7th ICML Workshop on Automated Machine Learning (2020)

A Simple Setting for Understanding Neural Architecture Search with Weight-Sharing

Mikhail Khodak KHODAK@CMU.EDU
Carnegie Mellon University

Liam Li ME@LIAMCLI.COM
Determined AI

Nicholas Roberts NCROBERT@CS.CMU.EDU
Carnegie Mellon University

Maria-Florina Balcan NINAMF@CS.CMU.EDU
Carnegie Mellon University

Ameet Talwalkar TALWALKAR@CMU.EDU
Carnegie Mellon University and Determined AI

Abstract

Neural architecture search (NAS) has emerged as a promising direction for research in automated machine learning by automating deep net design. The goal of this paper is to spur progress on its understudied learning-theoretic and algorithmic questions, with the purpose of developing new algorithms and informing existing approaches. Specifically, we consider the popular, state-of-the-art technique of weight-sharing—the simultaneous optimization of multiple networks using the same parameters—and propose feature map selection as a tractable problem for its analysis. As with NAS, we show in this setting that weight-sharing provides a useful signal to evaluate many configurations without individual training. Finally, we show how feature map selection yields a sample complexity justification for using bilevel rather than single-level optimization in NAS, a question raised in several recent works.

1. Introduction

An important tool for automating machine learning, neural architecture search has seen great progress in recent years (Real et al., 2019; Cai et al., 2019); in particular, *weight-sharing* (Pham et al., 2018) has led to fast algorithms with state-of-the-art results on canonical image classification and language modeling problems (Liu et al., 2019; Li and Talwalkar, 2019). At the same time, theoretical study of statistical and optimization questions in this area has been minimal. While NAS is at least as hard a problem as deep learning, we might still hope to gain insight from understanding certain sub-problems, sub-routines, or simple settings.

We take the latter approach and propose *feature map selection*—picking the best fixed data representations to use in a linear model—as a way to study NAS and weight-sharing. In this setting, we show empirically that weight-sharing provides a signal in the form of correlation between the performances of shared and standalone weights for individual configurations, just as in NAS. Using this, we design a simple method that outperforms hyperparameter tuning baselines on two tasks. Finally, we show how the simple setting also provides an explicit case where a bilevel objective improves over single-level optimization in terms of sample complexity and discuss insights and limitations of this analysis.

1.1 Background

In NAS we are interested in searching over a finite set of configurations $\mathcal{C}$, where each element $c \in \mathcal{C}$ is associated with a hypothesis class $H_c = \{h_{w,c} : w \in \mathcal{W}\}$ of functions $h_{w,c} : \mathcal{X} \mapsto \mathcal{Y}'$ parameterized by $\mathcal{W} \subset \mathbb{R}^d$. The joint hypothesis space over configurations and parameters is then $H(\mathcal{W}, \mathcal{C}) = \bigcup_{c \in \mathcal{C}} H_c = \{h_{w,c} : w \in \mathcal{W}, c \in \mathcal{C}\}$. As usual the goal of learning is to find $h_{w,c} \in H(\mathcal{W}, \mathcal{C})$ with low population risk $\ell_{\mathcal{D}}(w, c) = \ell_{\mathcal{D}}(h_{w,c}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\ell(h_{w,c}(x), y)$ for loss $\ell : \mathcal{Y}' \times \mathcal{Y} \mapsto \mathbb{R}$ and some distribution $\mathcal{D}$ over sample space $\mathcal{X} \times \mathcal{Y}$.

Given a dataset of examples from $\mathcal{D}$, a standard approach is to do regularized empirical risk minimization (ERM) over $H(\mathcal{W}, \mathcal{C})$. However, most NAS algorithms split the data into training and validation sets $T, V \subset \mathcal{X} \times \mathcal{Y}$ and instead solve the following bilevel problem:

$$\min_{w \in \mathcal{W}, c \in \mathcal{C}} \quad \ell_V(w, c) \quad \text{s.t.} \quad w \in \arg\min_{u \in \mathcal{W}} \quad \mathcal{L}_T(u, c), \tag{1}$$

where ℓ_V is the empirical risk over V and $\mathcal{L}_T$ is a (possibly) regularized objective over T.

A starting point for solving the bilevel optimization problem is random search (Bergstra and Bengio, 2012): randomly sample a configuration $c \in \mathcal{C}$, train the inner loop problem to completion, and repeat. More sophisticated methods adapt the distribution (Hutter et al., 2011) or allocate fewer resources to less-promising configurations (Li et al., 2018). A major drawback of these methods is the need to train configurations separately, which means only a limited part of the search space can be explored under resource constraints. Hence, Pham et al. (2018) proposed simultaneously exploring both the weight-space $\mathcal{W}$ and the configuration space $\mathcal{C}$ by partially training the weights w with minibatch gradient steps w.r.t. architectures c sampled from a parameterized distribution. These weights are called *shared* because they are trained to optimize a distribution of architectures over $\mathcal{C}$.

Despite the noise due to updating w.r.t. different architectures, weight-sharing has become an incredibly successful tool. A simple example that illustrates its surprising power is random search with weight-sharing (RS-WS) (Bender et al., 2018; Li and Talwalkar, 2019), where weights are trained by taking minibatch gradient steps w.r.t. *uniformly* sampled architectures. The bilevel object solved by RS-WS is thus

$$\min_{c \in \mathcal{C}} \ \ell_V(w^*, c) \quad \text{s.t.} \quad w^* \in \arg\min_{u \in \mathcal{W}} \ \mathop{\mathbb{E}}_{c \sim \text{Unif}(\mathcal{C})} \mathcal{L}_T(u, c)$$

i.e. shared-weights are trained to optimize the expected loss of a uniformly sampled architecture. This is followed by evaluations of different configurations $c \in \mathcal{C}$ using the resulting shared weights w^*. In the case of the two benchmarks studied by Li and Talwalkar (2019), highly performant architectures according to the shared-weights also had high ground truth performance. In Section 2 we observe analogous behavior for the feature map selection problem.

1.2 Limitations and Questions

Shared-weights seem to be able to encode a remarkable amount of signal in over-parameterized NAS models. However, many questions statistical and algorithmic questions about weight-sharing remain. One major question, that we study using our proposed simple setting, is why many practitioners find it useful to use bilevel optimization, which is a model selection technique, when we are effectively training a *single* supernet that subsumes the hyperparameters as regular model parameters (Li et al., 2020). Some recent work has

Algorithm 1: Feature map selection using successive halving with weight-sharing.

Input: training set T, validation set V, convex loss ℓ, set of feature map ϕ_c configurations $\mathcal{C}$, regularization parameter $\lambda > 0$

for *round* $t = 1, \ldots, \log_2 |\mathcal{C}|$ **do**

  for *datapoint* $(x, y) \in T \cup V$ **do**

    assign $c_x \sim \text{Unif}(\mathcal{C})$ i.i.d.

  $w_t^* \leftarrow$

  $$\arg\min_{w \in \mathbb{R}^d} \ \lambda \|w\|_2^2 + \sum_{(x,y) \in T} \ell(\langle w, \phi_{c_x}(x) \rangle, y)$$

  for *configuration* $c \in \mathcal{C}$ **do**

    $V_c \leftarrow \{(x, y) \in V : c_x = c\}$

    $s_c \leftarrow \frac{1}{|V_c|} \sum_{(x,y) \in V_c} \ell(\langle w_t^*, \phi_{c_x}(x) \rangle, y)$

  $\mathcal{C} = \{c \in \mathcal{C} : s_c \leq \text{Median}(\{s_c : c \in \mathcal{C}\})\}$

Result: Singleton set of configurations $\mathcal{C}$.

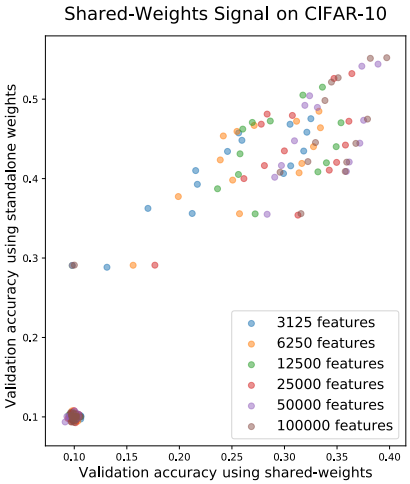

Figure 1: Validation accuracy of individual feature maps using shared weights compared to individual training.

questioned this trend empirically (Xie et al., 2019; Li et al., 2019, 2020), and the single-level case is perhaps more tractable to analyze (Li et al., 2020). In Section 3 we use our study of feature map selection, a hyperparameter configuration problem that can be viewed as the simplest form of NAS, to provide concrete excess risk bounds as part of a broader theory of why bilevel optimization has been found to lead to better generalization in practice.

2. Feature Map Selection: A Simple Setting for Understanding NAS

In this section, we introduce feature map selection and show how it can be viewed as a simple type of NAS with weight-sharing. Empirically, weight-sharing outperforms random search and Hyperband (Li et al., 2018) both when selecting random Fourier maps on CIFAR-10 and configuring Bag-of-n-Grams representations on IMDb, motivating further analysis.

In feature map selection each configuration $c \in \mathcal{C}$ corresponds to a feature map $\phi_c : \mathcal{X} \mapsto \mathbb{R}^d$ of the input to be passed to a linear classifier in $\mathcal{W} = \mathbb{R}^d$; the hypothesis space is then $H(\mathbb{R}^d, \mathcal{C}) = \{\langle w, \phi_c(\cdot) \rangle : w \in \mathbb{R}^d, c \in \mathcal{C}\}$. This can be viewed as a simple NAS problem, with the difference that in neural nets the maps ϕ_c also depend on parameter-weights w, whereas here we only parameterize the last layer. We can write the standard bilevel optimization for feature map selection in the form of (1) for regularization parameter $\lambda > 0$:

$$\min_{c \in \mathcal{C}} \ \sum_{(x,y) \in V} \ell(\langle w_c^*, \phi_c(x) \rangle, y) \quad \text{s.t.} \quad w_c^* = \arg\min_{w \in \mathbb{R}^d} \ \lambda \|w\|_2^2 + \sum_{(x,y) \in T} \ell(\langle w, \phi_c(x) \rangle, y) \quad (2)$$

Note that as a non-architectural hyperparameter, λ is not tuned by weight-sharing.

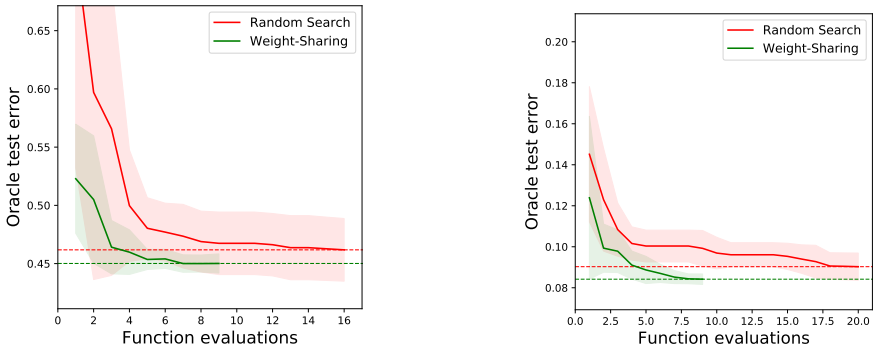

Figure 2: Oracle test-error on CIFAR-10 (left) and IMDb (right) as a function of number of solver calls. Here, *oracle* test-error refers to evaluation of a separately trained, non-weight-shared, classifier on the best config at any given round according to weight-sharing. All curves are averaged over 10 independent trials.

How can we use weight-sharing to approximate (2) without solving for w_c^* for each configuration $c \in \mathcal{C}$? We take inspiration from the RS-WS algorithm described in Section 1, which allocates a resource—a minibatch of training examples—to a configuration at each iteration. Analogously, in Algorithm 1, we propose to allocate training examples to feature maps: at each iteration t we solve an ERM problem with each point featurized by a random map $\phi_c, c \sim \text{Unif}(\mathcal{C})$. The result w_t is thus *shared* among the feature maps rather than being the minimizer for any single one; as with RS-WS we find that, despite being trained using the uniform distribution over configurations, as showin in Figure 1 the shared-weights w_t are much better classifiers for data featurized using the best maps. We use this as validation signal in a successive halving procedure approximating (2) in $\log_2 |\mathcal{C}|$ regression solves.

We evaluate Algorithm 1 on two problems: kernel ridge regression over random Fourier features (Rahimi and Recht, 2008) on CIFAR-10 and regularized SVM classification over hashed Bag-of-n-Gram representations (Wang and Manning, 2012) on IMDB.[1] The hyperparameters we tune are described in the Appendix. In Figure 1, we show that the performance of the shared weights found at the first stage of Algorithm 1 on CIFAR-10 is strongly correlated with the standalone validation accuracy; thus, as in NAS we can use weight-sharing to obtain a useful noisy indication of the performance of many configurations while training only one set of parameters. We can exploit this fact in Figure 2, where we see that Algorithm 1 obtains a better test accuracy in fewer calls to the ridge regression or SVM solver than random search. Interestingly, in terms of wallclock time the average weight-sharing solve is slower than the average single-configuration solve, but in the Appendix we describe how we can modify Algorithm 1 to outperform both random search and Hyperband (Li et al., 2018) in terms of both accuracy and time. Note that we do not compare to Hyperband in the above plot because it computes many more solutions to lower-dimensional problems and so the evaluations are incomparable.

[1] We make code available at `https://github.com/mkhodak/weight-sharing`

3. Generalization Guarantees for the Bilevel Problem

While in our experiments we have used the bilevel formulation, it is not immediately clear, in NAS or feature map selection, that the joint ERM problem $\min_{c\in\mathcal{C},w\in\mathcal{W}} \quad \lambda\|w\|_2^2 \quad + \sum_{(x,y)\in T\cup V} \ell(\langle w, \phi_c(x)\rangle, y)$ over the combined data would not also work. This question is interesting due to the widespread use of the bilevel formulation in NAS with weight-sharing. Especially when continuous relaxation Liu et al. (2019) is applied, architecture parameters in NAS appear more similar to regular model parameters rather than controls on the model complexity (Li et al., 2020), so it is reasonable to wonder why most NAS practitioners have used the bilevel formulation. In this section we give an analysis suggesting that decomposing the objective can improve generalization by adapting to the sample complexity of the best configuration in $\mathcal{C}$, whereas ERM suffers that of the worst. For feature map selection our theory gives concrete excess risk bounds.

We start with a key fact about the weights that optimize the bilevel problem: they are optima $\arg\min_{w\in\mathcal{W}} \mathcal{L}_T(w, c)$ of the inner objective, i.e. elements of the *version space* $H_{c,T} = \{h_{w,c} : w \in \arg\min_{u\in\mathcal{W}} \mathcal{L}_T(u, c)\}$ (Kearns et al., 1997, Equation 6). We use the following data-dependent quantification of how much the hypotheses are restricted by the inner optimization for $N(F, \varepsilon)$ the L^∞*-covering-number* of a set of functions F at scale $\varepsilon > 0$, i.e. the number of L^∞ balls in an ε-cover of F (Mohri et al., 2012, Equation 3.60):

Definition 3.1 *The* **version entropy** *of $H(\mathcal{C}, \mathcal{W})$ at scale $\varepsilon > 0$ induced by the objective $\mathcal{L}_T$ over training data T is $\Lambda(H, \varepsilon, T) = \log N\left(\bigcup_{c\in\mathcal{C}} H_{c,T}, \varepsilon\right)$.*

For finite $\mathcal{C}$, the version entropy is less than $\log|\mathcal{C}| + \max_{c\in\mathcal{C}} \log N(H_{c,T}, \varepsilon)$, so that the second term measures the worst-case complexity of the global minimizers of $\mathcal{L}_T$. In the feature selection problem, $\mathcal{L}_T$ is usually a strongly-convex loss due to regularization and so all version spaces are singleton sets, making the version entropy $\log|\mathcal{C}|$. In the other extreme case of nested model selection the version entropy reduces to the complexity of the version space of the largest model and so may not be informative. However, in practical problems such as NAS an inductive bias is often imposed via constraints on the number of input edges.

To bound the excess risk in terms of the version entropy, we first discuss an important assumption describing cases when we expect the bilevel approach to perform well:

Assumption 3.1 *There exists a $c^* \in \mathcal{C}$ s.t. $(w^*, c^*) \in \arg\min_{\mathcal{W}\times\mathcal{C}} \ell_{\mathcal{D}}(w, c)$ for some $w^* \in \mathcal{W}$ and s.t. w.h.p. over the training sample T at least one of the minima of the optimization induced by c^* and T has low excess risk, i.e. w.p. $1-\delta$ there exists $w \in \arg\min_{u\in\mathcal{W}} \mathcal{L}_T(u, c^*)$ s.t. $\ell_{\mathcal{D}}(h_{w,c^*}) - \ell_{\mathcal{D}}(h^*) \le \varepsilon^*(|T|, \delta)$ for excess risk ε^* and all $h^* \in H(\mathcal{W}, \mathcal{C})$.*

This assumption requires that the inner optimization objective does not exclude all good classifiers for the optimal configuration. It asks nothing of the other configurations in $\mathcal{C}$, which may be arbitrarily bad, nor of the hypotheses found by the procedure, but prevents the case where even minimizing the objective $\mathcal{L}_T(\cdot, c^*)$ does not provide a set of good weights. Note that if the inner optimization is simply ERM over the training set T, i.e. $\mathcal{L}_T = \ell_T$, then standard learning-theoretic guarantees will give $\varepsilon^*(|T|, \delta)$ decreasing in the size $|T|$ of the training set and increasing at most poly-logarithmically in $\frac{1}{\delta}$. With this assumption, we can show the following guarantee for solutions to the bilevel optimization (1):

Theorem 3.1 *If Assumption 3.1 holds, ℓ is B-bounded, and $(w, c) \in \mathcal{W} \times \mathcal{C}$ solves the bilevel objective* (1) *for space $H(\mathcal{W}, \mathcal{C})$ and data V, T, as in Section 1 then w.p. $1 - 3\delta$*

$$\ell_{\mathcal{D}}(h_{w,c}) \leq \min_{h \in H(\mathcal{W},\mathcal{C})} \ell_{\mathcal{D}}(h) + \varepsilon^*(|T|, \delta) + \inf_{\varepsilon > 0} 3\varepsilon + \frac{3B}{2}\sqrt{\frac{2}{|V|}\left(\Lambda(H, \varepsilon, T) + \log\frac{1}{\delta}\right)}$$

3.1 Excess Risk of Feature Map Selection

To use this theorem we must bound the version entropy. In feature map selection, strong-convexity induces a unique minimum for each each ϕ_c and thus a singleton version space, so the bound is $\log |\mathcal{C}|$. Then we can show the following for Lipschitz (e.g. hinge) losses:

Corollary 3.1 *For feature map selection with Lipschitz loss ℓ there exists $\lambda > 0$ s.t. bilevel optimization yields a hypothesis with excess risk less than $\mathcal{O}\left(\sqrt{\frac{\|w^*\|_2^2+1}{|T|}\log\frac{1}{\delta}} + \sqrt{\frac{1}{|V|}\log\frac{|\mathcal{C}|}{\delta}}\right)$.*

In the case of selection random Fourier approximations of kernels, we can show that we can compete with the optimal RKHS from among those associated with one of the configurations:

Corollary 3.2 *In feature map selection suppose each map $\phi_c, c \in \mathcal{C}$ is associated with a random Fourier feature approximation of a continuous shift-invariant kernel approximating an RKHS $\mathcal{H}_\phi$ and ℓ is the square loss. Then for sufficiently large d there exists $\lambda > 0$ s.t. w.p. $1-\delta$ solving* (2) *yields a hypothesis with excess risk w.r.t. $\mathcal{H}_\phi$ less than $\mathcal{O}\left(\frac{\log^2\frac{1}{\delta}}{\sqrt{|T|}} + \sqrt{\frac{1}{|V|}\log\frac{|\mathcal{C}|}{\delta}}\right)$.*

In both cases we get bounds almost identical to the excess risk achievable by knowing the best configuration beforehand, up to a term depending weakly on the number of configurations. This improves upon solving the regular ERM objective, where we have to contend with the possibly high complexity of the hypothesis space induced by the worst configuration.

3.2 Version Entropy and NAS

In simple settings Theorem 3.1 can guarantee excess risk almost as good as that of the (unknown) optimal configuration without assuming anything about the complexity or behavior of sub-optimal configurations. However, for NAS we do not have a bound on the version entropy, which now depends on all of $\mathcal{C}$. Whether the version space of deep networks is small compared to the number of samples is unclear, although we gather some evidence below. The question amounts to how many (functional) global optima are induced by a training set of size $|T|$. In an idealized spin-glass model, Choromanska et al. (2015, Theorem 11.1) suggest that the number of critical points is exponential *only* in the number of layers, which would yield a small version entropy. It is conceivable that the quantity may be further bounded by the complexity of solutions explored by the algorithm when optimizing $\mathcal{L}_T$ (Nagarajan and Kolter, 2017; Bartlett et al., 2017). On the other hand, Nagarajan and Kolter (2019) argue, with evidence in restricted settings, that even the most stringent implicit regularization cannot lead to a non-vacuous uniform convergence bound; if true more generally this would imply that the NAS version entropy is quite large.

4. Conclusion

Our work aims to promote feature map selection as a computationally and analytically tractable way to study NAS. Empirically, we show that weight-sharing provides a strong signal in this setting that can beat hyperparameter tuning baselines, while theoretical usefulness is demonstrated through an explanation for the prevalence of bilevel optimization in NAS. We believe this evidence suffices to spur research to improve and better understand NAS via feature map selection and to explore the connection's strengths and limitations.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

Peter Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, 2017.

Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

Anna Choromanska, Mikael Henaff, Michael Mathieu Gérard Ben Arous, and Yann LeCun. The loss surface of multilayer networks. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015.

Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four GPU hours. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the International Conference on Learning and Intelligent Optimization*, 2011.

Michael Kearns, Yishay Mansour, Andrew Y. Ng, and Dana Ron. An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27:7–50, 1997.

John Lafferty, Han Liu, and Larry Wasserman. Statistical machine learning. 2010.

Guilin Li, Xing Zhang, Zitong Wang, Zhenguo Li, and Tong Zhang. StacNAS: Towards stable and consistent differentiable neural architecture search. arXiv, 2019.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2019.

Liam Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

Liam Li, Mikhail Khodak, Maria-Florina Balcan, and Ameet Talwalkar. Geometry-aware gradient algorithms for neural architecture search. arXiv, 2020.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

Edward Loper and Steven Bird. NLTK: The natural language toolkit. arXiv, 2002.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, 2012.

Vaishnavh Nagarajan and J. Zico Kolter. Generalization in deep networks: The role of distance from initialization. arXiv, 2017.

Vaishnavh Nagarajan and J. Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2008.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, 2017.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2012.

Karthik Sridharan, Nathan Srebro, and Shai Shalev-Schwartz. Fast rates for regularized objectives. In *Advances in Neural Information Processing Systems*, 2008.

Sida Wang and Christopher D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, 2012.

Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. SNAS: Stochastic neural architecture search. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

Appendix A. Generalization Results

This section contains proofs of the generalization results in Section 3.

A.1 Settings and Main Assumption

We first describe the setting for which we prove our general result.

Setting A.1 *Let $\mathcal{C}$ be a set of possible architecture/configurations of finite size such that each $c \in \mathcal{C}$ is associated with a parameterized hypothesis class H_c=$\{h_{w,c} : \mathcal{X} \mapsto \mathcal{Y}' : w \in \mathcal{W}\}$ for input space $\mathcal{X}$, output space $\mathcal{Y}'$, and fixed set of possible weights $\mathcal{W}$. We will measure the performance of a hypothesis $h_{w,c}$ on an input $x, y \in \mathcal{X} \times \mathcal{Y}$ for some output space $\mathcal{Y}$ using a B-bounded loss function $\ell : \mathcal{Y}' \times \mathcal{Y} \mapsto [0, B]$. Note that while the examples below have unbounded loss functions, in practice they are explicitly or implicitly bounded by explicit or implicit regularization.*

We are given a training sample $T \sim \mathcal{D}^{|T|}$ and a validation sample $V \sim \mathcal{D}^{|V|}$, where $\mathcal{D}$ is some distribution over $\mathcal{X} \times \mathcal{Y}$. We will denote the the population risk by $\ell_{\mathcal{D}}(h_{w,c}) = \ell_{\mathcal{D}}(w, c) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\ell(h_{w,c}(x), y)$ and for any finite subset $S \subset \mathcal{X} \times \mathcal{Y}$ we will denote the empirical risk over S by $\ell_S(h_{w,c}) = \ell_S(w, c) = \frac{1}{|S|}\sum_{(x,y)\in S} \ell(h_{w,c}(x), y)$.

Finally, we will consider solutions of optimization problems that depend on the training data and architecture. Specifically, for any configuration $c \in \mathcal{C}$ and finite subset $S \subset \mathcal{X} \times \mathcal{Y}$ let $\mathcal{W}_c(S) \subset \mathcal{W}$ be the set of global minima of some optimization problem induced by S and c and let the associated version space (Kearns et al., 1997) be $H_c(S) = \{h_{w,c} : w \in \mathcal{W}_c(S)\}$.

We next give as examples two specific settings encompassed by Setting A.1.

Setting A.2 *For* **feature map selection**, *in Setting A.1 the configuration space $\mathcal{C}$ is a set of feature maps $\phi_c : \mathcal{X} \mapsto \mathbb{R}^d$, the set of weights $\mathcal{W} \subset \mathbb{R}^d$ consists of linear classifiers, for inputs $x \in \mathcal{X}$ the hypotheses are $h_{w,c}(x) = \langle w, \phi_c(x)\rangle$ for $w \in \mathcal{W}$, and so $\mathcal{W}_c(S)$ is the singleton set of solutions to the regularized ERM problem*

$$\arg\min_{w\in\mathcal{W}} \quad \lambda\|w\|_2^2 \quad + \sum_{(x,y)\in S} \ell(\langle w, \phi_c(x)\rangle, y)$$

for square loss $\ell : \mathcal{Y}' \times \mathcal{Y} \mapsto \mathbb{R}_+$ and some coefficient $\lambda > 0$.

Setting A.3 *For* **neural architecture search**, *in Setting A.1 the configuration space consists of all possible choices of edges on a DAG of N nodes and a choice from one of K operations at each edge, for a total number of configurations bounded by K^{N^2}. The hypothesis $h_{w,c} : \mathcal{X} \mapsto \mathcal{Y}'$ is determined by a choice of architecture $c \in \mathcal{C}$ and a set of network weights $w \in \mathcal{W}$ and the loss $\ell : \mathcal{Y}' \times \mathcal{Y} \mapsto \mathbb{R}_+$ is the cross-entropy loss. In the simplest case $\mathcal{W}_c(S)$ is the set of global minima of the ERM problem*

$$\min_{w\in\mathcal{W}} \quad \sum_{(x,y)\in S} \ell(h_{w,c}(x), y)$$

We now state the main assumption we require.

Assumption A.1 *In Setting A.1 there exists a good architecture $c^* \in \mathcal{C}$, i.e. one satisfying $(w^*, c^*) \in \arg\min_{\mathcal{W}\times\mathcal{C}} \ell_{\mathcal{D}}(w, c)$ for some weights $w^* \in \mathcal{W}$, such that w.p. $1-\delta$ over the drawing of training set $T \sim \mathcal{D}^{|T|}$ at least one of the minima of the optimization problem induced by c^* and T has low excess risk, i.e. $\exists\ w \in \mathcal{W}_{c^*}(T)$ s.t.*

$$\ell_{\mathcal{D}}(w, c^*) - \ell_{\mathcal{D}}(w^*, c^*) \leq \varepsilon^*(|T|, \delta) \tag{3}$$

for some error function ε^*.

Clearly, we prefer error functions ε^* that are decreasing in the number of training samples $|T|$ and increasing at most poly-logarithmically in $\frac{1}{\delta}$. This assumption requires that if we knew the optimal configuration *a priori*, then the provided optimization problem will find a good set of weights for it. We will show how, under reasonable assumptions, Assumption A.1 can be formally shown to hold in Settings A.2 and A.3.

A.2 Main Result

Our general result will be stated in terms of covering numbers of certain function classes.

Definition A.1 *Let H be a class of functions from $\mathcal{X}$ to $\mathcal{Y}'$. For any $\varepsilon > 0$ the associated L^∞* **covering number** *$N(H, \varepsilon)$ of H is the minimal positive integer k such that H can be covered by k balls of L^∞-radius ε.*

The following is then a standard result in statistical learning theory (see e.g. Lafferty et al. (2010, Theorem 7.82)):

Theorem A.1 *Let H be a class of functions from $\mathcal{X}$ to $\mathcal{Y}$ and let $\ell : \mathcal{Y}' \times \mathcal{Y} \mapsto [0, B]$ be an L-Lipschitz, B-bounded loss function. Then for any distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ we have*

$$\Pr_{S\sim\mathcal{D}^m}\left(\sup_{h\in H} |\ell_{\mathcal{D}}(h) - \ell_S(h)| \geq 3\varepsilon\right) \leq 2N\left(H, \varepsilon\right) \exp\left(-\frac{m\varepsilon^2}{2B^2}\right)$$

where we use the loss notation from Setting A.1.

Before stating our theorem, we define a final quantity, which measures the log covering number of the version spaces induced by the optimization procedure over a given training set.

Definition A.2 *In Setting A.1, for any sample $S \subset \mathcal{X} \times \mathcal{Y}$ define the* **version entropy** *to be $\Lambda(H, \varepsilon, S) = \log N\left(\bigcup_{c\in\mathcal{C}} H_c(S), \varepsilon\right)$.*

Theorem A.2 *In Setting A.1 let $(\hat{w}, \hat{c}) \in \mathcal{W} \times \mathcal{C}$ be obtained as a solution to the following optimization problem:*

$$\arg\min_{w\in\mathcal{W}, c\in\mathcal{C}} \quad \ell_V(w, c) \qquad \text{s.t.} \qquad w \in \mathcal{W}_c(T)$$

Then under Assumption A.1 we have w.p. $1-3\delta$ that

$$\ell_{\mathcal{D}}(\hat{w}, \hat{c}) \leq \ell_{\mathcal{D}}(w^*, c^*) + \varepsilon^*(|T|, \delta) + B\sqrt{\frac{1}{2|V|}\log\frac{1}{\delta}} + \inf_{\varepsilon>0} 3\varepsilon + B\sqrt{\frac{2}{|V|}\left(\Lambda(H, \varepsilon, T) + \log\frac{1}{\delta}\right)}$$

Proof We have for any $w \in \mathcal{W}_{c^*}(T)$ satisfying Equation 3, whose existence holds by Assumption A.1, that

$$\ell_{\mathcal{D}}(\hat{w},\hat{c}) - \ell_{\mathcal{D}}(w^*,c^*) \quad \leq \quad \underbrace{\ell_{\mathcal{D}}(\hat{w},\hat{c}) - \ell_V(\hat{w},\hat{c})}_{1} \quad + \quad \underbrace{\ell_V(\hat{w},\hat{c}) - \ell_V(w,c^*)}_{2}$$
$$+ \quad \underbrace{\ell_V(w,c^*) - \ell_{\mathcal{D}}(w,c^*)}_{3} \quad + \quad \underbrace{\ell_{\mathcal{D}}(w,c^*) - \ell_{\mathcal{D}}(w^*,c^*)}_{4}$$

each term of which can be bounded as follows:

1. Since $\hat{w} \in \mathcal{W}_{\hat{c}}(T)$ for some $\hat{c} \in \mathcal{C}$ the hypothesis space can be covered by the union of the coverings of $H_c(T)$ over $c \in \mathcal{C}$, so by Theorem A.1 we have that w.p. $1-\delta$
$$\ell_{\mathcal{D}}(\hat{w},\hat{c}) - \ell_V(\hat{w},\hat{c}) \leq \inf_{\varepsilon > 0} 3\varepsilon + B\sqrt{\frac{2}{|V|}\left(\Lambda(H,\varepsilon,T) + \log\frac{1}{\delta}\right)}$$
2. By optimality of the pair $(\hat{w},\hat{c})$ and the fact that $w \in \mathcal{W}_{c^*}(T)$ we have
$$\ell_V(\hat{w},\hat{c}) = \min_{c\in\mathcal{C}, w' \in \mathcal{W}_c(T)} \ell_V(w',\hat{c}) \leq \min_{w' \in \mathcal{W}_{c^*}(T)} \ell_V(w',c^*) \leq \ell_V(w,c^*)$$
3. Hoeffding's inequality yields $\ell_V(w,c^*) - \ell_{\mathcal{D}}(w,c^*) \leq B\sqrt{\frac{1}{2|V|}\log\frac{1}{\delta}}$ w.p. $1-\delta$
4. Assumption A.1 states that $\ell_{\mathcal{D}}(w,c^*) - \ell_{\mathcal{D}}(w^*,c^*) \leq \varepsilon^*(|T|\delta)$ w.p. $1-\delta$.

■

A.3 Applications

For the feature map selection problem, Assumption A.1 holds by standard results for ℓ_2-regularized ERM for linear classification (e.g. Sridharan et al. (2008)):

Corollary A.1 *In Setting A.2, suppose the loss function ℓ is Lipschitz. Then for regularization parameter $\lambda = \sqrt{\frac{1}{|T|}\log\frac{1}{\delta}}$ we have*

$$\ell_{\mathcal{D}}(w,c^*) - \ell_{\mathcal{D}}(w^*,c^*) \leq \mathcal{O}\left(\sqrt{\frac{\|w^*\|_2^2+1}{|T|}\log\frac{1}{\delta}}\right)$$

We can then directly apply Theorem A.2 and the fact that the version entropy is bounded by $\log|\mathcal{C}|$ because the minimizer over the training set is always unique to get the following:

Corollary A.2 *In Setting A.2 let $(\hat{w},\hat{c}) \in \mathcal{W}\times\mathcal{C}$ be obtained as a solution to the following optimization problem:*

$$\operatorname*{arg\,min}_{w\in\mathcal{W}, c\in\mathcal{C}} \quad \ell_V(w,c) \qquad s.t. \qquad w = \operatorname*{arg\,min}_{w\in\mathcal{W}} \quad \lambda\|w\|_2^2 + \sum_{(x,y)\in T} \ell(\langle w, \phi_c(x)\rangle, y)$$

Then

$$\ell_{\mathcal{D}}(\hat{w},\hat{c}) - \ell_{\mathcal{D}}(w^*,c^*) \leq \mathcal{O}\left(\sqrt{\frac{\|w^*\|_2^2+1}{|T|}\log\frac{1}{\delta}} + \sqrt{\frac{1}{|V|}\log\frac{|\mathcal{C}|+1}{\delta}}\right)$$

In the special case of kernel selection we can apply generalization results for learning with random features to show that we can compete with the optimal RKHS from among those associated with one of the configurations (Rudi and Rosasco, 2017, Theorem 1):

Corollary A.3 *In Setting A.2, suppose each configuration $c \in \mathcal{C}$ is associated with a random Fourier feature approximation of a continuous shift-invariant kernel that approximates an RKHS $\mathcal{H}_c$. Suppose ℓ is the squared loss so that $(\hat{w}, \hat{c}) \in \mathcal{W} \times \mathcal{C}$ is obtained as a solution to the following optimization problem:*

$$\operatorname*{arg\,min}_{w \in \mathcal{W}, c \in \mathcal{C}} \quad \ell_V(w, c) \qquad s.t. \qquad w = \operatorname*{arg\,min}_{w \in \mathcal{W}} \quad \lambda \|w\|_2^2 + \sum_{(x,y) \in T} \left(\langle w, \phi_c(x) \rangle - y\right)^2$$

If the number of random features $d = \mathcal{O}(\sqrt{|T|} \log \sqrt{|T|}/\delta)$ and $\lambda = 1/\sqrt{|T|}$ then w.p. $1 - \delta$ we have

$$\ell_{\mathcal{D}}(h_{\hat{w},\hat{c}}) - \min_{c \in \mathcal{C}} \min_{h \in \mathcal{H}_c} \ell_{\mathcal{D}}(h) \leq \mathcal{O}\left(\frac{\log^2 \frac{1}{\delta}}{\sqrt{|T|}} + \sqrt{\frac{1}{|V|} \log \frac{|C| + 1}{\delta}} \right)$$

In the case of neural architecture search we are often solving (unregularized) ERM in the inner optimization problem. In this case we can make an assumption weaker than Assumption A.1, namely that the set of empirical risk minimizers contains a solution that, rather than having low excess risk, simply has low generalization error; then applying Hoeffding's inequality yields the following:

Corollary A.4 *In Setting A.1 let $(\hat{w}, \hat{c}) \in \mathcal{W} \times \mathcal{C}$ be obtained as a solution to the following optimization problem:*

$$\operatorname*{arg\,min}_{w \in \mathcal{W}, c \in \mathcal{C}} \quad \ell_V(w, c) \qquad s.t. \qquad w \in \operatorname*{arg\,min}_{w' \in \mathcal{W}} \ell_T(w', c)$$

Suppose there exists $c^* \in \mathcal{C}$ satisfying $(w^*, c^*) \in \arg\min_{\mathcal{W} \times \mathcal{C}} \ell_{\mathcal{D}}(w, c)$ for some weights $w^* \in \mathcal{W}$ such that w.p. $1 - \delta$ over the drawing of training set $T \sim \mathcal{D}^{|T|}$ at least one of the minima of the optimization problem induced by c^* and T has low generalization error, i.e. $\exists\ w \in \arg\min_{w' \in \mathcal{W}} \ell_T(w', c^*)$ s.t.

$$\ell_{\mathcal{D}}(w, c^*) - \ell_T(w^*, c^*) \leq \varepsilon^*(|T|, \delta)$$

for some error function ε^*. Then we have w.p. $1 - 4\delta$ that

$$\ell_{\mathcal{D}}(\hat{w}, \hat{c}) \leq \ell_{\mathcal{D}}(w^*, c^*) + \varepsilon^*(|T|, \delta) + B\sqrt{\frac{1}{2|V|} \log \frac{1}{\delta}} + B\sqrt{\frac{1}{2|T|} \log \frac{1}{\delta}}$$
$$+ \inf_{\varepsilon > 0} 3\varepsilon + B\sqrt{\frac{2}{|V|}\left(\Lambda(H, \varepsilon, T) + \log \frac{1}{\delta}\right)}$$

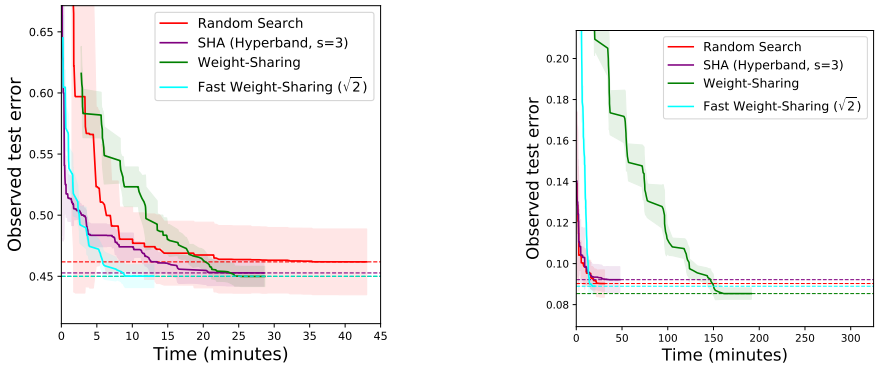

Figure 3: Observed test-error on CIFAR-10 (left) and IMDb (right) as a function of number of time. All curves are averaged over 10 independent trials.

Appendix B. Feature Map Selection Details

B.1 Fast Weight-Sharing Algorithm

In addition to the weight-sharing approach described in Algorithm 1, we evaluate a variant we call *Fast Weight-Sharing* in which at each round t the feature dimension used is a constant multiplicative factor greater than that used on the previous round (we use $\sqrt{2}$, finding doubling to be too aggressive), with the final dimension d reached only at the last round. This is reminiscent of the resource allocation scheme of Li et al. (2018), who after each round of successive elimination give the promoted arms multiplicatively more features. The results, showing that Fast Weight-Sharing outperforms the strong baseline of successive halving and random search, are displayed in Figure 3.

All results are for 10 independent trials of each algorithm with the maximum number of features used by all competing algorithms set to 100K. For both the Weight-Sharing and Fast Weight-Sharing algorithms we started with 256 different configurations. For Figure 1 we varied the feature dimension as shown and considered the correlation between shared-weights performance and standalone performance fo 32 different configurations. The remaining settings are in the provided code: `https://github.com/mkhodak/weight-sharing`.

B.2 Hyperparameter Tuning Setup

For selecting feature maps on CIFAR-10 and IMDb we used the Ridge regression and SVM solvers from `scikit-learn` (Pedregosa et al., 2011) to solve the inner ℓ_2-regularized ERM problems. The regularization was fixed to $\lambda = \frac{1}{2}$ since weight-sharing does not tune non-architectural hyperparameters; the same λ was used for all search algorithms. We tested whether including λ in the search space helped random search and found that it did not, or even caused random search to do worse; this is likely due to the bandwidth parameter playing a similar role.

B.2.1 Random Fourier Features on CIFAR-10

For the search space we used the same kernel configuration settings as Li et al. (2018, Table 5) but replacing the regularization parameter by the option to use the Laplace kernel instead of the Gaussian kernel. The data split used was the standard 40K/10K/10K for training/validation/testing.

B.2.2 Hashed Bag-of-n-Grams on IMDb

For the search space we tuned whether to just tokenize on spaces, split on punctuation, or use the NLTK (Loper and Bird, 2002) tokenizer; whether to remove stopwords; whether to lowercase; the n-gram order between 1-3; whether to binarize bins; whether to weight using Naive-Bayes (Wang and Manning, 2012) or SIF (Arora et al., 2017); the constant α for these weighting schemes between $[10^{-5}, 10^{1}]$ on a logarithmic scale; and whether to use nothing, normlizing, or averaging as preprocessing. The data split was 25K/12.5K/12.5K for training/validation/testing.