico. Ormai ho perso il conto delle ore passate con voi in aula studio. Che vi fosse la più epocale delle nevicate o il caldo torrido di agosto, non abbiamo mai saltato un giorno. Per questo, ora che tutto è finito, ancora fatico a immaginare i miei pomeriggi senza tavoloni, senza pause caffè, senza chiusura alle 22. Senza di voi.

Alberto, Simone, Vanessa e Arianna, mi avete affiancato in quest'ultimo anno di università, forse il più spensierato. Lo ammetto, con voi mi sono divertito davvero tanto. Se penso di dover dire addio ai pomeriggi trascorsi assieme in portineria, mi assale un senso di profonda malinconia. Avrete sempre un posto speciale nel mio cuore.
Marco, Andrea, Luca, Lorenzo, Federico e Riccardo, non mi sono certo dimenticato di voi. Siete stati un'importante costante della mia vita e spero con tutto il mio cuore che continuerete a esserlo per sempre. Se fossi Calvin, sareste i miei Hobbes. Se fossi Linus, sareste i miei Charlie Brown. Se fossi Scrooge McDuck, sareste la mia Numero Uno. Perché siete tutto ciò di cui ho bisogno. Grazie per essere dei veri amici.
Vorrei infine ringraziare il Prof. Ardagna non solo per avermi guidato in questa magnifica esperienza, ma per l'infinita disponibilità e la grande vicinanza che ha dimostrato in questi ultimi difficili mesi.
Ora penso al futuro e a cosa succederà. Provo a immaginarmi tra qualche anno. Mi chiedo dove sarò, con chi sarò. Proietto le mie ambizioni, le mie speranze, i miei sogni, convinto più che mai a trasformarli in realtà.

> *"E chi dice che non accadrà? Chi può dirmi che le mie fantasie non si avvereranno?*
> *Almeno questa volta."*
>
> – J.D., "Il mio finale". *Scrubs*, 2009.

Contents

Chapter 1

Introduction

In the last couple of years, the contribution of automated machine learning is exploded in the field of data science. The process of automating machine learning has made it much more accessible to those who are not specialized: approaching it as a black box model, a user can exploit it in order to find a solution to a specific task without having any knowledge about that field.
The resulting success of the automation of processes has led to a rising demand to extend the same principle to architecture engineering, where increasingly more complex neural network architectures are designed manually even today. Despite the neural networks success as powerful and flexible models specialized in many crucial learning tasks in image, speech and natural language understanding, the handcrafted design remains a main constraint in terms of time taken and resources spent: there is no guideline which grants a good intuition into the best network design, as well as we cannot predict the human efforts involved in its production.
Neural Architecture Search is thus the logical next step in automating machine learning, as the process of automating architecture engineering: its object is to maximize the expected accuracy of an automatically generated neural network, starting from scratch, that could rival with the best human-invented architecture for the same task.
Today, the mission of automated machine learning has become the artificial intelligence democratization, lowering the entry barrier and making it aviable to the largest possibile community of developers, researchers and businesses. Google Cloud AI services, for instance, counts today more than 10,000 businesses, including many famous companies. However, its AutoML suite costs $20 per hour and provides only the final result, not also the procedure. In addition, the final optimized model provided by Google becomes accessible to the user only within its platform.
Neural Architecture Search constitutes the main research interest of automated machine learning, aiming to generate a robust and well-performing neural architecture by selecting and combining different basic components from a predefined search space. Due to the breadth of the topic, the NAS algorithms are grouped accordingly to three dimensions: the search space indicates which neural networks should be taken into account, the search strategy outlines the strategy to adopt for the search space exploration, while the performance estimation strategy defines how to compare the considered neural networks.

The first major objective is therefore to make a conscious decision regarding the proper NAS algorithm to choose as our starting point, considering the goal to predict the training time required by each neural network belonging to the search space, in order to impose a maximum training time limit to it. In fact, the literature uniquely considers a network performance as a synonym for its accuracy. In our work, we have decided to extend the performance meaning to the network training time, since we want to provide guarantees in terms of time, in case it is necessary to retrain the network from scratch on a new dataset.
Our choice fell on Progressive Neural Architecture Search, a NAS algorithm that exploits a simple-to-complex approach, starting from the simplest models of the search space and progressively adding them units in order to create new structures, pruning out the unpromising ones. In this way, we can be sure that any architecture derived from another one with training time higher than the maximum limit will not be considered by the algorithm, focusing only on networks with lower training time under the same accuracy.
From here on out the thesis is structured in a progressive way, starting from a deepening of the literature, passing through the illustration of the proposed algorithm and its implementation and arriving to the tests we carried out.
Chapter 2 is entirely devoted to a comprehensive explanation of the state-of-the-art, as regards the Neural Architecture Search. Here we describe the three dimensions in which it is parametrized.
Chapter 3 shows how Progressive Neural Architecture Search works, i.e. the algorithm we have decided to adopt as the starting point in our research, accordingly to the three dimensions already mentioned.
Chapter 4 illustrates Pareto-Optimal Progressive Neural Architecture Search, that is the algorithm we developed.
Chapter 5 describes the Pareto-Optimal Progressive Neural Architecture Search implementation for each described dimension.
Chapter 6 lists the tests we have run with their specification, in order to obtain a final neural network that can compete with the one found by PNAS in terms of accuracy, but with a significantly lower training time.
Chapter 7 sums up the entire work done, suggesting some possible future development.
Appendix A lists all the required specifications adopted for our experiments, with their versions.
Appendix B is a detailed user guide in which we explain how to launch the software tool we developed.

Chapter 7

Conclusions and future work

This thesis work has been motivated by the lack of interest from neural architecture search in the final user, not considering its hardware requirements and economic limitations. Once we have identified the training time as a crucial point to be minimized in order to meet the user needs in the search for the best neural network, we have analyzed the state-of-the-art, looking for an algorithm well suited for our problem to identify a trade-off between the model accuracy and its training time.
The choice fell on Progressive Neural Architecture Search, a method that considers the search for the architecture structures in order of increasing complexity: after defining a cell as a recursively stacked element of the network topology, composed by a basic unit defined as block, we have explained how the search algorithm relies on reinforcement learning, by using a LSTM controller that picks up the K cells with the best predicted accuracy to train them and evaluate the loss function and the relevant gradient, in order to update its weights for the next prediction.
We have proposed the Pareto-Optimal Progressive Neural Architecture Search algorithm, based on the concept of training time prediction. In our search for the best predictor, we have considered three regressor models and a heuristic algorithm: Ridge regression, NNLS, XGBoost and block sum. For this purpose, we have also introduced three new mechanisms for the observation encoding that facilitate the regressors time prediction: the operator reindex, the sliding blocks mechanism and the initial thrust improvement.
We have implemented a Pareto optimality solution that considers training time and accuracy as state variables, in order to choose the K child networks with the best accuracy among those which belong to the Pareto front.
Finally, we have carried out a series of tests with the following progressive objectives:

1. Find the best predictor among those considered on PNAS.
2. Test the actual improvement of the encoding methods.
3. Evaluate the time prediction performance on POPNAS.
4. Compare our results with the literature.

XGBoost and block sum have been identified not to be suitable for a time prediction problem with increasing complexity: our analyses make a clear point on the fact that they are extrimely incline to underestimate the training time. As the best predictor, we have selected NNLS with dynamic operator reindex, obtaining satisfactory results: a high accuracy of the final model for the classification problem as been reached in the face of a huge reduction in terms of training time.
In the future we have planned some minor changes in the algorithm and its implementation, in order to further improve, if possible, the time prediction and bring benefits to the final end-user.
The accuracy of the obtained time prediction strongly depends on the predictor type we decide to adopt and its constraints. In the future, we aim to explore other predictor fields, in order to find a new regressor with a less average prediction error with respect to the one we have adopted. A different kind of approach may be to generate the observations translating the categorical variables by means of the one-hot encoding, making the operators independent of the training time ranking at runtime.
Since we have at our disposal only one GPU at time, the training step is organized in a way that trains one child network by one. The total time spent at each iteration is given by the sum of the training times of the top-K architectures. Another important future objective we set is to modify the child networks managing in order to exploit a multi-GPU approach: this new asset will allow to scale the time spent at each iteration by the number of used GPUs, drastically reducing the total training time.

Bibliography

[1] Thomas Elsken, Jan Hendrik Metzen and Frank Hutter. *Neural Architecture Search: A Survey.* Journal of Machine Learning Research 20, 2019.
https://arxiv.org/abs/1808.05377

[2] Xin He, Kaiyong Zhao and Xiaowen Chu. *AutoML: A Survey of the State-of-the-Art.*
https://arxiv.org/abs/1908.00709

[3] Marc-André Zöller and Marco F. Huber. *Benchmark and Survey of Automated Machine Learning Frameworks.*
https://arxiv.org/abs/1904.12054

[4] Radwa Elshawi, Mohamed Maher and Sherif Sakr. *Automated Machine Learning: State-of-The-Art and Open Challenges.*
https://arxiv.org/abs/1906.02287

[5] Frank Hutter, Lars Kotthoff, Joaquin Vanschoren. *Automated Machine Learning: Methods, Systems, Challenges.* Springer, 2019. Chapter 1, pages 3-38.
https://doi.org/10.1007/978-3-030-05318-5

[6] Ke Li and Jitendra Malik. *Learning to Optimize.* International Conference on Learning Representations, 2017.
https://arxiv.org/abs/1606.01885

[7] James R. Lloyd, David Duvenaud, Roger Grosse, Joshua Tenenbaum and Zoubin Ghahramani. *Automatic Construction and Natural-Language Description of Nonparametric Regression Models.* AAAI Conference on Artificial Intelligence, 2014.
https://arxiv.org/abs/1606.01885

[8] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana Elias B. Khalil and Deepak Turaga. *Learning Feature Engineering for Classification.* International Joint Conference on Artificial Intelligence Main track, 2017.
https://www.ijcai.org/Proceedings/2017/0352

[9] Sanjay Krishnan, Michael J. Franklin, Ken Goldberg and Jiannan Wang. *ActiveClean: An Interactive Data Cleaning Framework For Modern Machine Learning.* International Conference on Management of Data, 2016.
https://dl.acm.org/doi/10.1145/2882903.2899409

[10] Barret Zoph, Vijay Vasudevan, Jonathon Shlens and Quoc V. Le. *Learning Transferable Architectures for Scalable Image Recognition.* Conference on Computer Vision and Pattern Recognition, 2018.
`https://arxiv.org/abs/1707.07012`

[11] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao and Cheng-Lin Liu. *Practical Block-wise Neural Network Architecture Generation.* Conference on Computer Vision and Pattern Recognition, 2018.
`https://arxiv.org/abs/1708.05552`

[12] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang and Kevin Murphy. *Progressive Neural Architecture Search.* International Conference on Machine Learning, 2018.
`https://arxiv.org/abs/1806.09055`

[13] Hanxiao Liu, Karen Simonyan and Yiming Yang. *DARTS: Differentiable Architecture Search.* International Conference on Learning Representations, 2019.
`https://arxiv.org/abs/1806.09055`

[14] Sirui Xie, Hehui Zheng and Chunxiao Liu, Liang Lin. *SNAS: Stochastic Neural Architecture Search.* International Conference on Learning Representations, 2019.
`https://arxiv.org/abs/1812.09926`

[15] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le and Jeff Dean. *Efficient Neural Architecture Search via Parameter Sharing.* International Conference on Machine Learning, 2018.
`https://arxiv.org/abs/1802.03268`

[16] Liam Li and Ameet Talwalkar. *Random Search and Reproducibility for Neural Architecture Search.* Conference on Uncertainty in Artificial Intelligence, 2019.
`https://arxiv.org/abs/1902.07638`

[17] Barret Zoph and Quoc V. Le. *Neural Architecture Search with Reinforcement Learning.* International Conference on Learning Representations, 2017.
`https://arxiv.org/abs/1611.01578`

[18] Geoffrey F. Miller, Peter M.Todd and Shailesh U. Hegde. *Designing Neural Networks using Genetic Algorithms.* International Conference on Genetic Algorithms, 1989.
`https://dl.acm.org/citation.cfm?id=94034`

[19] Masanori Suganuma, Shinichi Shirakawa and Tomoharu Nagao. *A Genetic Programming Approach to Designing Convolutional Neural Network Architectures.* Genetic and Evolutionary Computation Conference, 2017.
`https://arxiv.org/abs/1704.00764`

[20] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams and Nando de Freitas. *Taking the Human Out of the Loop: A Review of Bayesian Optimization.* Deep Learning Summit Boston, 2015.
`https://ieeexplore.ieee.org/document/7352306`

[21] Javier Gonzalez and Zhenwen Dai. *Gpyopt: A Bayesian Optimization Framework in Python.* 2016.
http://sheffieldml.github.io/GPyOpt/index.html

[22] Jasper Snoek. *Spearmint Bayesian Optimization Codebase.* 2019.
https://github.com/HIPS/Spearmint

[23] James Bergstra, Daniel Yamins and David D. Cox. *Hyperopt: Distributed Asynchronous Hyper-parameter Optimization.* 2013.
http://hyperopt.github.io/hyperopt

[24] Marius Lindauer, Katharina Eggensperger, Matthias Feurer, Stefan Falkner, André Biedenkapp and Frank Hutter. *SMAC v3: Algorithm Configuration in Python.* 2017.
https://github.com/automl/SMAC3

[25] Hanwen Liang, Shifeng Zhang, Jiacheng Sun, Xingqiu He, Weiran Huang, Kechen Zhuang and Zhenguo Li. *DARTS+: Improved Differentiable Architecture Search with Early Stopping.* 2019.
https://arxiv.org/abs/1909.06035

[26] Ronald J. Williams. *Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning.* Machine Learning, 8(3): 229-256, May 1992.
https://doi.org/10.1007/BF00992696

[27] Christopher Watkins. *Learning From Delayed Rewards.* 1989.
http://www.cs.rhul.ac.uk/~chrisw/thesis.html

[28] Julian F. Miller. *Cartesian Genetic Programming: Its Status and Future.* In Genetic Programming and Evolvable Machines, 2019.
https://doi.org/10.1007/s10710-019-09360-6

[29] Jasper Snoek, Hugo Larochelle and Ryan P. Adams. *Practical Bayesian Optimization of Machine Learning Algorithms.* Advances in Neural Information Processing Systems, 2012.
https://arxiv.org/abs/1206.2944

[30] James Bergstra, Daniel Yamins and David D. Cox. *Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures.* International Conference on Machine Learning, 2013.
http://proceedings.mlr.press/v28/bergstra13.html

[31] Frank Hutter, Holger H. Hoos, Kevin Leyton-Brown and Kevin P. Murph. *SMAC v3: Algorithm Configuration in Python.* Annual conference on Genetic and evolutionary computation, 2009.
https://doi.org/10.1145/1569901.1569940

[32] Yi-Qi Hu, Yang Yu, Wei-Wei Tu, Qiang Yang, Yuqiang Chen and Wenyuan Dai. *Multi-Fidelity Automatic Hyper-Parameter Tuning via Transfer Series Expansion.* Conference on Artificial Intelligence, 2019. https://doi.org/10.1609/aaai.v33i01.33013846

[33] Aaron Klein, Stefan Falkner, Jost T. Springenberg and Frank Hutter. *Learning Curce Prediction with Bayesian Neural Networks.* International Conference on Learning Representations, 2017.
https://openreview.net/forum?id=S11KBYclx

[34] Tobias Domhan, Jost Tobias Springenberg and Frank Hutter. *Speeding Up Automatic Hyperparameter Optimization of Deep Neural Networks by Extrapolation of Learning Curves.* Proceedings of the 24th International Joint Conference on Artificial Intelligence, 2015.
https://www.aaai.org/ocs/index.php/IJCAI/IJCAI15/paper/view/11468

[35] Tao Wei, Changhu Wang, Yong Rui and Chang Wen Chen. *Network Morphism.* International Conference on Machine Learning, 2016.
https://arxiv.org/abs/1603.01670

[36] Frank Hutter, Holger H. Hoos and Kevin Leyton-Brown. *Sequential Model-Based Optimization for General Algorithm Configuration.* International Conference on Learning and Intelligent Optimization, 2011.
https://www.cs.ubc.ca/~hutter/papers/10-TR-SMAC.pdf

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun *Deep Residual Learning for Image Recognition.* Conference on Computer Vision and Pattern Recognition, 2016.
https://arxiv.org/abs/1512.03385

[38] Diederik P. Kingma, Jimmy Ba. *Adam: A Method for Stochastic Optimization.* International Conference on Learning Representations, 2015.
https://arxiv.org/abs/1412.6980

[39] Tianqi Chen, Carlos Guestrin. *XGBoost: A Scalable Tree Boosting System.* International Conference on Knowledge Discovery and Data Mining, 2016.
https://arxiv.org/abs/1603.02754