Neural Architecture Search and tinyML

Daniel Duclos-Cavalcanti

***Abstract*—There is no denying the increasing success that Deep Neural Networks (DNNs) have displayed across various tasks such as image classification, speech recognition, machine translation and many others. This progress can be attributed largely due to *architecture engineering*, which is a process that requires immense domain expertise, intuition and some amount of trial and error. This is not ideal as it is both time consuming and error-prone. Neural Architecture Search (NAS) rises as the next logical step aiming to both automate architecture discovery and further the understanding of the inner workings of DNNs. This field has grown remarkably within the last 5 years and among the many challenges of NAS, there are concerns regarding computational costs and time feasability. This however changes within the context of *tinyML*, which is an expanding field at the intersection of machine learning and embedded systems. Due to the resource constrained conditions involved in most embedded devices and tinyML applications, the stages concerning training and performance evaluation are substantially faster in comparison to the more complex models used in the research of NAS. Thus presenting an interesting opportunity to explore NAS applications within a different paradigm. Throughout this work, an overview of existing research in NAS, specifically concerned with the use of evolutionary algorithms and reinforcement learning will be presented, as well as highlighting relevant applicabilities to tinyML.**

I. Introduction

The advancement of deep learning, although extremely beneficial, has also caused a continuous demand for architecture design. This coupled with a growing model complexity still requires ample time and expert knowledge to continue this progress.

Finally, after the work in [1] proposed by Google, it was shown for the first time that NAS algorithms have the potential to find models that rival the current state of the art. However, done so in an automated fashion, minimizing human participation. Since then, many different methods have appeared.

To better visualize NAS, one can categorize it within three dimensions [2]:

- **Search Space**: defines all the possible architectures that can in principle be considered.
- **Search Strategy**: defines how the *search space* is explored by the algorithm.
- **Performance Estimation Strategy**: defines how performance is evaluated at every architecture iteration.

A. Search Space

NAS is an optimization problem, whose search space is the defining factor to its complexity. The smaller the search space is, the faster the search may converge, as well as requiring less computational resources. This comes at the cost of less freedom to explore unseen architectures and also possibly limiting the complexity of the design.

The simplest approach is to define an architecture as a *chain-structured neural network*, which essentially consists of a sequence of layers whose inputs are the output of their preceding layer. In this case the space is parametrized by maximum number of layers, type of operations per layer, and hyperparameters conditioned on the chosen operation.

The next step consists then of including more modern design elements such as skip connections, which has already been seen in [1] and [3]. This allows to build more complex *multi-branch networks*, which cannot be described as simple sequential layer chaining, but as a structure where each layer's input is a function of previous layer outputs. This increases significantly the degrees of freedom of architecture design, which leads to a much larger search space.

Another predominant trend includes the search within *cells* or *blocks*, initially considered by works such as [4] and [5]. The proposal here is break-off architectures in cells, such that the search space is then designated within a single cell per time. This drastically reduces the search space, as there are substantially less layers within a cell in comparison to an entire architecture. Additionally, subdivision of architectures into blocks is considered a good design practice, which also enables easy transferability of blocks to other data sets.

This drives then the discussion between *micro-architectures* versus *macro-architectures*. The macro perspective attemps to determine how cells should be connected and how many are needed to build a model. On the other hand, the micro perspective aims to find the optimal structure for each cell. Ideally, both viewpoints should be optimized jointly.

B. Search Strategy

II. State of the Art

In this section an analysis of the available literature on the topic is done. This section may be split or subdivided into several sections or subsections.

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

III. Conclusion

Put the conclusions of the work here. The conclusion is like the abstract with an additional discussion of open points.

REFERENCES

[1] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[2] M. Wistuba, A. Rawat, and T. Pedapati, "A survey on neural architecture search," *arXiv preprint arXiv:1905.01392*, 2019.

[3] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, "Large-scale evolution of image classifiers," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 2902–2911. [Online]. Available: https://proceedings.mlr.press/v70/real17a.html

[4] Z. Zhong, J. Yan, W. Wu, J. Shao, and C.-L. Liu, "Practical block-wise neural network architecture generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2423–2432.

[5] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.