Master Thesis - Expose

Simon Neumeyer

March 31, 2021

1 Introduction

State-of-the-art neural networks for image classification inhibit an increasing complexity in their topology which implies the need for neural architecture search (NAS), the process of automatically searching the topology of a neural network [LSV+17]. NAS methodologies can be characterized by what search space, search strategy and evaluation strategy they are using [EMH19]. Much of the recent work incorporates empirical knowledge about well performing handcrafted structures into the search space, such as filters, pooling layers or hierarchy in general. [EMH19] note that this fact prevents the discovery of fundamentally new topologies. While they therefore consider it important to apply NAS to less explored domains than image classification, it might also be worthy to incorporate less empirical knowledge into the search space. Regarding the search strategy, evolutionary approaches have already been applied long before the recent engagement in NAS and are still popular [EMH19]. More recently approaches like bayesian optimization, reinforcement learning and quite recently differentiable architecture search [LSV+17] have achieved state-of-the-art results. Differentiable architecture embed architectures in a continuous search space and use gradient descent for optimization. A benefit of differentiable architecture search is that it works outside the context of blackbox optimization that requires the costly evaluation of searched architectures, and therefore is very efficient. It further comes almost inherently with weight sharing, another way of reducing computation cost. In this work we want to apply differentiable architecture search in the image classification domain on less typical search spaces in the regard that these search spaces do not inhibit well-known structures like filters, pooling layers and so forth but consist mainly of densly connected layers. Doing so enables us to examine the performance of empirically unfamiliar network structures.

2 Related Work

[LSV+17] searches an hierarchical search space that consists of several identical building blocks, so called cells. The space the cell is being searched in can be represented by a set of directed acyclic graphs (DAG) with a fixed number of nodes. Entries from a fixed set of feature maps respectively operations are being placed on the edges of the DAG. Continuouity is achieved by applying all operations on each edge and aggregating the outputs with a scalar weight. These architecture weights are then optimized together with the network weights in

a bi-level optimization. [XZLL18] improves results of [LSV+17] by leaving the search space and evaluation strategy as is but instead of evaluating the continuous architecture evaluating a discrete architecture sampled from a distribution that is parametrized by the architecture weights and optimized to sample well-performing architectures with higher probability. [LZS+19] elaborates on [LSV+17] by regularizing the heavy usage of skip-connections, [CXWT19] by targeting the bias induced due to training only a single cell instead of the final architecture consisting of several stacked cells. For networks in the continuous search space of differentiable architecture search approaches often consisting of a high number of parameters, the networks might even cross the boundaries for fitting into a single GPU. To overcome that restriction and so heavily reducing the computation cost, [CZH18] are describing an approach that reduces the network size during weight optimization by sampling a discrete architecture from the continuous one-shot model using weight binarization. Dependent on the methodologies examined in this work, a similar approach might be necessary.

3 Goal

We want to apply differentiable architecture search methods in the image classification domain with few priors on the network structure. This might enable us to discover and study the performance of network structures different from those being typically applied. In detail we address the following research questions:

- Which possible ways of implementing differentiable architecture search exist and how do they differ in efficiency, performance and stability?
- Is the performance of any of our networks better than the performance of a densly connected neural network with the same amount of layers?
- How big does the search space need to be to find architectures with good performance?
- Is there a correlation between certain kind of DAGs (depth, number of connections) and the performance of the corresponding network?
- Is the correlation independent from the dataset?
- Is there a possibility to define the search space in a way to contain well-reknown structures like filters or pooling?

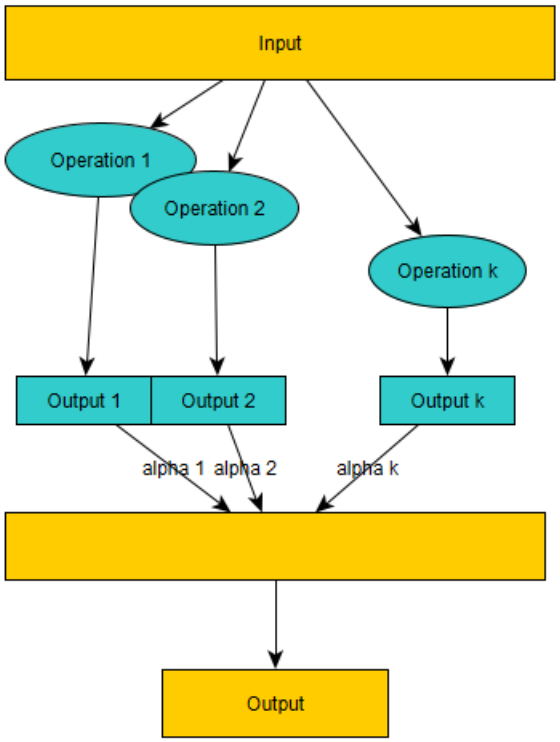
4 Methodology

4.1 Setting

We consider k building blocks, the building blocks each connected to the input and output layer and weighted with an architecture parameter $\alpha = (\alpha_1, \ldots, \alpha_k)$. Each building block itself consists of l layers that are connected according to the structure of a corresponding DAG with l vertices. The inputs for each layer are aggregated, e.g. by summation. The weights between layers i and j, $1 <= i < j <= l$, are shared amongst all building blocks. Setting the connection between two layers in a building block either active, in case there is

an edge between the vertices of the corresponding DAG, or inactive is achieved by applying a mask on the network weights.

Figure 1: Continuous neural network genotype

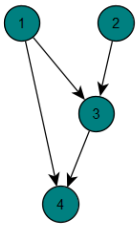


4.2 Graph sampling

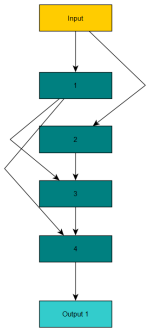
In a first step we choose only k DAGs and no sampling takes place. In a next step after each evaluation phase we sample k DAGs according to a density function applied to a space containing all DAGs that apply to certain criterias such as number of vertices. Note that without further restrictions on the DAG, the search space grows in $\mathcal{O}(2^{n^2})$ in the number of nodes n. With restricting the depth of skip-connections we still grow exponentially in n.

4.3 Optimization

The network weights and the architecture weights get trained taking turns. We refer to the two differnt training phases as weight optimization and architecture optimization. During architecture optimization backpropagation on the architecture parameters provide feedback on how well the different building blocks perform compared to each other. We use this feedback to update the corresponding sampling probabilities. It is possible to either use the same split of data for weight and architecture optimization both or different splits, it needs to be figured out which approach is better suited.



(a) Directed acyclic graph with 4 vertices



(b) building block corresponding to the DAG

Figure 2: Sampled DAG with its corresponding neural building block

4.4 Hyper parameters

The following hyper parameters are involved:

- number k of building blocks
- duration and dataset split of weight respectively architecture optimization
- for both weight and architecture optimization general hyper parameters like learning rate and batch size

References

[CXWT19] Xin Chen1, Lingxi Xie2, Jun Wu1, and Qi Tian. Progressive darts bridging the depth gap between search. 2019.

[CZH18] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. 12 2018.

[EMH19] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey, 2019.

[LSV+17] Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical representations for efficient architecture search. 11 2017.

[LZS+19] Hanwen Liang, Shifeng Zhang, Jiacheng Sun, Xingqiu He, Weiran Huang, Kechen Zhuang, and Zhenguo Li. Darts+: Improved differentiable architecture search with early stopping. 9 2019.

[XZLL18] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: Stochastic neural architecture search. 12 2018.