Improving Local Search for Neural Architecture Search

Meyssa Zouambi[1], Clarisse Dhaenens[1], Julie Jacques[1,2]

[1] Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

[2] Faculté de Gestion, Economie et Sciences, Lille Catholic University, Lille, France

{meyssa.zouambi, clarisse.dhanaens}@univ-lille.fr; julie.jacques@univ-catholille.fr

Keywords: *Neural Architecture Search, Local Search, Optimization.*

1 Introduction

Neural architecture search (NAS) is a set of techniques that aims at the automation of architecture engineering; it is a subfield of AutoML, which encloses all the methods that reduce human interventions in the design of machine learning models like hyperparameter optimization and meta learning.

Local search (LS), on the other hand, is a famous metaheuristic that has been around for many years. It is extensively used for optimization problems due to its simplicity and efficiency.

Recent works that studied local search approaches for NAS proved that such a method is competitive with state-of-the-art methods. In [2], authors explore the theoretical characterization of the landscape and its effects on the performance of local search. It demonstrates that hill climbing -the simplest form of LS- can outperform many popular state-of-the-art algorithms when the noise in the evaluation pipeline is reduced to a minimum.

Another work that confirms these results is found in [1], where LS is employed in a multi-objective context. It shows that a simple, parameter-less local search competes with state-of-the-art evolutionary algorithms, even up to thousands of evaluated architectures in the multi-objective setting. LS is therefore a method that is very easy to implement yet yields competitive performances against more complex algorithms. On top of that, it can naturally exploit strategies that accelerate global search time like network morphism and weight inheritance.

2 Method description

In the context of NAS, a solution is an architecture represented by the set of operations/parameters that define it, and the search space is made up of all possible architecture configurations.

Our algorithm (LS-Weights) is an improved version of hill-climbing specifically designed for NAS. Hill-climbing updates the current solution as soon as it finds a better one in the neighborhood. For this reason, the speed of this method relies on the order in which the neighbors are evaluated.

In order to avoid sampling too many neighbors that perform badly, it is important to notice that certain operations/edges are more relevant than others inside an architecture.

For this reason, choosing for example between convolution operations vs zero-operations should not have the same weight during the sampling. We thought that adding probabilities to the sampling procedure, when creating the neighborhood, would lead to improve solutions more quickly during the search process.

In our method, we rank the operations based on their role in the network. Operations that have a main role such as convolution operations have the highest rank, while secondary operations such as skip connections and none operations should have a lower rank. We assign higher probabilities to the operations with higher ranks to quickly assess the neighbors containing them during the search.

3 Results and conclusion

We evaluate our method on NASBench-201, which is a popular NAS benchmark that consists of 15,625 unique architectures, with precomputed training, validation, and test accuracies for 200 epochs on CIFAR-10, CIFAR-100, and ImageNet-16-120.

We define that two solutions from this benchmark are neighbors if they differ by exactly one operation in one of their edges. We use the validation accuracy as the search metric, we also report the final test accuracy of the model selected. Since the best architectures on the validation set are not necessarily the same as the test set, we use the oracle of the validation accuracy as an upper bound.

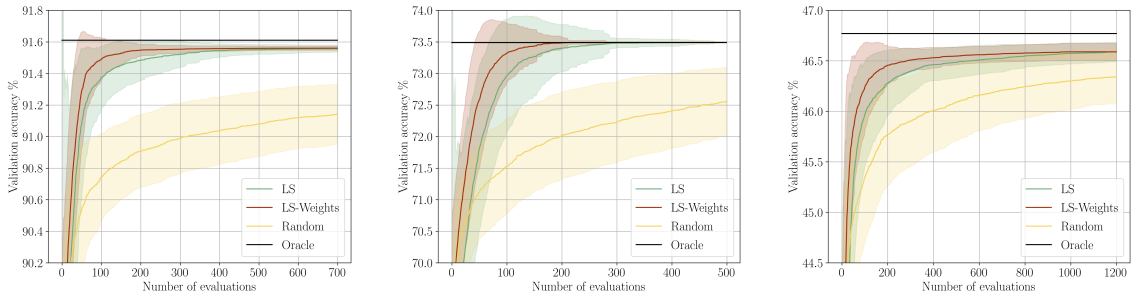
Table 1 reports the mean and standard deviation of the number of evaluated samples during the search. The number of sampled architectures is a strong indicator of speed in NAS.

We can see in Figure 1 that our method significantly improve local search's computational time while reaching similar accuracies.

As a perspective, we aim at setting the ranks of operations automatically during the search, by learning the contribution of different types of operations inside a network in an *online* manner.

TAB. 1: The table indicates the mean and std of the number of evaluated architectures before convergence.

Method	**Cifar10**	**Cifar100**	**ImageNet**
LS	415±377.77	106±74.27	615±417.71
LS-Weights	**203±216.48**	**66±40.94**	**361±286.17**



(a) Cifar10 (b) Cifar100 (c) ImageNet16-120

FIG. 1: The evolution of the validation accuracy across the number of sampled architectures.

References

[1] Tom Den Ottelander, Arkadiy Dushatskiy, Marco Virgolin, and Peter AN Bosman. Local search is a remarkably strong baseline for neural architecture search. In *International Conference on Evolutionary Multi-Criterion Optimization*, 2021.

[2] Colin White, Sam Nolen, and Yash Savani. Exploring the loss landscape in neural architecture search. *arXiv preprint arXiv:2005.02960*, 2020.