Towards a Continuous Hyperparameter Representation for Neural Networks

William Guss* Other Author† Other Author‡

September 12, 2017

Abstract

*Email: wguss@berkeley.edu
†Email: other@berkeley.edu
‡Email: other@berkeley.edu

Contents

1 Planning & Unorganized Results (*)

1.1 General Todos and deadlines

1. **TODO: Formalize capacity loss.** This could be expexcted training loss or expected test accuracy, etc. The goal with this step is to find some sort of notion of loss which accounts for how when adding capacity, (randomly initialized), the gains to performance will almost never be immediate but in how quickly learning converges and how accurate the model is on the test set. In providing the formalizaiton, we aim to either show that *no local*[1]*, gradient based NAS exists* or propose a new gradient based NAS which descends this new expected capacity loss.

2. **TODO: Select a conference for submission.**

3. **TODO: Finish motivation section.**

4. **TODO: Write problem statement**

5. **TODO: Formalize (and write about) the continuous parameterization of NNs.**

   - **TODO: Formalize continuous parameterization of hidden dimension.**
   - **TODO: Formalize continuous parameterization of certain hyperparameters in convolutional layers.** This is especially important if it is not possibleto parameterize a gain in hidden dimension. We might be able to pull off just making certain convolution hyperparameters continuous (e.g. stride).
   - **TODO: Formalize continuous parameterization of computation graph topology.**

6. **TODO: Populate reading list section.**

7. **TODO: Propose several experiments.** Furthermore what follows are just a few ideas.

   - **TODO: Create an experiment which compares DNAS to continuous paramaeterization NAS.**
   - **TODO: Create an experiment which uses DDPG or A-NAF to perform reinforcement learning on the continuous representation.** We want to show that despite our heradling of "local" search methods, the entire architecture search community can use this unified representation to their benefit.

     An interesting sub-experiment here would be something to do with comparison to how RL performs on a descrete representation of hyperaparameters using DDPG

[1] Local in this case means that the search method is with respect to ascending the immediate training accuracy, not local in the sense that the search is local in hyperparameter space, ie. $d(\theta_t, \theta_{t+1}) < \epsilon$ where θ are model hyper parameters and d is some distance measure.

with a non-differentaible transition function[2] If continuous outperforms the standard results, it becomes adventageous outside of DNAS literature like DNAS and Neural Fabrics.

8. **TODO: Create a dataset of datasets from UCI and OpenAI Gym extension.**

1.2 Motivation & Goal

In recent years, the success of deep learning has revolutionized machine learning in application to computer vision, reinforcement learning, and natural language processing[3]. In addition to their optimal performance, neural networks require little to no feautre engineering on the part of the practitioner. However in most recent approaches, considerable would-be feature engineering effort is placed on neural network design; that is, the price of automatically learning features in neural networks is that the pracitioner must then specify with some inductive bias what parameterixation and network capacity is optimal for learning those features. For example, the choice of convolutional or fully-connected layers in computer vision, or depth, width, and other topological properties of neural networks, all define the set of features which can be–or probabilistically will be–learned over the course of training. To paraphrase Andrew Ng, "coming up with features is difficult, time-consuming, and requries expert knowledge;" and by analogy neural network design has become the new feature engineering of machine learning.

In an attempt to eliminate the dpeendence of deep learning on architecture engineering, two major approaches have been set forth: the first, neural architecture search, specifies architectures as existing in a unified hyperparameter space and then performs various search methods therein; the second, hyperparameter-free deep learning , parameterizes the space of architectures so that the aforementioned hyperparameters are themeselves learned using the same optimizer as the weights and parameters they specify. In essence neural architecture search is a learned–and sometimes brute force[4]–approximation of the practitioner, which shares striking similarities to the traditional feature selection methods of statistical machine learning[5]. On the other hand, current hyperparameter-free approaches .... in direct opposition with the idea of extensions and random initializations.

TODO: Discuss various solutions in aggregate i.e. neural architecture search, differentiable methods.

TODO: Suggest new method whic hdoes not rely on

[2] This basically forces the continuous actions of the actor to be in a softmax distribution and then selects the argmax as the discrete action.

[3] **TODO: cite**

[4] **TODO: Cite random search of architecture paper if this does exist**

[5] **TODO: Cite old ML automatic feature search methods**

1.3 Questions/Hypothesis

- How to do it?
- Is the optimization problem tractable (e.g. will it lead to getting stuck in local optima which are far from the global optimum)?
- Use DFMs to represent widtha
- What about depth? It might not matter
- Graphons = DFMs (how many times you apply surface to itself determines how deep in the network you go)
- Takes approaches in fabrics and DAS and fully realizes a continuous optimization, how does performance compare in the continuous version?
- How do you scale continuous hyperparameter representation search? What search methods are optimal in this space (gradient)? https://arxiv.org/abs/1609.01596
- Is there any substantial benefit from separating the optimization of hyperparameters and parameters using direct feedback alignment?
- Gradient-based Hyperparameter Optimization through Reversible Learning

1.4 Theory

1.4.1 Desired Results

1.4.2 Some Exposition

1.5 Experiments

The following are a set of desired experiments to verify the newly proposed hyperparameter representation.

1.6 Reading List

1.7 Related Notes

- Continuous Hidden Dimension
- **Some Thoughts on Local Search on Hidden Units.** Let $\mathcal{N}$ be the $\mathfrak{n}$-discrete instatntiation of the following DFM

$$\mathcal{O} : \boxed{\mathbb{R}^n} \xrightarrow{\mathfrak{d}} \boxed{L^1(E(\gamma))} \xrightarrow{\mathfrak{f}} \boxed{\mathbb{R}}$$

where $E : \mathbb{R} \to \mathcal{L}(\mathbb{R})$ is a function which parameterizes the domain over which the $\mathfrak{f}$-functional integrates.

It was concluded in the last note that if $E(\gamma) = [0, \gamma] \in \mathcal{L}(\mathbb{R})$ then we have the following problem for the piecewise constant parameterization of weights on $\mathfrak{f}, \mathfrak{d}$. Let $F : \mathbb{R} \to \mathbb{R}$ be some loss function, and then computation of the local gradient ascent path gives

$$\begin{aligned}
\frac{\partial F}{\partial \gamma} &= \frac{dF}{dy^2}\frac{\partial y^2}{\partial \gamma} \\
&= \frac{dF}{dy^2} \cdot \left[\frac{\partial}{\partial \gamma} \int_{[0,\gamma]} \sum_{k=1}^{\infty} [\sigma \circ \mathfrak{d}(x)](u) \chi_{k\cdot[0,1]}(u) W_k^1 \; d\mu(u) \right]_{\mathfrak{n}} \\
&= \frac{dF}{dy^2} \cdot \left[\sum_{k=1}^{\infty} [\sigma \circ \mathfrak{d}(x)](\gamma) \chi_{k\cdot[0,1]}(\gamma) W_k^1 \right]_{\mathfrak{n}} \\
&= \frac{dF}{dy^2} \cdot y^1_{\lfloor\gamma\rfloor} W^1_{\lfloor\gamma\rfloor}.
\end{aligned}$$

In otherwords, gradient ascent on F with respect to γ will increase γ if the error will decrease when the contribution of the last output neuron is increased (in magnitude); that is, if $\gamma' > \gamma$ then $(\gamma - \lfloor\gamma\rfloor)$ increases, and thus E decreases by virtue of the term

$$\int_{\lfloor\gamma\rfloor\cdot[0,1]} y^1(u) W^1_{\lfloor\gamma\rfloor} \; d\mu(u) = (\gamma - \lfloor\gamma\rfloor) y^1_{\lfloor\gamma\rfloor} W^1_{\lfloor\gamma\rfloor}$$

increasing. Searching over γ is effecitvely the same as spending extra time changing the weight $W^1_{\lfloor\gamma\rfloor}$ using two linearly dependent parameters, $(\gamma - \lfloor\gamma\rfloor)$ and $W^1_{\lfloor\gamma\rfloor}$, itself[6].

Thus we are led to the question: *Is hyperparameter search a matter of immediate model accuracy or expected capacity for accuracy, and in that distinction, does optimizing hyperparameters with respect to model accuracy coorespond to optimization on model capactiy and vice versa?* Let us examine this question in the following context.

Above, we noted that a local search on γ decreased error in exactly the same fashion as standard gradient descent, but a step in γ of more than integral amount can increase error. To see this let $k = \lfloor\gamma\rfloor$. When $\Delta\gamma > 1$ then the $(k+1)$th neuron is then "enabled" so-to-speak. However, this $(k+1)$th neuron may perform a computation that increases error and so in the next step of gradient descent $\Delta\gamma$ would be negative, retreating away from the added model capacity of a randomly intiialized $(k+1)$th neuron. That is not to say that γ might not increase again, repeating the process, or in the limit of such oscilations the update $W^1_{k+1} - \alpha\partial E/\partial W^1_{k+1} \to W^1_{k+1}$, will eventually contribute to model accuracy, but relying on these dynamics as a result with no guarentees of convergence is questionable. Despite the fact that $\mathcal{N}$ may need additional model capacity[7], local

[6] An additonal conclusion is, at least by analogy, that local search on $E(\gamma)$ at any one place assumes that adjacent neurons have similar values

[7] There are functions which are unlearnable without a sufficient number of neurons for example.

search on capacity with respect to accuracy may not yield the required capacity to increase accuracy in the limit.

Baring that local search doesn't necessarily yield the desired properties, we might now consider a global search. The more general setting is of course considering E as a function of variable support geometry. In particular let $E : C_*^\infty(\mathbb{R}) \to \mathcal{L}(\mathbb{R})$ so that $f \mapsto \text{supp}(|f|/2 + f/2)$. Not that $C_*^\infty(\mathbb{R})$ is the set of infinitely differentiable functions which vanish on at most a μ-null set. Then computation of a gradient descent step becomes a variational problem

$$\begin{aligned}
\frac{\delta F}{\delta f} &= \frac{\partial F}{\partial y^2}\frac{\delta y^2}{\delta f} \\
&= \frac{\partial F}{\partial y^2}\left[\frac{\delta}{\delta f}\int_{E(f)} y^1\omega_1 \, d\mu\right]_{\mathfrak{n}} \\
&= \frac{\partial F}{\partial y^2}\left[\frac{\delta}{\delta f}\lim_{\kappa\to\infty}\int_{\mathbb{R}} \sigma(\kappa f)y^1\omega_1 \, d\mu\right]_{\mathfrak{n}}
\end{aligned}$$

Now we attempt to compute the functional derivative of integration; that is, let $J_\kappa[f] = \int_{\mathbb{R}} \sigma(\kappa f)y^1\omega_1 \, d\mu$. Then

$$\begin{aligned}
\frac{\delta J}{\delta f} &= \lim_{\epsilon\to 0}\lim_{\kappa\to\infty}\frac{J_\kappa[f+\epsilon\phi] - J_\kappa[f]}{\epsilon} \\
&= \lim_{\kappa\to\infty}\left[\frac{d}{d\epsilon}J_\kappa[f+\epsilon\phi]\right]_{\epsilon=0} \\
&= \lim_{\kappa\to\infty}\left[\int_{\mathbb{R}}\frac{d}{d\epsilon}\sigma(\kappa(f+\epsilon\phi))y^1\omega_1 \, d\mu\right]_{\epsilon=0} \\
&= \lim_{\kappa\to\infty}\left[\int_{\mathbb{R}}\sigma'(\kappa(f+\epsilon\phi))\kappa\phi y^1\omega_1 \, d\mu\right]_{\epsilon=0} \\
&= \int_{\mathbb{R}}\lim_{\kappa\to\infty}\sigma'(\kappa f)\kappa\phi y^1\omega_1 \, d\mu. \\
&= \int_{\mathbb{R}}\delta(f(u))\phi(u)y^1(u)\omega_1(u) \, d\mu(u). = \sum_{z\in Z(f)}\phi(z)y^1(z)\omega_1(z)
\end{aligned}$$

where $Z(f)$ is the set of zeroes of f. Thus we yield a functional gradient ascent step via the linear approximation

$$\begin{aligned}
J[\phi] &= J[f] + \frac{\delta J}{\delta f}[\phi - f] + \frac{1}{2t}\|\phi - f\|^2 \\
0 &= 0 + \frac{\delta J}{\delta f} + \frac{\phi - f}{t} \\
\phi &= f - t\left(\psi \mapsto \sum_{z\in Z(f)}\psi(z)y^1(z)\omega_1(z)\right).
\end{aligned}$$

In addition to the previous update rule, we come to the redundant conclusion that $\delta J/\delta f|_{f=\Gamma} = 0$ for Γ with no zeros, and therefore when $\Gamma < 0$ we have only found a minimum for J. In any case, the gradient in the hard limit of k is zero at every point at which $f(u)$ is non-zero. Using a soft limit $\kappa \not\to \infty$ we get an ascent direction in the continuum of our initial search on γl that is

$$\phi = f - \sigma'(\kappa f)\kappa\phi \sum_{k=1}^{\infty} \chi_{k\cdot[0,1]} \left[\sigma \circ \delta(x)\right] W_k^1.$$

However, we face the same question as to whether optimziation on the loss function would yield ascent in the direction of random vectors for the purpose of capacity and not by their similarity actions to values which decrease error.

- **Fuzzy Vector Spaces as a Solution to Continuous Parameterization of Hidden Dimension.**[8] Before we digress into a discussion of meta-loss functions[9], we will consider an alternative approach to continuous parameterization.

  At its core, our task is to at least answer the following question: how is it possible to exhibit a real number of neurons on any given layer? As we saw in the previous notes, DFMs yield an answer that although locally satisfactory is inhibited globally. Instead, we might equate the aforementioned question with that of linear algebra: are there spaces with real algebraic[10] dimension? In a category theoretic sense, this question has an answer, but all examples (see Fibonacci categories) of fractional algebraic dimension lose the linear structure and any computable parameterization. Despite this, a last resort is to take literally the dimension of a vector space as the cardinality of its basis, and again our question is reduced to the following: do there exist linearly independent sets in $\mathbb{R}^{\mathbb{N}}$ with real cardinality. The space they 'span' in such a regime has the desired property for our continuous parameterization.

  A plausible answer to these questions is fuzzy vector spaces. In these spaces, a basis is a fuzzy set of vectors which both span their space and are linearly independent to eachother. As fuzzy sets, their cardinality, and hence the dimensionality of the spece, is not an integer, but a fuzzy integer; that is to say, their cardinality is a postivie real number. Without digressing into a full discussion of fuzzy vector spaces, it suffices to think of this approach as similar to that of dropout. Each neuron (and potential neuron) has a grade of membership, and there is a constraint on the sum of these grades. As learning occurs the constraint changes and sume neurons are given larger

[8] This note is speculatory, and if time permits, a prerequisites seciton will be added to this document wherein a full discussion of fuzzy-vector spaces, and proofs of the optimizations there involved will be presented.

[9] *Meta-loss functions* means those which are conducive to increasing capacity in spite of immediate test error.

[10] The reader who has not studied measure theory need not concern themselves with this qualifier. However, when we discuss real dimension, we mean not that the space has some fractal, real Hausdorff dimension, but that the space potentially has a module structure and a Fusion rule from which trace defines dimension.

and larger gradings untill they are 'members'. The grading of a neuron is a value in $[0, 1]$ and as such can be thought of as a probability, in essence the 'number' of hidden units is an expectation on the activity of those hidden units; in a sense we gradient ascend on which neurons will be active in expectation and not in the magnitudes of their contributions.

Do these expectations some how code for capacity, or just merely introduce noise into the process? **TODO: Finish the note: add all definitions for fuzzy spaces, provide gradient ascent analysis.**

- **Model Capacity and Expectation.**