UNDERSTANDING NEURAL ARCHITECTURE SEARCH BY ITS ARCHITECTURE PARAMETERS

Nicholas Roberts, Yingyu Liang, Frederic Sala
University of Wisconsin-Madison
{nick11roberts, yliang, fredsala}@cs.wisc.edu

ABSTRACT

Neural Architecture Search (NAS) has shown promising empirical results as an AutoML technique for accelerating the design process of of deep neural networks. Specifically, NAS with weight-sharing is a popular and state-of-the-art family of NAS algorithms which simultaneously optimize multiple architectures through the use of parameter sharing. In this work, we make initial steps toward a theoretical understanding of the properties of the *architecture parameters* of NAS with weight-sharing – a crucial consideration when deriving a final discrete architecture post-search. Prior theoretical work on NAS with weight-sharing studies the generalization ability of weight-sharing under bilevel optimization and ignores practical desiderata of the architecture parameters themselves: (i) architecture parameters for a given layer are usually constrained to the simplex and (ii) search spaces are typically discrete, so a discrete architecture is derived from the mixture parameters post-search by taking the $\arg\max$. In this work, we use the setting of a previous analysis of activation function search using weight-sharing in two-layer networks to study the behavior of simplex-constrained architecture parameters. We find that in a search space of two activation functions, weight-sharing recovers architecture parameters that discretize to the optimal final architecture in the discrete search space. In a discrete search space of three or more activation functions, we derive a sufficient conditions for the same behavior to hold. This in turn gives rise to an initial theoretical understanding of $\arg\max$ discretization and for architecture ranking.

Weight-Sharing, Neural Architecture Search, AutoML

1 INTRODUCTION

The design of deep neural network architectures for a given problem requires a significant investment in compute resources, domain experts, and trial-and-error. The field of automated machine learning (AutoML) aims to ease the burden of these types of investments by automating various aspects of the machine learning pipeline; in particular, Neural Architecture Search (NAS) has shown to be a promising direction in AutoML toward automating the neural network design process. Weight-sharing is a popular family of NAS algorithms which involves training all architectures in a NAS search space simultaneously while sharing parameters between them (Pham et al., 2018). NAS with weight-sharing has continued to demonstrate state-of-the-art empirical results on standard image classification and language tasks (Liu et al., 2019; Li & Talwalkar, 2020; Li et al., 2021), as well as in under-explored domains which can enjoy an even greater potential benefit from this line of work (Roberts et al., 2021).

Recently, theorists have made progress toward understanding generalization in NAS with weight-sharing (Khodak et al., 2020; Oymak et al., 2021), though connections to the discretization and ranking properties of weight-sharing methods remain theoretically under-studied. In this work, we theoretically investigate the properties of the architecture parameters after search and their connections to discretization. We use a refinement of the problem setting of Oymak et al. (2021)—activation function selection for two-layer networks. The setting of Oymak et al. (2021) does not impose realistic structure on the architecture parameters, whereas weight-sharing methods typically constrain the architecture parameters of each layer to the simplex as a convex relaxation of the otherwise discrete search space. We show that these added assumptions yield sufficient conditions under which the architecture obtained by evaluating every discrete architecture in the search space using the shared weights and the architecture obtained by discretizing the architecture parameters are the same.

2 Related work

Neural Architecture Search with Weight-Sharing NAS with Weight-Sharing was first introduced by Pham et al. (2018) with ENAS, which learns a controller model trained using policy gradients to modify a child network with parameter shared between architecture choices. Soon after, DARTS emerged as the canonical state-of-the-art weight-sharing method which posed a continuous relaxation to the a bilevel formulation of the NAS problem, which could be learned end-to-end using gradient-based optimization of both the shared-weights as well as the architecture parameters (Liu et al., 2019). In particular, they posed NAS as a search problem over computation graphs, where each edge of the computation graph corresponds to a decision between several layer choices, and each node represents a state or hidden representation in network. At each edge of the computation graph, continuous relaxation of the NAS problem is formulated as a learnable convex combination of the outputs of each of the layer choices for that edge, which is depicted in Figure 1. In DARTS, this learnable convex combination is parameterized by a softmax activation applied to a vector of architecture parameters. The bilevel optimization procedure in DARTS involves alternately applying gradient updates to the architecture parameters using the validation set and applying gradient steps to the shared weights using the training set. After the bilevel training procedure terminates, a discrete architecture is derived by taking the $\arg\max$ over the architecture parameters at each edge, at which point this architecture is re-trained from scratch. Later, Li et al. (2021) proposed GAEA, which replaces the softmax parameterization of the architecture parameters with parameters lying directly on the simplex and updated using exponentiated gradient descent. This leads to faster convergence to sparse architecture parameters and improved empirical performance over DARTS and other methods. Beyond selecting the best architecture, an important consideration for resource-constrained NAS is whether a search method can rank the architectures in its search space by their generalization performance. The goal of doing this would be to identify the best architecture subject to certain architecture constraints that might be difficult to encode into the search space a priori. Despite the success of NAS with weight-sharing, some have found that weight-sharing methods lead to poor architecture selection and poor ranking performance, i.e., the ranking given by the relative performance of each architecture evaluated using the shared weights is, in practice, different from the ranking induced by training each architecture from scratch (Yu et al., 2020).

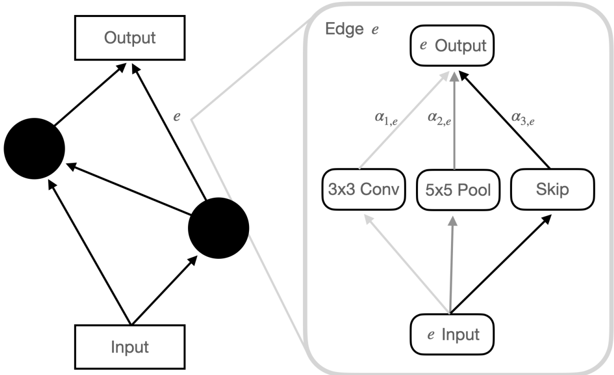

Figure 1: A simplified computation graph used in weight-sharing methods. Each edge is parameterized as a convex combination of the outputs of each layer choice at that edge. Here, the architecture parameters for edge e are the mixture weights $\boldsymbol{\alpha} = \{\alpha_{i,e}\}_{i=1}^{3} \in \Delta^2$ and Δ^2 denotes the unit simplex with 3 vertices.

Weight-sharing theory Recently, NAS with weight-sharing has garnered interest from the learning theory community. In particular, Khodak et al. (2020) proposes a simple setting for the theoretical analysis of weight-sharing: feature map selection for single-layer networks. In this setting, they provide generalization bounds on weight-sharing under the typical bilevel optimization scheme as well as a sample complexity justification for the use of bilevel optimization as opposed to single-level ERM. More recently, Oymak et al. (2021) proposed a multi-layer setting to study weight sharing for the problem of activation function selection, similarly under bilevel optimization. They argue that the lower-level optimization problem of optimizing the shared weights can always achieve zero training loss due to overparameterization, which itself motivates the use and study of generalization in weight-sharing with bilevel optimization. They obtain generalization bounds for their setting in the lazy-training regime and show that the search algorithm itself identifies the best model/architecture pair in the search space.

3 Preliminaries, problem formulation, and context

Data setting We will use a similar data setting as Oymak et al. (2021), but we make an additional assumption about the data generating process. Denote $(\mathbf{x}, y) \sim \mathcal{D}$ where $\mathbf{x} \in \mathcal{X}$ and $y = G(x) \in \mathcal{Y}$ as the data distribution of input features and labels, where G is a deterministic labeling function. Define the population risk as $\mathcal{L}(f) = \mathbb{E}_{\mathcal{D}}[\ell(y, f(\mathbf{x}))]$ for a given loss function ℓ and hypothesis $f : \mathcal{X} \to \mathcal{Y}$.

Hypothesis class and optimization specifics In this work, we impose more structure on the architecture parameters than Oymak et al. (2021), who allows the space of architecture parameters to be any subset of the unit ℓ_1 ball. Denote our architecture parameter vector as $\boldsymbol{\alpha} \in \Delta^{h-1} \subseteq \mathbb{R}^h$ where Δ^{h-1} is the unit simplex with h vertices, which is indeed a subset of the unit ℓ_1 ball, so the corresponding results of Oymak et al. (2021) still hold in our setting.

We will use the same two-layer activation function selection setting as Oymak et al. (2021). Specifically, for some activation function σ, we consider hypotheses of the form

$$\mathcal{F}_\sigma = \{f_\sigma(\cdot, \mathbf{W}) | f_\sigma(\mathbf{x}, \mathbf{W}) = \mathbf{v}^\top \sigma(\mathbf{W}\mathbf{x}), \mathbf{x} \in \mathbb{R}^k, \mathbf{W} \in \mathbb{R}^{k \times d}, \mathbf{v} \in \mathbb{R}^d\}.$$

Denote our convex combination of h activation functions using architecture parameters $\boldsymbol{\alpha}$ as $\sigma_{\boldsymbol{\alpha}}(z) = \sum_{i=1}^h \boldsymbol{\alpha}_i \sigma_i(z)$. Then, the extended hypothesis class over the architecture parameters and the shared weights is denoted as $\mathcal{F}_{\sigma_{\boldsymbol{\alpha}}}$. Following Oymak et al. (2021), we will consider binary classification with $y \in \{-1, +1\}$ and population loss

$$\mathcal{L}(f) = \mathcal{L}(f, y) = \frac{1}{2}\mathbb{E}[(y - f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{x}, \mathbf{W}))^2]$$

although our results extend to real-valued labels as well.

4 A geometric understanding of architecture parameters

We will now summarize our main result and the assumptions needed in our analysis. We assume that we obtain optimal shared weights $\mathbf{W}^*$ and architecture parameters $\boldsymbol{\alpha}$ from some bilevel optimizer.

Informal statement of the main result Our main result is stated informally as follows: for any pair of discrete architectures given by σ_i, σ_j where $\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_j}(\mathbf{W}^*))$, and under certain sufficient conditions on the population risks between certain continuous architectures on the boundary of the unit simplex and the discrete architectures $f_{\sigma_i}, f_{\sigma_j}$, we have that $\boldsymbol{\alpha}_j < \boldsymbol{\alpha}_i$.

This result has several immediate practical consequences. Namely, after search, the architecture parameters are typically discretized, i.e., the architecture $f_{\sigma_k} : k = \arg\max_i \boldsymbol{\alpha}_i$ is obtained and retrained from scratch. If the sufficient conditions (which will be derived in a later subsection) do not hold, then the discretization process might result in a potentially suboptimal architecture. Furthermore, the ability for the shared weights to rank architectures by their optimality under standalone weights is an important consideration for resource constrained settings. In these settings, the search space under consideration might contain architectures which do not meet some criterion for the downstream task – e.g. constraints on the parameter count, latency considerations, or other practical considerations. Ranking provides a solution to this problem by allowing for the best architecture to be selected subject to certain practical constraints. This is typically done by evaluating each discrete architecture in the search space using the shared weights, selecting the architecture with the best generalization performance according to the shared weights subject to the practical constraints, and retraining the weights from scratch. Our result suggests that under sufficient conditions hold for every pair of architectures in the ranking, the architecture parameters themselves can be examined and ranked instead of evaluating every architecture in the discrete search space.

Required assumptions In our analysis, we make several assumptions regarding the relative optimality of each of the discrete architectures, the geometry of the continuous architecture parameters returned by search, and on the relationship between the model returned by search and the label generating process. Without loss of generality with regard to the indexing of the h activation functions in our search space, we assume the following statement about the ordering of the population risk (which is given by $\mathcal{L}(f_\sigma(\mathbf{W})) = \mathcal{L}(f_\sigma(\mathbf{W}), y) = \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(y - f_\sigma(\mathbf{x}, \mathbf{W}))^2]$).

Assumption 1. *We have a total order on the architecture ranking with respect to the population risk, as evaluated using the shared weights. Namely, we have*

$$\mathcal{L}(f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*)) < ... < \mathcal{L}(f_{\sigma_h}(\mathbf{W}^*)).$$

Assumption 1 is mostly realistic, as it is assumed in practice that when evaluating the architecture ranking using the shared weights, architectures will perform differently. We also require that the architecture parameters all be nonzero. Formally, we assume the following.

Assumption 2. *The architecture parameters returned by the search procedure lie on the interior of the unit simplex $\boldsymbol{\alpha} \in \operatorname{int}_{\mathbb{R}^h} \Delta^{h-1}$.*

Assumption 2 is realistic and is generally assumed in practice for gradient based search methods, which is exactly what necessitates the typical $\arg\max$ discretization procedure before retraining the final searched architecture from scratch. Finally, we make the strong assumption that the labels are generated by some model in the extended hypothesis class $\mathcal{F}_{\sigma_{\boldsymbol{\alpha}}}$, and that this is the exact model returned by the search procedure.

Assumption 3. *We assume realizability. In other words, under perfect optimization, the shared weights and architecture parameters returned by the search procedure are identical to the label generating process, $y = f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{x}, \mathbf{W}^*)$.*

Note that Assumption 3 is unrealistic and would somewhat defeat the purpose of post-search discretization and retraining since it would mean that the model returned by search has zero population risk. We nonetheless make this assumption for ease of exposition. We leave removing this assumption to future work. In the following subsections, we will provide formal statements and proofs of our main result with increasing levels of generality. We begin with search spaces of two and three activation functions to build intuition, and along the way, we will add components to obtain the result for more generic activation function search spaces with finitely many discrete architectures.

4.1 Two activation functions

We now present a formal statement and proof of our result in the case of a search space of two activation functions. Note that in this simplified setting, we do not require additional sufficient conditions nor Assumption 2 until we proceed to more general cases.

Theorem 1 (Maximum $\boldsymbol{\alpha}$ for two activations)**.** *Let $h = 2$ and under Assumptions 1 and 3, let $\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*))$, then it holds that $\boldsymbol{\alpha}_2 < \boldsymbol{\alpha}_1$.*

Proof. Begin with $\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*))$.

$$\begin{aligned}
\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) &= \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(f_{\sigma_1}(\mathbf{x}, \mathbf{W}^*) - y)^2] \\
&\overset{(1)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(f_{\sigma_1}(\mathbf{x}, \mathbf{W}^*) - f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{x}, \mathbf{W}^*))^2] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}(\boldsymbol{\alpha}_1\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) + \boldsymbol{\alpha}_2\sigma_2(\mathbf{W}^{*\top}\mathbf{x})))^2] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[((1-\boldsymbol{\alpha}_1)\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \boldsymbol{\alpha}_2\mathbf{v}\sigma_2(\mathbf{W}^{*\top}\mathbf{x}))^2] \\
&\overset{(2)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(\boldsymbol{\alpha}_2\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \boldsymbol{\alpha}_2\mathbf{v}\sigma_2(\mathbf{W}^{*\top}\mathbf{x}))^2] \\
&= \boldsymbol{\alpha}_2^2\frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}\sigma_2(\mathbf{W}^{*\top}\mathbf{x}))^2] \\
&= \boldsymbol{\alpha}_2^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\sigma_2}(\mathbf{W}^*))
\end{aligned}$$

where (1) holds by Assumption 3 and (2) holds since $\boldsymbol{\alpha} \in \Delta^1$. By the same argument, we have that

$$\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*)) = \boldsymbol{\alpha}_1^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\sigma_2}(\mathbf{W}^*)).$$

Finally, we obtain the following

$$\begin{gathered}
\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*)) \\
\Rightarrow \boldsymbol{\alpha}_2^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\sigma_2}(\mathbf{W}^*)) < \boldsymbol{\alpha}_1^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\sigma_2}(\mathbf{W}^*)) \\
\Rightarrow \boldsymbol{\alpha}_2 < \boldsymbol{\alpha}_1
\end{gathered}$$

as required. □

4.2 Three activation functions and the simplex interior factorization

Next, we provide a formal statement and proof of the slightly more general setting of three activation functions. In this setting, we employ a key lemma about the factorization of points on the interior of the unit 2-simplex, and we use this to derive a sufficient condition under which the result to holds for three activation functions.

Lemma 2 (Factorization of $\operatorname{int}_{\mathbb{R}^3}\Delta^2$)**.** *Let $\boldsymbol{\alpha} \in \operatorname{int}_{\mathbb{R}^3}\Delta^2$. Then $\boldsymbol{\alpha}_2$ and $\boldsymbol{\alpha}_3$ can be expressed in terms of $(1-\boldsymbol{\alpha}_1)$ as follows:*

$$\boldsymbol{\alpha}_2 = (1-\boldsymbol{\alpha}_1)\boldsymbol{\beta} \qquad \boldsymbol{\alpha}_3 = (1-\boldsymbol{\alpha}_1)(1-\boldsymbol{\beta})$$

with $\boldsymbol{\beta} \in (0,1)$.

Proof. Beginning with $\boldsymbol{\alpha}_2 = (1-\boldsymbol{\alpha}_1)\boldsymbol{\beta}$, we have $\boldsymbol{\beta} = \frac{\boldsymbol{\alpha}_2}{1-\boldsymbol{\alpha}_1}$ and $1-\boldsymbol{\beta} = \frac{1-\boldsymbol{\alpha}_1-\boldsymbol{\alpha}_2}{1-\boldsymbol{\alpha}_1} = \frac{\boldsymbol{\alpha}_3}{1-\boldsymbol{\alpha}_1}$ which both hold because $\|\boldsymbol{\alpha}\|_1 = 1$ and $\boldsymbol{\alpha}_i \in (0,1) \quad \forall i \in \{1,2,3\}$ which are consequences of $\boldsymbol{\alpha} \in \operatorname{int}_{\mathbb{R}^3}\Delta^2$. Finally, $\boldsymbol{\beta} = \frac{\boldsymbol{\alpha}_2}{1-\boldsymbol{\alpha}_1} = \frac{\boldsymbol{\alpha}_2}{\boldsymbol{\alpha}_2+\boldsymbol{\alpha}_3} \in (0,1)$ holds for the same reasons. □

Lemma 2 allows us to write all of the architecture parameters in terms of one parameter, which admits the required factorization for our result to hold for three activation functions. Next, we provide a statement of the sufficient condition that we will derive in Theorem 4.

Condition 3 (f_{σ_1} is sufficiently far from a convex combination of f_{σ_2} and f_{σ_3})**.** *Let $h = 3$. Then for $\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*))$ and parameters $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2 \in (0,1)$,*

$$\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\boldsymbol{\beta}_1\sigma_2+(1-\boldsymbol{\beta}_1)\sigma_3}(\mathbf{W}^*)) \geq \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*), f_{\boldsymbol{\beta}_2\sigma_1+(1-\boldsymbol{\beta}_2)\sigma_3}(\mathbf{W}^*)).$$

Intuitively, Condition 3 says that we need $f_{\sigma_1}(\mathbf{W}^*)$ to be far enough away in population risk from some continuous architecture specified by architecture parameters $[0, \boldsymbol{\beta}_1, (1-\boldsymbol{\beta}_1)]^\top \in \operatorname{bd}_{\mathbb{R}^3}\Delta^2$. The parameters $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ in Condition 3 are obtained using their respective Lemma 2 factorizations on $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2$. With this, we go on to show that Condition 3 is sufficient for $\boldsymbol{\alpha}_2 < \boldsymbol{\alpha}_1$ to hold.

Theorem 4 (Maximum $\boldsymbol{\alpha}$ for three activations)**.** *Let $h = 3$ and under Assumptions 1, 2, and 3, let $\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*))$. Then Condition 3 is sufficient for $\boldsymbol{\alpha}_2 < \boldsymbol{\alpha}_1$ to hold.*

Proof. Beginning with $\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*))$, we have

$$\begin{aligned}
\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) &= \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(f_{\sigma_1}(\mathbf{x},\mathbf{W}^*) - y)^2] \\
&\overset{(1)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(f_{\sigma_1}(\mathbf{x},\mathbf{W}^*) - f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{x},\mathbf{W}^*))^2] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) \\
&\quad - \mathbf{v}(\boldsymbol{\alpha}_1\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) + \boldsymbol{\alpha}_2\sigma_2(\mathbf{W}^{*\top}\mathbf{x}) + \boldsymbol{\alpha}_3\sigma_3(\mathbf{W}^{*\top}\mathbf{x})))^2] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[((1-\boldsymbol{\alpha}_1)\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \boldsymbol{\alpha}_2\mathbf{v}\sigma_2(\mathbf{W}^{*\top}\mathbf{x}) - \boldsymbol{\alpha}_3\mathbf{v}\sigma_3(\mathbf{W}^{*\top}\mathbf{x}))^2] \\
&\overset{(2)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[((1-\boldsymbol{\alpha}_1)\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) \\
&\qquad - (1-\boldsymbol{\alpha}_1)\boldsymbol{\beta}_1\mathbf{v}\sigma_2(\mathbf{W}^{*\top}\mathbf{x}) \\
&\qquad - (1-\boldsymbol{\alpha}_1)(1-\boldsymbol{\beta}_1)\mathbf{v}\sigma_3(\mathbf{W}^{*\top}\mathbf{x}))^2] \\
&= (1-\boldsymbol{\alpha}_1)^2\frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(\mathbf{v}\sigma_1(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}(\boldsymbol{\beta}_1\sigma_2(\mathbf{W}^{*\top}\mathbf{x}) + (1-\boldsymbol{\beta}_1)\sigma_3(\mathbf{W}^{*\top}\mathbf{x})))^2] \\
&= (1-\boldsymbol{\alpha}_1)^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\boldsymbol{\beta}_1\sigma_2+(1-\boldsymbol{\beta}_1)\sigma_3}(\mathbf{W}^*))
\end{aligned}$$

where (1) holds by Assumption 3 and (2) holds by Assumption 2 and Lemma 2. The same argument leads to the following for $\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*))$

$$\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*)) = (1-\boldsymbol{\alpha}_2)^2\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*), f_{\boldsymbol{\beta}_2\sigma_1+(1-\boldsymbol{\beta}_2)\sigma_3}(\mathbf{W}^*))$$

Finally, we obtain the following expression

$$\begin{gathered}
\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_2}(\mathbf{W}^*)) \\
\Rightarrow (1-\boldsymbol{\alpha}_1)^2\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\boldsymbol{\beta}_1\sigma_2+(1-\boldsymbol{\beta}_1)\sigma_3}(\mathbf{W}^*)) < (1-\boldsymbol{\alpha}_2)^2\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*), f_{\boldsymbol{\beta}_2\sigma_1+(1-\boldsymbol{\beta}_2)\sigma_3}(\mathbf{W}^*)).
\end{gathered}$$

Notice that Condition 3 implies the following

$$\begin{aligned}
(1-\boldsymbol{\alpha}_1)^2 &< (1-\boldsymbol{\alpha}_2)^2\frac{\mathcal{L}(f_{\sigma_2}(\mathbf{W}^*), f_{\boldsymbol{\beta}_2\sigma_1+(1-\boldsymbol{\beta}_2)\sigma_3}(\mathbf{W}^*))}{\mathcal{L}(f_{\sigma_1}(\mathbf{W}^*), f_{\boldsymbol{\beta}_1\sigma_2+(1-\boldsymbol{\beta}_1)\sigma_3}(\mathbf{W}^*))} \\
&\leq (1-\boldsymbol{\alpha}_2)^2 \\
\Rightarrow \boldsymbol{\alpha}_2 &< \boldsymbol{\alpha}_1.
\end{aligned}$$

Hence Condition 3 is sufficient for $\boldsymbol{\alpha}_2 < \boldsymbol{\alpha}_1$. □

4.3 Beyond three activation functions

We now generalize Lemma 2 to handle any finite number of architectures and provide a more general form of Condition 3, which we derive in our generalization of Theorem 4.

Lemma 5 (Recursive factorization of $\operatorname{int}_{\mathbb{R}^h}\Delta^{h-1}$)**.** *Let $\boldsymbol{\alpha} \in \operatorname{int}_{\mathbb{R}^h}\Delta^{h-1}$. Then $\forall i \in [h]\backslash\{1\}$, $\boldsymbol{\alpha}_i$ can be expressed in terms of $(1-\boldsymbol{\alpha}_1)$ as follows*

$$\boldsymbol{\alpha}_i = (1-\boldsymbol{\alpha}_1)\boldsymbol{\gamma}_i$$

with $\boldsymbol{\gamma} \in \operatorname{int}_{\mathbb{R}^{h-1}}\Delta^{h-2}$.

Proof. $\boldsymbol{\alpha} \in \operatorname{int}_{\mathbb{R}^h}\Delta^{h-1}$ implies that $\boldsymbol{\alpha}_i \in (0,1) \quad \forall i \in [h]$, so we have
$\boldsymbol{\gamma}_i = \frac{\boldsymbol{\alpha}_i}{1-\boldsymbol{\alpha}_1} = \frac{\boldsymbol{\alpha}_i}{\sum_{i=2}^h \boldsymbol{\alpha}_i} \in (0,1)$. Furthermore, $\|\boldsymbol{\gamma}\|_1 = \sum_{j=2}^h \frac{\boldsymbol{\alpha}_j}{\sum_{i=2}^h \boldsymbol{\alpha}_i} = 1$.
This implies that $\boldsymbol{\gamma} \in \operatorname{int}_{\mathbb{R}^{h-1}}\Delta^{h-2}$. □

Lemma 5 is a straightforward generalization of Lemma 2 to the unit simplex in h-dimensions. This admits the required factorization for our generalization of Theorem 4 to search spaces of size h. Indeed, this factorization can be applied recursively to $\boldsymbol{\gamma}$, but we do not need this fact for the proof of Theorem 7. Next, we generalize Condition 3 to handle h activation functions.

Condition 6 (f_{σ_i} is sufficiently far from a convex combination of $\{f_{\sigma_k}\}_{k\in[d]\setminus\{i\}}$)**.** *Let h be the size of the search space. Then for $\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_j}(\mathbf{W}^*))$ and parameters $\boldsymbol{\gamma}, \boldsymbol{\gamma}' \in \operatorname{int}_{\mathbb{R}^{h-1}} \Delta^{h-2}$,*

$$\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{i\}} \gamma_k \sigma_k}(\mathbf{W}^*)) \geq \mathcal{L}(f_{\sigma_j}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{j\}} \gamma'_k \sigma_k}(\mathbf{W}^*)).$$

Condition 6 is a generalization of Condition 3 to an arbitrary number of activation functions. Here, we obtain $[0, \boldsymbol{\gamma}]^\top, [0, \boldsymbol{\gamma}']^\top \in \operatorname{bd}_{\mathbb{R}^h} \Delta^{h-1}$ using the factorizations of $\boldsymbol{\alpha}$ in Lemma 5 with respect to $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$. We now go on to show our main result—Condition 6 is sufficient for $\boldsymbol{\alpha}_j < \boldsymbol{\alpha}_i$ to hold.

Theorem 7 (Maximum $\boldsymbol{\alpha}$ for activation search)**.** *Let $h \geq 2$ and under Assumptions 1, 2, and 3, let $\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_j}(\mathbf{W}^*))$. Then Condition 6 is sufficient for $\boldsymbol{\alpha}_j < \boldsymbol{\alpha}_i$ to hold.*

Proof. Beginning with $\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*))$, we have

$$\begin{aligned}
\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*)) &= \frac{1}{2}\mathbb{E}_{\mathcal{D}}[(f_{\sigma_i}(\mathbf{x}, \mathbf{W}^*) - y)^2] \\
&\overset{(1)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}[(f_{\sigma_i}(\mathbf{x}, \mathbf{W}^*) - f_{\sigma_{\boldsymbol{\alpha}}}(\mathbf{x}, \mathbf{W}^*))^2] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}\left[\left(\mathbf{v}\sigma_i(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}\sum_{k=1}^{h} \boldsymbol{\alpha}_k \sigma_k(\mathbf{W}^{*\top}\mathbf{x})\right)^2\right] \\
&= \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}\left[\left((1-\boldsymbol{\alpha}_i)\mathbf{v}\sigma_i(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}\sum_{k\in[h]\setminus\{i\}} \boldsymbol{\alpha}_k \sigma_k(\mathbf{W}^{*\top}\mathbf{x})\right)^2\right] \\
&\overset{(2)}{=} \frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}\left[\left((1-\boldsymbol{\alpha}_i)\mathbf{v}\sigma_i(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}\sum_{k\in[h]\setminus\{i\}} (1-\boldsymbol{\alpha}_i)\gamma_k \sigma_k(\mathbf{W}^{*\top}\mathbf{x})\right)^2\right] \\
&= (1-\boldsymbol{\alpha}_i)^2\frac{1}{2}\mathbb{E}_{\mathcal{D}_\mathcal{X}}\left[\left(\mathbf{v}\sigma_i(\mathbf{W}^{*\top}\mathbf{x}) - \mathbf{v}\sum_{k\in[h]\setminus\{i\}} \gamma_k \sigma_k(\mathbf{W}^{*\top}\mathbf{x})\right)^2\right] \\
&= (1-\boldsymbol{\alpha}_i)^2\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{i\}} \gamma_k \sigma_k}(\mathbf{W}^*))
\end{aligned}$$

where (1) holds by Assumption 3 and (2) holds by Assumption 2 and Lemma 5. Applying this to $\mathcal{L}(f_{\sigma_j}(\mathbf{W}^*))$ as well, we have

$$\begin{gathered}
\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*)) < \mathcal{L}(f_{\sigma_j}(\mathbf{W}^*)) \\
\Rightarrow (1-\boldsymbol{\alpha}_i)^2\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{i\}} \gamma_k \sigma_k}(\mathbf{W}^*)) < (1-\boldsymbol{\alpha}_j)^2\mathcal{L}(f_{\sigma_j}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{j\}} \gamma_k \sigma_k}(\mathbf{W}^*)).
\end{gathered}$$

Then Condition 6 implies the following

$$\begin{aligned}
\Rightarrow (1-\boldsymbol{\alpha}_i)^2 &< (1-\boldsymbol{\alpha}_j)^2 \frac{\mathcal{L}(f_{\sigma_j}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{j\}} \gamma_k \sigma_k}(\mathbf{W}^*))}{\mathcal{L}(f_{\sigma_i}(\mathbf{W}^*), f_{\sum_{k\in[h]\setminus\{i\}} \gamma_k \sigma_k}(\mathbf{W}^*))} \\
&\leq (1-\boldsymbol{\alpha}_j)^2 \\
\Rightarrow \boldsymbol{\alpha}_j &< \boldsymbol{\alpha}_i.
\end{aligned}$$

Thus Condition 6 is sufficient for $\boldsymbol{\alpha}_j < \boldsymbol{\alpha}_i$. □

5 Conclusions and future work

We have shown that in a particular setting of NAS with weight-sharing, activation function search spaces of arbitrary size in two-layer networks, one can derive sufficient conditions under which the optimal discrete architecture according to the shared weights is the same as the architecture obtained by applying $\arg\max$ discretization to the architecture weights. In search spaces of exactly two activation functions, no sufficient condition is necessary.

There are several clear extensions to this work. One of which arises by noting that our analysis does not depend on properties of the activation functions themselves—we can easily generalize our analysis to search spaces over a much larger class of functions. Indeed, if we continue to assume that we obtain optimal shared weights, we can generalize the analysis to include parameterized functions including various types of convolutions, which are typical of NAS search spaces. Another such extension is to consider more complex architectures such as the computation graphs featured in Figure 1 or DARTS cells Liu et al. (2019). However, issues may arise when dealing with nonlinearities, so extensions to computation graphs may have to be limited to deep linear networks. On the other hand, all operations in standard NAS search spaces are 1-positively homogeneous, including ReLU, max pooling, and all types of convolutions as they are linear (Roberts et al. (2021) noted that other NAS operations including average pooling, identity, and the zero operations are all essentially special cases of convolutions), so this property might be useful as well. Extending this analysis to computation graphs will require proving that in both layer composition, and in layer addition, the architecture parameters factor together to some point in a higher-dimensional simplex. Using these two arguments, we must show that arbitrary computation graphs can be factored such that the architecture parameters at each edge factor to a vector on a higher dimensional simplex where each vertex corresponds to an architecture topology in the search space. Finally, we must show that choosing the maximum point on this simplex over the entire search space is identical to locally choosing the $\arg\max$ at each edge. We leave this analysis to future work.

REFERENCES

Mikhail Khodak, Liam Li, Nicholas Roberts, Maria-Florina Balcan, and Ameet Talwalkar. A simple setting for understanding neural architecture search with weight-sharing. In *7th ICML Workshop on Automated Machine Learning*, 2020. URL https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_46.pdf.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In Ryan P. Adams and Vibhav Gogate (eds.), *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pp. 367–377. PMLR, 22–25 Jul 2020. URL https://proceedings.mlr.press/v115/li20c.html.

Liam Li, Mikhail Khodak, Nina Balcan, and Ameet Talwalkar. Geometry-aware gradient algorithms for neural architecture search. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=MuSYkd1hxRP.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eYHoC5FX.

Samet Oymak, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural architecture search with train-validation split. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8291–8301. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/oymak21a.html.

Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4095–4104. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/pham18a.html.

Nicholas Roberts, Mikhail Khodak, Tri Dao, Liam Li, Christopher Re, and Ameet Talwalkar. Rethinking neural operations for diverse tasks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=je4ymjfb5LC.

Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1loF2NFwr.