PAC-Bayes Compression Bounds So Tight That They Can Explain Generalization

Sanae Lotfi[*] New York University
Sanyam Kapoor[*] New York University
Marc Finzi[*] New York University
Andres Potapczynski[*] New York University
Micah Goldblum New York University
Andrew Gordon Wilson New York University

Abstract

While there has been progress in developing non-vacuous generalization bounds for deep neural networks, these bounds tend to be uninformative about why deep learning works. In this paper, we develop a compression approach based on quantizing neural network parameters in a linear subspace, profoundly improving on previous results to provide state-of-the-art generalization bounds on a variety of tasks, including transfer learning. We use these tight bounds to better understand the role of model size, equivariance, and the implicit biases of optimization, for generalization in deep learning. Notably, we find large models can be compressed to a much greater extent than previously known, encapsulating Occam's razor.

1 Introduction

There has been great effort to understand why deep learning models generalize so well on common datasets, despite the fact that these models have many more parameters than the number of training datapoints and can fit random labels [70]. These observations are not explainable through classical statistical learning theory such as VC dimension or Rademacher complexity which focus on uniform convergence over the hypothesis class [51]. The PAC-Bayes framework, in contrast, gives a convenient way of constructing bounds where the generalization gap depends on the deep learning model found by training rather than the hypothesis set as a whole. Using this framework, many different potential explanations have been proposed drawing on properties of a deep learning model that are induced by the training dataset, such as low spectral norm [55], noise stability [2], flat minima [30], derandomization [53], robustness, and compression [2, 71].

In this work, we show that neural networks are substantially more compressible than previously believed when paired with structured training datasets. Constructing tighter generalization bounds than have been previously achieved, we show that this compression *alone* is sufficient to explain much of the generalization of neural networks.

In particular:

1. We develop a new approach, that we introduce in Section 4 for, training compressed neural networks that adapts the compressed size to the difficulty of the problem. We train in selected random subspaces of the parameters and perform learned quantization. Consequently, we achieve extremely low compressed sizes for neural networks at a given accuracy level, which is a requirement for our tight bounds.
2. Using a prior encoding Occam's razor and our compression scheme, we construct the best generalization bounds to date on image datasets, both with data-dependent and data-

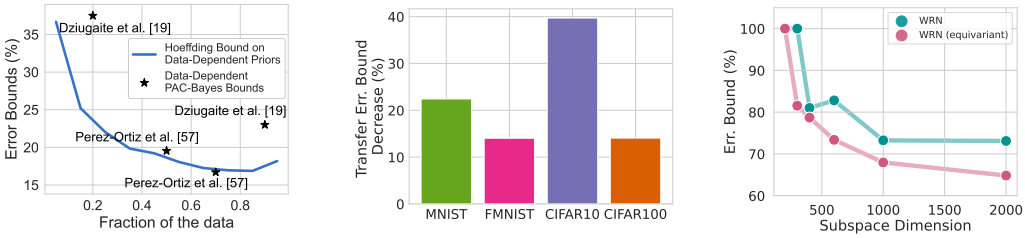
[*]Equal contribution.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

independent priors as we show in Section 5. We also show how transfer learning improves compression and thus our generalization bounds, explaining the practical performance benefits of pre-training.

3. The success of our bounds shows that data-independent bounds can compete with data-dependent bounds while simultaneously conferring significantly stronger explanatory power for why neural networks generalize as we show in Section 5.2.
4. Through the lens of compression and our bounds, we are able to help explain in Section 6 why deep learning models generalize on structured datasets like CIFAR-10, but not when structure is broken such as by shuffling the pixels or shuffling the labels, the benefits of equivariant models, why CNNs outperform MLPs, double descent, and whether the implicit regularization of SGD is necessary for generalization.

While we achieve state-of-the-art bounds, our focus is on how we can use bounds to better understand generalization behaviour, rather than achieve good bounds in isolation. Figure 1 illustrates several of our observations, regarding data dependent bounds, the effect equivariance, and transfer learning. We describe these findings, as well as several others enumerated above, in detail. Our code is available here.



(a) Pitfalls of data-dependent bounds (b) Transfer learning bounds (c) test

Figure 1: **(a)** We show that the Hoeffding bound computed on *data-dependent priors only* and evaluated on the remainder of the training data achieves error bounds that are competitive or even better than data-dependent bounds obtained by previous works. This shows that data-dependent PAC-bounds do not explain generalization any further than the pre-trained model *solely* does, without explaining the generalization process obtained through fine-tuning the model, and hence posterior contraction. **(b)** We show the reduction in data-independent error bounds between pre-trained models and models trained from scratch. Transfer: SVHN to MNIST, CIFAR-10 to FMNIST, ImageNet to both CIFAR-10 and CIFAR-100. Our bounds theoretically prove that transfer learning improve generalization in image classification. **(c)** We compare rotationally equivariant C_8 WideResNet against a WideResNet with the same number of parameters on a rotationally equivariant RotMNIST dataset. The rotationally equivariant models are more compressible and therefore have better bounds than their non-equivariant counterparts when paired with a dataset that also has the rotational symmetry. Our bounds hence provide theoretical support for learning equivariances.

2 Related Work

We delineate recent techniques for achieving non-vacuous PAC-Bayes bounds for neural networks.

Optimizing the PAC-Bayes Bound Dziugaite and Roy [18] obtain the first non-vacuous generalization bounds for deep stochastic neural networks on binary MNIST. To do so, the authors construct a relaxation of the Langford and Seeger [40] bound and optimize it with the objective of finding a posterior distribution that covers a large volume of low-loss solutions around a local minimum obtained using SGD. Rivasplata et al. [58] further extend the idea by developing novel relaxations of PAC-Bayes bounds based on Blundell et al. [6].

Model Compression and PAC-Bayes Bounds Noting the robustness of neural networks to small perturbations [29, 30, 39, 38, 36, 55, 8], Arora et al. [2] develop a compression-based approach that uses noise stability and the ability to reconstruct weight matrices with random projections is used to study generalization of neural networks. Subsequently, Zhou et al. [71] develop a PAC-Bayes bound that uses the representation of a compressed model in bits, and add noise stability through the use

Table 1: Recent non-vacuous PAC-Bayes bounds obtained on popular image classification datasets in deep learning. $\star$ indicates bounds obtained using data-dependent priors (Section 5.2). A ✗ signifies that either the method does not support multi-class problems or that it is completely reliant on data dependent priors and therefore cannot result in data-independent bounds. Additionally, we add Binary MNIST for reference to a benchmark used in earlier works.

	Non-vacuous PAC-Bayes bounds (%)				
Reference	Binary MNIST	MNIST	FMNIST	CIFAR-10	CIFAR-100
Dziugaite and Roy [18]	16.1	✗	✗	✗	✗
Rivasplata et al. [58]	2.2	✗	✗	✗	✗
Zhou et al. [71]		46	91.6	100	100
Dziugaite et al. [19]		$11^\star$	$38^\star$	$23^\star$	✗
Pérez-Ortiz et al. [57]		$21.7/1.5^\star$	49.1	$90.0/16.7^\star$	100
Our bounds		$\mathbf{11.6/1.4}^*$	$\mathbf{32.8/10.1}^*$	$\mathbf{58.2/16.6}^*$	$\mathbf{94.6/44.4}^*$

of Gaussian posteriors and Gaussian mixture priors. Furthermore, they achieve the smallest model representations through pruning and quantization [27, 10].

Data-Dependent Priors Dziugaite et al. [19] demonstrate that for linear PAC-Bayes bounds such as Thiemann et al. [62], a tighter bound can be achieved by choosing the prior distribution to be data-dependent, i.e., the prior is trained to concentrate around low loss regions on held-out data. More precisely, the authors show that the optimal data-dependent prior is the conditional expectation of the posterior given a subset of the training data. They approximate this data-dependent prior by solving a variational problem over Gaussian distributions. They evaluate the bounds for SGD-trained networks on data-dependent priors obtaining tight bounds on MNIST, Fashion MNIST, and CIFAR-10. In a similar vein, Pérez-Ortiz et al. [57] combine data-dependent priors [19] with the PAC-Bayes with Backprop (PBB) [58] to obtain *state-of-the-art* PAC-Bayes non-vacuous bounds for MNIST and CIFAR-10 using data-dependent priors.

Table 1 summarizes the PAC-Bayes bounds obtained by these works alongside our bounds.

Our upper bound on the KL term closely follows Zhou et al. [71], but with key improvements: (i) we train in lower dimensional subspaces using intrinsic dimensionality and FiLM subspaces which proves to be more effective and adaptable than pruning, (ii) we consider a more effective quantization scheme with variable length codes and quantization aware training, and (iii) we take advantage of the increased compression provided by transfer learning and data dependent priors.

As a result, we obtain state-of-the-art non-vacuous bounds for MNIST, Fashion MNIST, CIFAR-10, and CIFAR-100 in both the from scratch and transfer learning setting as shown in Table 1. Motivated by the tightness of these bounds, we seek to better understand the role of equivariance, stochastic optimization, and data dependent bounds, for explaining generalization.

Ding et al. [16] is another closely related work which optimizes the PAC-Bayes bounds to predict the transferability of various upstream models to a downstream task. In contrast, we aim to explain why neural networks generalize through compression. This is different than predicting transferability as there are several factors other than compression (which we demonstrate is sufficient, but not necessary for generalization) that could make a model perform well on a downstream task.

3 A Primer on PAC-Bayes Bounds

PAC-Bayes bounds are fundamentally an expression of Occam's razor: simpler descriptions of the data generalize better. As an illustration, consider the finite hypothesis generalization bound. Let $\hat{R}(h) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(h\left(x_i\right), y_i\right)$ be the empirical risk of a hypothesis h from a finite hypothesis set $\mathcal{H}$ for the 0-1 loss ℓ, and $R(h) = \mathbb{E}[\hat{R}(h)]$ its population risk. With probability at least $1-\delta$, the population risk for a hypothesis h using n data samples satisfies

$$R(h) \leq \hat{R}(h) + \sqrt{\frac{\log|\mathcal{H}| + \log(1/\delta)}{2n}}, \tag{1}$$

bounded by the empirical risk and a complexity term $\log|\mathcal{H}|$ counting the number of bits needed to specify any $h \in \mathcal{H}$.

But what if we don't believe that each hypothesis is equally likely? If we consider that a given prior distribution P over $\mathcal{H}$ concentrates around the optimal hypothesis, then we can construct a variable length code that uses fewer bits for the more likely hypothesis. With the best possible code for P, a given hypothesis h will take $\log_2 \frac{1}{P(h)}$ bits to represent. Using this prior may only results in a smaller complexity term even if the hypothesis set is large, so long as the hypotheses that are consistent with the data are also likely under the prior.

The number of bits required can further be reduced from $\log_2 \frac{1}{P(h)}$ to $\mathbb{KL}(Q \parallel P)$ by considering a distribution of "good" solutions Q. If we don't care which element of Q we arrive at, we can gain some bits *back* from this insensitivity (which could be used to code a separate message). The average number of bits to code a sample from Q using the prior P is the cross entropy $\mathbb{H}(Q, P)$, and one gets on average the entropy $H(Q)$ bits back from being agnostic about which sample $h \sim Q$ to use, yielding the KL-divergence between Q and P: $\mathbb{H}(Q, P) - \mathbb{H}(Q) = \mathbb{KL}(Q \parallel P)$.

With these improvements of the finite hypothesis bound: replacing $\log |\mathcal{H}|$ with $\mathbb{KL}(Q \parallel P)$, and sampling a hypothesis $h \in \mathcal{H}$, one arrives (with minor bookkeeping) at the PAC-Bayes bound introduced in McAllester [49]. With probability at least $1 - \delta$,

$$\underset{h \sim Q}{\mathbb{E}} [R(h)] \leq \underset{h \sim Q}{\mathbb{E}} [\hat{R}(h)] + \sqrt{\frac{\mathbb{KL}(Q \parallel P) + \log(n/\delta) + 2}{2n - 1}}. \tag{2}$$

Many refinements of Eq. (2) have been developed [40, 48, 7, 62] but retain the same character. The lower the ratio of the KL to the number of data points n, the tighter the gap between empirical and expected risk. In this work, we use the tighter Catoni [7] variant of the PAC-Bayes bound (see Appendix J for details).

Universal Prior. Connecting back to Occam's razor, one can define a prior that explicitly penalizes the minimum compressed length of the hypothesis, also known as the universal prior [60]: $P(h) = 2^{-K(h)}/Z$, where K is the *prefix* Kolmogorov complexity [33] of h (the length of the shortest program that produces h and also delimits itself), and $Z \leq 1$.[2] Using a point mass posterior on a single hypothesis h^*, $\mathbb{KL}(Q \parallel P)$ from the bound reduces to

$$\mathbb{KL}\left(\mathbf{1}_{[h=h^*]} \parallel P(h)\right) = \log \frac{1}{P(h^*)} \leq K(h^*) \log 2 \leq l(h^*) \log 2 + 2 \log l(h^*),$$

where $l(h)$ is the length of a given program that reproduces h not including the delimiter, which we can use when upper bounding the KL. For convenience, we can condition on using the same method for compression and decompression for all elements of the prior. If we can reduce the compressed length $l(h)$ of the hypotheses we find that fits the training data, then we can improve the tightness of the PAC-Bayes bound.

Model Compression. *Model compression* aims to find a nearly equivalent model that can be expressed in a fewer bits and is useful for deployments on mobile devices or for improving inference time on specialized hardware [9, 10]. For computing PAC-Bayes generalization bounds, however, only the model size matters. Therefore, we can employ compression methods which may otherwise be unfavorable in practice due to worse computational requirements. *Pruning* and *quantization* are among the most widely used methods for model compression. In this work, we rely on quantization (Section 4.2) to achieving tighter generalization bounds.

4 Tighter Generalization Bounds via Adaptive Subspace Compression

Training a neural network involves taking many gradient steps in a high-dimensional space $\mathbb{R}^D$. Although D may be large, the loss landscape has been found to be simpler than typically believed [14, 26, 23]. Analogous to the notion of intrinsic dimensionality more generally, Li et al. [44] attempt to find the lowest dimensional subspace in which the network can be trained and still fit the training data. The weights of a neural network $\theta \in \mathbb{R}^D$ are parametrized in terms of an initialization θ_0 and a projection $w \in \mathbb{R}^d$ to a lower dimensional subspace through a fixed matrix $P \in \mathbb{R}^{D \times d}$,

$$\theta = \theta_0 + Pw. \tag{3}$$

[2] The universal prior similar to the discrete hypothesis prior from Zhou et al. [71] but setting $m(h) = 2^{-2 \log_2 l(h)}$ rather than the flat $m(h) = 2^{-72}$.

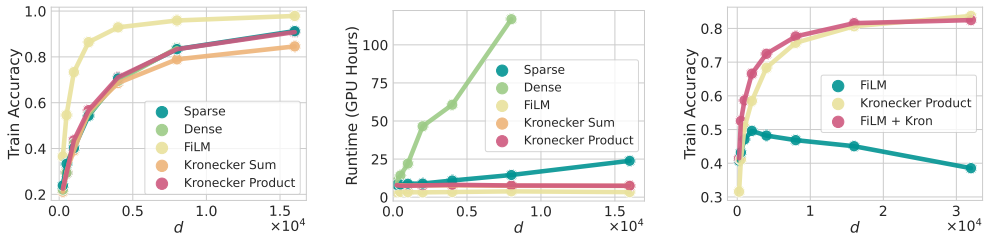


Figure 2: **(Left)** Different projection operators P (Section 4.1) used for transfer learning from Imagenet to CIFAR-10 on a ResNet-34 across different subspace dimensions d. Kronecker product, Sparse, and Dense perform almost identically **(Center)** Kronecker product runs with substantially reduced the runtime cost compared to the Sparse or Fastfood matrices used by Li et al. [44]. **(Right)** Training from scratch on CIFAR-10. The FiLM projector alone is unable to fit the data when training from scratch, and instead a sum of FiLM and Kronecker Product projectors perform the best.

To facilitate favorable conditioning during optimization, P is chosen to be approximately orthonormal $P^\top P \approx I_{d\times d}$ and for scalability, Li et al. [44] utilize random normal matrices of the form $P \sim \mathcal{N}(0,1)^{D\times d}/\sqrt{D}$, and their sparse approximations [45, 42].

In its original form, intrinsic dimensionality (ID) is used as a scientific tool to measure the complexity of the learning task. Unlike other methods like pruning, ID has the benefit of scaling dramatically up or down in size [$^{\text{S}}_{\text{K}}$] with difficulty of the problem. We find that ID combined with quantization can serve as an effective model compression method. We note that ideas similar to the intrinsic dimensionality of a model have been explored in the context of model compression for estimating bounds [2] [$^{\text{S}}_{\text{K}}$] [$^{\text{M}}_{\text{F}}$]. For our work, the ability to find the intrinsic dimension $d \ll D$ has profound implications for the compressibility of the models, and therefore our ability to construct generalization bounds. As demonstrated by Zhou et al. [71], the compressibility of neural network models has a direct connection to generalization and allows us to compute non-vacuous PAC-Bayes bounds for transfer learning. Building upon ID, we describe our approach next. [$^{\text{S}}_{\text{K}}$]

4.1 Finding Better Random Subspaces to Train In

To further improve upon the scalability and effectiveness of the projections P used by Li et al. [44], we introduce three novel methods.

Kronecker Sum Projector: Using the Kronecker product $\otimes$, we construct the matrix $P_\oplus = (\mathbf{1}\otimes R_1 + R_2 \otimes \mathbf{1})/\sqrt{2D}$ where $R_1, R_2 \sim \mathcal{N}(0,1)^{\sqrt{D}\times d}$ and $\mathbf{1}$ is the vector of all ones in $\mathbb{R}^{\sqrt{D}}$. From the fact that $R_1 \perp\!\!\!\perp R_2$ and the entries are standard normal, one can verify that $P_\oplus^\top P_\oplus = I_{d\times d} + \mathcal{O}(1/\sqrt{D})$.

Kronecker Product Projector: Alternatively, we can form the matrix $P_\otimes = Q_1 \otimes Q_2/\sqrt{D}$ with the smaller $Q_1, Q_2 \sim \mathcal{N}(0,1)^{\sqrt{D}\times\sqrt{d}}$, and again this matrix is approximately orthogonal: $P_\otimes^\top P_\otimes = (Q_1^\top Q_1/\sqrt{D}) \otimes (Q_2^\top Q_2/\sqrt{D}) = I \otimes I + O(D^{-1/4}) = I_{d\times d} + O(D^{-1/4})$. [3]

The matrix vector multiply $w \mapsto Pw$ for both the above projectors can be performed in time $O(d\sqrt{D})$ and $O(\sqrt{dD})$ respectively, rather than the $O(dD)$ that is required by the dense random matrix. Figure 2 demonstrates the runtime speedup and training performance improvement in comparison to the methods using by Li et al. [44]. Notably, the Kronecker-structured projections retain the fidelity of the dense random matrix while being orders of magnitude faster than the alternative operators when scaling to larger values of d.

FiLM projector: BatchNorm parameters have an outsized effect on the downstream task performance relative to their size, and this fact has been used in Featurewise independent Linear Mod-

[3] As neither D nor d is typically a perfect square, we concatenate a dense random matrix to pad out the difference between D, d, and a perfect square. As $(\sqrt{D}+1)^2 = D + 2\sqrt{D} + 1$, we have that the size of this padding is at most $\sqrt{D}\times\sqrt{d}$, so it does not increase the asymptotic cost of performing the matrix vector multiplies.

ulation (FiLM) [56, 17] for efficient control of neural networks in many different settings. Several authors have explored performing finetuning for transfer learning solely on these parameters and the final linear layer [35]. Drawing on these observations, we construct a projection matrix P_{FiLM} where only columns corresponding with BatchNorm or head parameters are non-zero and sampled from $\mathcal{N}(0,1)^d/\sqrt{D}$, which we also show in Figure 2. While the FiLM projector is highly effective for transfer learning (shown in Figure 2 left), the performance saturates quickly when training from scratch. For this reason, when training from scratch we employ the sum $P_{\text{FiLM}+\otimes} = (P_{\text{FiLM}} + P_{\otimes})/\sqrt{2}$, which outperforms the two projectors individually as shown in Figure 2 right.

4.2 Quantization Scheme and Training

Through quantization, the average number of bits used per parameter can be substantially reduced. When optimizing purely for model size rather than efficiency on specialized hardware, one can choose non-linearly spaced quantization levels which are learned, and use variable length coding schemes as shown in Han et al. [27]. On the other side, the straight through estimator has been central to learning weights in binary neural networks [31]. We combine these ideas to simultaneously optimize the quantized weights and the quantization levels for maximum compression.

Given the full precision weights $w = [w_1, \ldots, w_d] \in \mathbb{R}^d$ and a vector $c = [c_1, \ldots c_L] \in \mathbb{R}^L$ of L quantization levels, we construct the quantized vector $\hat{w} = [\hat{w}_1, \ldots, \hat{w}_d]$ such that $\hat{w}_i = c_{q(i)}$ where $q(i) = \arg\min_k |w_i - c_k|$. The quantization levels c are learned alongside w, where the gradients are defined using the straight through estimator [3, 68]:

$$\frac{\partial \hat{w}_i}{\partial w_j} = \delta_{ij} \quad \text{and} \quad \frac{\partial \hat{w}_i}{\partial c_k} = 1_{[q(i)=k]} \tag{4}$$

We initialize c with uniform spacing between the minimum and maximum values in parameter vector w, which performs better than k-means initialization [11]. To further compress the network, we use a variable length code in the form of arithmetic coding [47], which takes advantage of the fact that certain quantization levels are more likely than others. Given probabilities p_k (empirical fractions) for cluster c_k, arithmetic coding of w takes at most $\lceil d \times \mathbb{H}(p) \rceil + 2$ bits, where $\mathbb{H}(p)$ is the entropy $\mathbb{H}(p) = -\sum_k p_k \log_2 p_k$. For a small number of quantization levels, we find it makes a small improvement in the compression over more common Huffman coding.

In summary, we use $\lceil d \times \mathbb{H}(p) \rceil + 2$ bits for coding the quantized weights $\hat{w}$, $16L$ bits for the codebook c (represented in half precision), and additional $L \times \lceil \log_2 d \rceil$ bits for representing the probabilities p_k, arriving at $l(w) \leq \lceil d \times \mathbb{H}(p) \rceil + L \times (16 + \lceil \log_2 d \rceil) + 2$.

5 Empirical Non-Vacuous Bounds

5.1 Non-Vacuous PAC-Bayes Bounds

To compute our bounds on benchmark datasets, we use the joint approach of projected subspace training, parameter quantization, and optimizing the subspace dimension with respect to the bounds as summarized in Algorithm 1. We also optimize other bound hyperparameters such as the learning rate of the quantization-aware training. We offer more details about bound optimization in Appendix E. Our results for *data-independent* priors, i.e., priors that are not pre-trained on a subset of the training data, are shown in Table 2. We derive the first non-vacuous bounds on FashionMNIST, CIFAR-10, and CIFAR-100 without data-dependent priors. These results are of significance as we argue in Section 5.2 that using data-dependent priors are not explanatory about the learning process. In particular, we improve over the compression bound results obtained by Zhou et al. [71] on MNIST from $< 46\%$ to 11.55%. We also dramatically improve their compression results as we reduce the compressed size for the best bound by 94% bringing it down from 6.23 KB to 0.38KB for MNIST with LeNet5. The tightness of our SOTA subspace compression bounds allows us to improve the understanding of several deep learning phenomena as discussed in Section 6. See Appendix B for additional results.

Algorithm 1 Compute PAC-Bayes Bound.

Inputs: Neural network f_θ, Training dataset $\{x_i, y_i\}_{i=1}^n$, Clusters L, Intrinsic dimension d, Confidence $1-\delta$, and Prior distribution P.
function COMPUTE_BOUND($f_\theta, L, d, (x_i, y_i)_{i=1}^n, \delta, P$)
    $w \leftarrow$ TRAIN_ID(f_θ, d, $(x_i, y_i)_{i=1}^n$) ▷ (Section 4.1)
    $\hat{w} \leftarrow$ TRAIN_QUANTIZE(w, L, $(x_i, y_i)_{i=1}^n$)
    Compute quantized train error $\hat{R}(\hat{w})$.
    $\mathbb{KL}(Q, P) \leftarrow$ GET_KL($\hat{w}, P$) ▷ (Section 3)
    return GET_CATONI_BOUND($\hat{R}(\hat{w})$, $\mathbb{KL}(Q, P)$, δ, n) ▷ (Section 3)
end function
function TRAIN_QUANTIZE($w, L, (x_i, y_i)_{i=1}^n$) ▷ (Section 4.2)
    Initialize $c \leftarrow$ GET_CLUSTERS(w, L)
    for $i = 1$ to quant_epochs **do**
        $c \leftarrow c - \rho \nabla_c \mathcal{L}(w, c)$ and $w \leftarrow w - \rho \nabla_w \mathcal{L}(w, c)$
    end for
    return $\hat{w}$
end function
function GET_KL($(\hat{w}, P)$)
    c, count $\leftarrow$ GET_UNIQUE_VALS_COUNTS($\hat{w}$)
    message_size $\leftarrow$ DO_ARITHMETIC_ENCODING($\hat{w}$, c, count)
    message_size $\leftarrow$ message_size + hyperparam_search ▷ (Appendix A)
    return message_size $+ 2 \times \log$(message_size)
end function

Table 2: Our PAC-Bayesian Subspace Compression Bounds compared to *state-of-the-art* (SoTA) bounds. All results are obtained with 95% confidence, i.e. $\delta = .05$. The sign † refers to data-independent SOTA numbers that we computed using [57], which we run on the additional datasets.

Dataset	Data-independent priors		Data-dependent priors	
	Err. Bound (%)	SoTA (%)	Err. Bound (%)	SoTA (%)
MNIST	**11.6**	21.7 [57]	**1.4**	1.5 [57]
+ SVHN Transfer	**9.0**	16.1†		
FashionMNIST	**32.8**	46.5†	**10.1**	38 [19]
+ CIFAR-10 Transfer	**28.2**	30.1†		
CIFAR-10	**58.2**	89.9†	**16.6**	16.7 [57]
+ ImageNet Transfer	**35.1**	54.2†		
CIFAR-100	**94.6**	100†	**44.4**	–
+ ImageNet Transfer	**81.3**	98.1†		

5.2 Data-Dependent PAC-Bayes Bounds

So far, we demonstrated the strength of our bounds on *data-independent* priors, where we considerably improve on the state-of-the-art. However, a number of recent papers have considered data-dependent priors as a way of achieving tighter bounds [57, 19]. In this setup, the training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ is partitioned into two parts, $\mathcal{D}_a$ and $\mathcal{D}_b$, with length $n-m$ and m. The first dataset is used to construct a data dependent prior $P(h \mid \mathcal{D}_a)$, and then the bound is formed over the remaining part of the process: the adaptation of the prior $P(h \mid \mathcal{D}_a)$ to the posterior $Q(h)$ using the data $\mathcal{D}_b$. The empirical risk is computed over $\mathcal{D}_b$ only. Intuitively, using dataset $\mathcal{D}_a$ it is possible to construct a much tighter prior over the possible neural network solutions.

In our setting, similar to transfer learning, we use the prior $P_{\mathcal{D}_a}(\theta) = 2^{-K(\theta|\theta_{\mathcal{D}_a})}/Z$ where for compression we use $\theta = \theta_{\mathcal{D}_a} + Pw$, and $\theta_{\mathcal{D}_a}$ is the solution found by training the model (without random projections) on the data $\mathcal{D}_a$ rather than initializing randomly. With these data-dependent priors, we achieve the best bounds in Table 2.

However, our adaptive approach exposes a significant downside of data-dependent priors. To the extent that PAC-Bayes bounds can be used for explanation, in this case they only yield insights into the procedure used to adapt $P_{\mathcal{D}_a}(\theta)$ to Q using $\mathcal{D}_b$, any learning that is done in finding $P_{\mathcal{D}_a}(\theta)$ is not constrained or explained by the bound (we have no information on the compressibility of $\theta_{\mathcal{D}_a}$). Given the ability to adapt the size of the KL to the difficulty of the problem, it is possible to squeeze all of the learning into $P_{\mathcal{D}_a}(\theta)$ and none in this adaption to Q. This happens as the $KL \to 0$, which we find happens empirically (or very nearly so) for many splits of the data where $n - m$ is large. Setting $Q(\theta) = \mathbf{1}_{[\theta=\theta_{\mathcal{D}_a}]}$, the KL has only the contribution from the optimization over d: $\mathbb{KL}(Q||P_{\mathcal{D}_a}) \leq \log D$ we find that the bound is nothing more than a variant of the simple Hoeffding bound where $\mathcal{D}_b$ is the validation set $R\left(\theta_{\mathcal{D}_a}\right) \leq \hat{R}_{\mathcal{D}_b}\left(\theta_{\mathcal{D}_a}\right) + \sqrt{\frac{\log(Dm/\delta)+2}{2m-1}}$.

We can see this phenomenon in where we compare existing data-dependent bounds to the simple Hoeffding bound which relies only on validation accuracy to guarantee generalization. We can consider the Hoeffding bound as the simplest data-dependent bound without any fine-tuning so that the *prior*, a single pre-trained checkpoint, is directly evaluated on held-out validation data and the KL-divergence term can be shed. If another data-dependent bound cannot achieve significantly stronger guarantees than the Hoeffding bound, then it only explains that neural networks generalize because they have low validation error which is no explanation for generalization at all. Indeed, we see in Figure 1 that the strength of existing data-dependent bounds relies entirely on validation performance and not on other informative properties of neural networks like compressibility.

This phenomenon can also be seen in Appendix D, where optimization over $d = 1, ..., D$ leads to very small values of the KL when $n - m$ is a large fraction of n, and hence highly simple solutions approaching the Hoeffding bound or confidence interval. This demonstrates that data-dependent priors based PAC-Bayes bounds are not particularly meaningful, and attempts to optimize these bounds will lead inevitably to the statistical bounds formed from the validation set such as a simple Hoeffding bound or confidence interval. While these bounds are correct, they are only informative about the difference in performance of the pretrained model on validation data and test data.

Note that this is in contrast with the data-independent bounds that constrain and explain generalization for the entirety of the learning process. Similarly, our transfer learning bounds meaningfully constrain what happens in the finetuning on the downstream task, but they do not constrain the prior determined from the upstream task.

5.3 Non-Vacuous PAC-Bayes Bounds for Transfer Learning

By directly interpreting PAC-Bayes bounds through the lens of compression, we immediately see the benefits of employing an upstream dataset for transfer learning. Transfer learning allows us to constrain the prior around parameters consistent with the upstream task, and hopefully the downstream task as well, reducing the KL-divergence between the prior and the posterior and leading to even tighter bounds as we show in Table 2. We obtain unequivocally tighter bounds using transfer learning for data-independent priors, providing a theoretical certification that transfer learning can improve generalization. Our PAC-Bayes transfer learning approach also indicates that transfer learning can boost generalization whenever codings optimized on a pre-training task are more efficient for encoding a downstream posterior than an a priori guess made before seeing data. In contrast, downstream tasks which differ greatly from the upstream task may only be consistent with models that are not compressible under the learned prior, a scenario which may describe negative transfer. See Appendix C for more details.

6 Understanding Generalization through PAC-Bayes Bounds

The classical learning theory viewpoint of uniform convergence focuses on properties of the hypothesis class as a whole, such as its size. In contrast, the PAC-Bayes perspective shows that the ability to generalize is not merely a result of the hypothesis class but also results from the exact dataset at hand and the individual functions in the hypothesis class which we fit to that dataset. After all, many elements of our hypothesis class are not compressible, yet in order to guarantee generalization, we need to choose the ones that are. Real datasets actually contain a tremendous amount of structure, or else we could not learn from them as famously argued by Hume [32] and No Free Lunch theorems [65, 25]. This high degree of structure in real-world datasets is reflected in the compressibility of the

functions (i.e. neural networks) we find in our hypothesis class which fit them. In this section, we examine exactly how dataset structure manifests in compressible models by applying our generalization bounds, and we see what happens when this structure is broken, for example by shuffling pixels or fitting random labels. Corrupting the dataset degrades its compressibility, and generalization with it.

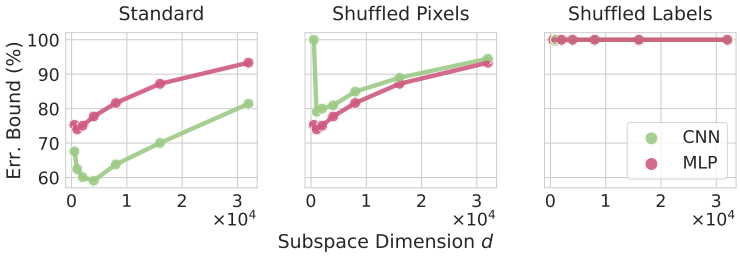


Figure 3: Our PAC-Bayes bound computed using various subspace dimensions for a fixed size convolutional network and MLP, both with 500k parameters. We train on **(left)** CIFAR-10, **(center)** CIFAR-10 with shuffled pixels, **(right)** CIFAR-10 with shuffled labels. Structure in the dataset induces structure in the model. As structure is removed from the dataset, models which fit the data become much less compressible, hence generalize worse.

MLPs vs CNNs: It is well known that convolutional neural networks (CNNs) generalize much better than standard multilayer perceptrons (MLPs) with alternating fully-connected layers and activation functions on image classification problems, even when controlling for the number of parameters. We find that this fact is reflected in the improved compressibility of CNNs and consequently our generalization bounds, shown in Figure 3 (left) on the image dataset CIFAR-10 using the same number of parameters for the two models. The CNN is able find a more structured and lower description length explanation of the training data.

Shuffled Pixels: However, when this image structure is broken by shuffling the pixels, we find that CNNs are no better at generalizing than MLPs, and this is reflected in their compressed sizes. The CNNs become substantially less compressible when trained on this dataset, and hence our bounds show them generalizing worse than MLPs, see Figure 3 (center). MLPs do not suffer when this structure is broken since they never used it in the first place.

Shuffled Labels: When the structure of the dataset is entirely broken by shuffling the labels, the compressibility of the models (both for CNNs and MLPs) which fit the random labels is lost. Regardless of the subspace dimension used, our generalization bounds are all at 100% error as shown in Figure 3 (right). It is not possible to fit the training data using low subspace dimensions, and when using a large enough dimension to fit the data, the compressed size of the model is larger than the training data and hence the generalization bounds are vacuous.

Equivariance: Designing models which are *equivariant* to certain symmetry transformations has been a guiding principle for the development of data-efficient neural networks in many domains [12, 13, 63, 64, 22, 34]. While intuitively it is clear that respecting dataset symmetries severely improves generalization, relatively little has been proven for neural networks [21, 72, 4, 20]. We compress and evaluate rotationally equivariant and non-equivariant Wide ResNets [64, 69] trained on MNIST and a rotated version of MNIST. As shown in Figure 1 (c), the rotationally equivariant models are more compressible and provably generalize better than their non-equivariant counterparts when paired with a dataset that also has the rotational symmetry. We expand further on these experiments in Appendix F.

Is Stochasticity Necessary for Generalization? It is widely hypothesized that the implicit biases of SGD help find solutions which generalize better. For example, Arora et al. [1] argue that there is no regularizer that replicates the benefits of gradient noise. Wu et al. [66], Smith et al. [59], and Li et al. [46] advocate that gradient noise is necessary to achieve state-of-the-art performance. In contrast, recent work by Geiping et al. [24] shows that full-batch gradient descent can match state-of-the-art performance.

We train ResNet-18 and LeNet5 models on CIFAR-10 and MNIST, respectively, using full-batch and stochastic gradient descent with different intrinsic dimensionalities. We provide the training details

in Appendix G, but it is worth noting that, unlike Geiping et al. [24], we do not use flatness-seeking regularization for full-batch training. For MNIST with LeNet5, the best generalization bounds that we obtain are 11.55% and 11.20% using stochastic gradient descent (SGD) and full-batch training respectively. The best generalization bounds that we obtain for CIFAR-10 with ResNet-18 are 74.68% and 75.3% using SGD and full-batch training respectively. We also extend this analysis to SVHN to MNIST transfer learning with LeNet5 and obtain PAC-Bayes bounds of 9.0% and 8.7% using SGD and full-batch training respectively.

These close theoretical guarantees on the generalization error for both SGD and full-batch training suggest that while the implicit biases of SGD may be helpful, they are not at all necessary for understanding why neural networks generalize well. We expand this analysis in Appendix G.

Double Descent: Our bounds are also tight enough to predict the double descent phenomenon noted in Nakkiran et al. [52]. See Appendix I for a depiction of these experiments and a discussion of their significance.

7 Discussion

In this work, we have constructed a new method for compressing deep learning models which is highly adaptive to the level of structure in the model and the training dataset. According to the Occam's prior which considers shorter compressed length models to be more likely, we construct state-of-the-art generalization bounds across a variety of settings. Through our bounds and the viewpoint of Occam's razor and compressibility, we show how generalization relates to the structure in the dataset and the structure in the model, and we are able to explain aspects of neural network generalization for natural image datasets, shuffled pixels, shuffled labels, equivariant models, and for stochastic training.

Limitations We note that despite the power of this compression scheme and our bounds, there are certain aspects of deep learning generalization that remain unexplained. Our compression bounds still show a preference towards models with a smaller number of base parameters as shown in Appendix H, despite the fact that larger models tend to generalize better. While we have achieved better model compression than previous works, it is unlikely that we are close to theoretical limits. We hope that with a further improved compression scheme (possibly using nonlinear parameter projections), we might find that larger deep learning model solutions are compressible to smaller sizes than smaller models. There is also a question of whether models not trained in a special way can be related and bounded by the performance of the compressed models, perhaps leveraging ideas from Nagarajan and Kolter [50] and others investigating this from a theoretical lens, and this would be helpful for applying our findings more broadly. Additionally, while our bounds show that the compressibility of our models implies generalization, we make no claims about the reverse direction. We believe however that the compression approach and Occam's razor have yet untapped explanatory power in deep learning.

Acknowledgements

This research is supported by an Amazon Research Award, NSF I-DISRE 193471, NIH R01DA048764-01A1, NSF IIS-1910266, and NSF 1922658 NRT-HDR: FUTURE Foundations, Translation, and Responsibility for Data Science. This work is supported in part through the NYU IT High Performance Computing resources, services, and staff expertise.

References

[1] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 244–253. PMLR, 2018. 9

[2] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*,

volume 80 of *Proceedings of Machine Learning Research*, pages 254–263. PMLR, 2018. 1, 2, 5

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients through Stochastic Neurons for Conditional Computation. *Preprint arXiv:1308.3432v1*, 2013. 6

[4] Alberto Bietti, Luca Venturi, and Joan Bruna. On the sample complexity of learning under geometric stability. *Advances in Neural Information Processing Systems*, 34, 2021. 9

[5] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com. 20

[6] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1613–1622. JMLR.org, 2015. 2

[7] Olivier Catoni. PAC-Bayesian Supervised Classification: the Thermodynamics of Statistical Learning. *Institute of Mathematical Statistics Lecture Notes*, 2007. 4, 23

[8] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer T. Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 2

[9] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017. 4

[10] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 35(1):126–136, 2018. 3, 4

[11] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the limit of network quantization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 6

[12] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2990–2999. JMLR.org, 2016. 9

[13] Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 9

[14] Yann N. Dauphin, Razvan Pascanu, Çaglar Gülçehre, KyungHyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2933–2941, 2014. 4

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009. 19, 24

[16] Nan Ding, Xi Chen, Tomer Levinboim, Beer Changpinyo, and Radu Soricut. Pactran: Pac-bayesian metrics for estimating the transferability of pretrained models to classification tasks. *arXiv preprint arXiv:2203.05126*, 2022. 3

[17] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. *Distill*, 3:e11, 2018. 6

[18] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017. 2, 3

[19] Gintare Karolina Dziugaite, Kyle Hsu, Waseem Gharbieh, Gabriel Aprino, and Daniel M. Roy. On the Role of Data in Pac-Bayes Bounds. *The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021. 3, 7, 17

[20] Bryn Elesedy. Group symmetry in pac learning. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022. 9

[21] Bryn Elesedy and Sheheryar Zaidi. Provably strict generalisation benefit for equivariant models. In *International Conference on Machine Learning*, pages 2959–2969. PMLR, 2021. 9

[22] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3165–3176. PMLR, 2020. 9

[23] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P. Vetrov, and Andrew Gordon Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8803–8812, 2018. 4

[24] Jonas Geiping, Micah Goldblum, Phillip E. Pope, Michael Moeller, and Tom Goldstein. Stochastic Training Is Not Necessary For Generalization. *The 10th International Conference on Learning Representations (ICLR)*, 2022. 9, 10, 19

[25] Christophe Giraud-Carrier and Foster Provost. Toward a justification of meta-learning: Is the no free lunch theorem a show-stopper. In *Proceedings of the ICML-2005 Workshop on Meta-learning*, pages 12–19, 2005. 8

[26] Ian J. Goodfellow and Oriol Vinyals. Qualitatively characterizing neural network optimization problems. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 4

[27] Song Han, Huizi Mao, and William J. Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *The 4th International Conference on Learning Representations (ICLR)*, 2016. 3, 6, 17

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. 19

[29] Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13, 1993. 2

[30] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997. 1, 2

[31] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in neural information processing systems*, 29, 2016. 6

[32] David Hume. *A Treatise of Human Nature*. Oxford University Press, 1978. revised P.H. Nidditch. 8

[33] Marcus Hutter et al. Algorithmic complexity. 2008. 4

[34] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 9

[35] Fahdi Kanavati and Masayuki Tsuneki. Partial transfusion: on the expressive influence of trainable batch norm parameters for transfer learning. In *Medical Imaging with Deep Learning*, pages 338–353. PMLR, 2021. 6

[36] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 2

[37] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 19, 24

[38] John Langford. *Quantitatively tight sample complexity bounds*. PhD thesis, Carnegie Mellon University, 2002. 2

[39] John Langford and Rich Caruana. (not) bounding the true error. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 809–816. MIT Press, 2001. 2

[40] John Langford and Matthias Seeger. Bounds for Averaging Classifiers. *Tech. rep CMU-CS-01-102, Carnegie Mellon University*, 2001. 2, 4

[41] Hugo Larochelle, Dumitru Erhan, Aaron C. Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 473–480. ACM, 2007. 21

[42] Quoc V. Le, Tamás Sarlós, and Alex Smola. Fastfood: Approximate kernel expansions in loglinear time. *ArXiv*, abs/1408.3060, 2013. 5

[43] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324, 1998. 19, 24

[44] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 4, 5

[45] Ping Li, Trevor J. Hastie, and Kenneth Ward Church. Very sparse random projections. In *KDD '06*, 2006. 5

[46] Zhiyuan Li, Sadhika Malladi, and Sanjeev Arora. On the Validity of Modeling SGD with Stochastic Differential Equations (SDEs). *Preprint arXiv:2102.12470*, 2021. 9

[47] David JC MacKay, David JC Mac Kay, et al. *Information theory, inference and learning algorithms*. Cambridge university press, 2003. 6

[48] Andreas Maurer. A Note on the PAC Bayesian Theorem. *Preprint arXiv 041099v1*, 2004. 4

[49] David A McAllester. PAC-Bayesian Model Averaging. *Proceedings of the 12th Annual Conference on Learning Theory (COLT)*, pages 164–170, 1999. 4

[50] Vaishnavh Nagarajan and J Zico Kolter. Deterministic pac-bayesian generalization bounds for deep networks via generalizing noise-resilience. *arXiv preprint arXiv:1905.13344*, 2019. 10

[51] Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[52] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 10

[53] Jeffrey Negrea, Gintare Karolina Dziugaite, and Daniel Roy. In defense of uniform convergence: Generalization via derandomization with an application to interpolating predictors. In *International Conference on Machine Learning*, pages 7263–7272. PMLR, 2020. 1

[54] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 19, 24

[55] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 1, 2

[56] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial*

Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 3942–3951. AAAI Press, 2018. 6

[57] María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepersvári. Tighter Risk Certificates for Neural Networks. *Journal of Machine Learning Research 22 (2021) 1-40*, 2021. 3, 7, 17

[58] Omar Rivasplata, Vikram M Tankasali, and Csaba Szepesvári. Pac-bayes with backprop. *arXiv preprint arXiv:1908.07380*, 2019. 2, 3

[59] Samuel L. Smith, Erich Elsen, and Soham De. On the generalization benefit of noise in stochastic gradient descent. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9058–9067. PMLR, 2020. 9

[60] Ray J Solomonoff. A formal theory of inductive inference. part i. *Information and control*, 7 (1):1–22, 1964. 4

[61] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 19

[62] Niklas Thiemann, Christian Igel, Oliver Wintenberger, and Yevgeny Seldin. A Strongly Quasiconvex PAC-Bayesian Bound. *28th Annual Conference on Learning Theory (COLT)*, 2017. 3, 4

[63] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018. 9

[64] Maurice Weiler and Gabriele Cesa. General e(2)-equivariant steerable cnns. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14334–14345, 2019. 9, 21

[65] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997. 8

[66] Jingfeng Wu, Wenqing Hu, Haoyi Xiong, Jun Huan, Vladimir Braverman, and Zhanxing Zhu. On the noisy gradient descent that generalizes as SGD. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 10367–10376. PMLR, 2020. 9

[67] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. 19, 24

[68] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley J. Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 6

[69] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*. BMVA Press, 2016. 9

[70] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64 (3):107–115, 2021. 1

[71] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1, 2, 3, 4, 5, 6, 23, 24

[72] Sicheng Zhu, Bang An, and Furong Huang. Understanding the generalization benefit of model invariance from a data perspective. *Advances in Neural Information Processing Systems*, 34, 2021. 9

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We provide evidence for the claims made in the abstract and introduction in Section 4, Section 5.1, and Section 6.
   (b) Did you describe the limitations of your work? [Yes] We describe limitations in Section 7.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Appendix J.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix J.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix E.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Appendix.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix E.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] See Appendix K.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Our code is submitted in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Supplementary Material for PAC-Bayes Compression Bounds So Tight That They Can Explain Generalization

Appendix Outline

The appendix is organized as follows.

- In Appendix B, we report results for additional bounds for SVHN and ImageNet. We also report the compression size corresponding to our best bound values and compare it to the compression size obtained through standard pruning. Furthermore, in Appendix B.1 we prove why models cannot both be compressible and fit random labels.
- In Appendix C, we show how our PAC-Bayes bound benefit from transfer learning.
- In Appendix D, we discuss data-dependent priors and their effect on the subspace dimension optimization.
- In Appendix E, we detail our experimental setup including models, datasets, and hyperparameter settings for training and bound computation.
- In Appendix F, we provide a compression perspective to why equivariant models may be more desirable for generalization.
- In Appendix G, we further discuss how our through our PAC-Bayes compression bounds, we provide evidence that SGD is not necessary for generalization.
- In Appendix H, we ablate the model size and show how it impacts our bounds and compressibility, we identify the best performing size of models for our bounds.
- In Appendix I, we present our observations on double descent and their preditability from our PAC-Bayes bounds.
- In Appendix J, we expand our theoretical discussion and emphasize conceptual differences between our method and previous ones in the literature.
- Lastly, in Appendix K we provide licensing information on the datasets we use.

A Subspace Dimension Optimization and Hyperparameters in the Universal Prior

The smaller the chosen intrinsic dimension d, the more similar θ is to the initialization θ_0 in Eq. (3). Consequently, that value of θ is more likely under the universal prior given the shorter description length. Note that in this prior, we condition on the random seed used to generate θ_0 and P. As we optimize over different parameters such as the subspace dimension $d = 1, .., D$, and possibly other hyperparameters such as the learning rate, or number of quantization levels L, we must encode these into our prior and thus pay a penalty for optimizing over them. We can accomodate this very simply by considering the hypothesis h as not just specifying the weights, but also specifying these hyperparameters: $h = (\theta, d, L, \mathrm{lr})$, and therefore using the universal prior $P(h) = 2^{-K(h)}/Z$ we pay additional bits for each of these quantities: $K(h) \leq K(\theta \mid d, L) + K(d) + K(L) + K(\mathrm{lr})$. If we optimize over a fixed number H of distinct values known in advance for a given hyperparameter such as L, then we can code L using this information in only $\log_2(H)$ bits. In general, we can also bound the dimensionalities searched over by the maximum D so that $K(d) \leq \lceil \log_2 D \rceil$ in any case.

For transfer learning, we replace θ_0 with a learned initialization $\theta_{\mathcal{D}}$ that is found using the pretraining task and data $\mathcal{D}$. With the ID compression, the universal prior $P(h \mid \mathcal{D}) \propto 2^{-K(h|\mathcal{D})}$ will place higher likelihood on solutions θ that are close to the pre-training solution $\theta_{\mathcal{D}}$.

B Additional Results

In addition to the results reported in Table 2, we report the best bounds for SVHN and ImageNet-1k as well as the corresponding compressed size in Tables 3 and 4. In Table 3 we show how compressing the model via intrinsic dimension (ID) yields better results than standard pruning. In this table, we basically run our method but substitute ID with pruning and then proceed by quantizing the remaining weights and encoding them through arithmetic encoding. When pruning we used the standard iterative procedure following Han et al. [27], for the MNIST model we pruned 98.8% of the weights, for the FMNIST model 97.0% of the weights, for the SVHN model 98.8% of the weights and for both the CIFAR-10 and CIFAR-100 models we pruned 52.1% of the weights and stopped there as the accuracy dropped significantly if we kept pruning.

Error bars on our bounds: We re-run the bounds computation for 10 times and observe that the values are consistent. On average, we obtain $\pm 0.5\%$ variation in our bounds for models trained from scratch and $\pm 0.1\%$ variation for transfer learning models.

Table 3: Using our subspace method rather than pruning yields substantially higher compression ratios and hence tighter generalization bounds. We report our error bounds (%) and compressed size ($\mathbb{KL}$ (KB)), 1 KB = 8192 bits. First, we compress the model weights using ID, quantizing its values and then storing them through arithmetic encoding. We then report the bounds obtained by only switching ID to standard pruning. All results are data-independent and obtained with 95% confidence, i.e. $\delta = .05$.

Dataset	ID + Quant + Arith		Pruning + Quant + Arith	
	Err. Bound (%)	$\mathbb{KL}$ (KB)	Err. Bound (%)	$\mathbb{KL}$ (KB)
MNIST	**11.6**	0.4	47.9	6.5
+ SVHN Transfer	**9.0**	0.4		
FashionMNIST	**32.8**	0.8	54.9	3.5
+ CIFAR-10 Transfer	**28.2**	0.9		
SVHN	**36.1**	1.3	74.4	4.3
+ ImageNet Transfer	**29.1**	1.4		
CIFAR-10	**58.2**	1.2	100.0	57.8
+ ImageNet Transfer	**35.1**	1.0		
CIFAR-100	**94.6**	4.1	99.9	50.7
+ ImageNet Transfer	**81.3**	2.8		

Table 4: Our PAC-Bayesian Subspace Compression Bounds with data-dependent priors compared to state-of-the-art PAC-Bayes non-vacous data-dependent bounds. All results are obtained with 95% confidence, i.e. $\delta = .05$.

Dataset	Err. Bound (%)	SoTA (%)
MNIST	**1.4**	1.5 [57]
FashionMNIST	**10.1**	38 [19]
SVHN	**8.7**	–
CIFAR-10	**16.6**	16.7 [57]
CIFAR-100	**44.4**	–
ImageNet	**40.9**	–

B.1 Models that can fit random labels cannot be compressed

Our ability to construct nonvacuous generalization bounds rests on the ability to construct models which both fit the training data and are highly compressible. However, when the structure in the dataset has been completely destroyed by shuffling the labels, then we do not find that our models are compressible (shown in Figure 3 right). This is not just an empirical fact, but one that can be proven apriori: models which fit random labels cannot be compressed. While this result is a trivial consequence of complexity theory, we present an argument here for illustration.

Almost all random datasets are incompressible

When sampling labels uniformly at random, almost all datasets are not substantially compressible. Given a dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$ (where we are only considering the labels y_i, and conditioning on the inputs x_i), and denoting $|\mathcal{D}|$ as the length of the string of labels, the probability that a given

dataset can be compressed to size $|\mathcal{D}|-c$ is less than 2^{-c+1}. To see this, one must consider that there are only $\sum_{i=0}^{|\mathcal{D}|-c} 2^i \leq 2^{|\mathcal{D}|-c+1}$ programs of length $\leq |\mathcal{D}| - c$ (fewer still when restricting to self delimiting programs), and there are $2^{|\mathcal{D}|}$ possible datasets. Therefore averaging over all randomly labeled datasets the fraction which are compressible to less than or equal to $|\mathcal{D}| - c$ bits is at most $2^{|\mathcal{D}|-c+1}/2^{|\mathcal{D}|} = 2^{-c+1}$.

A compressible model which fits the data is a compression of the dataset

Let prior P that includes a specification of the model architecture, and the model h which outputs probabilities for each of the outcomes: $p(y = k \mid x_i) = h(x_i)_k$. We can decompose the (prefix) Kolmogorov complexity of the dataset (given the prior) as

$$K(\mathcal{D} \mid P) \leq K(\mathcal{D} \mid h, P) + K(h \mid P). \tag{5}$$

The term $K(\mathcal{D} \mid h, P)$ can be interpreted as a model fit term and upper bounded by the total negative log likelihood simply using the model probabilities as a distribution to encode the labels: $K(\mathcal{D} \mid h, P) \leq -\sum_i \log_2 h(x_i)_{y_i} + 1 = \mathrm{NLL}(\mathcal{D} \mid h) + 1$.

Using the fact that almost all random datasets are incompressible, and choosing $c = 1 + \log_2(1/\delta)$, we have that with probability at least $1 - \delta$ over all randomly sampled datasets $K(\mathcal{D} \mid P) > |\mathcal{D}| - \log_2(1/\delta) - 1$. Plugging into Eq. (5) and rearranging, we have with probability $1 - \delta$,

$$K(h|P) \geq |\mathcal{D}| - \mathrm{NLL}(\mathcal{D} \mid h) - \log_2(1/\delta) - 2, \tag{6}$$

In Figure 6 we plot the quantity $K(h \mid P) + \mathrm{NLL}(\mathcal{D} \mid h)$ which represents the compressed size of the dataset achieved by our model (related to the minimum description length principle). We see that the value is considerably lower than the size of the dataset $|\mathcal{D}|$, emphasizing that real machine learning datasets such as CIFAR-10 have a very low Kolmogorov complexity and are very unlike those with random labels.

C Transfer Learning Bounds

We show the expanded results both with and without transfer learning in Table 3. When finetuning from ImageNet we use the larger EfficientNet-B0 models rather than the small convnet. Despite the fact that the model is significantly larger than the convnet or resnet models that we use to achieve the best bounds for from scratch training, the difference between the finetuned and pretrained models is highly compressible.

D Data Dependent Priors

We observe that when using data dependent priors, our optimization over the subspace dimension (and the complexity of the model used to fit the data when measured against the prior) favors very low dimensions and low KL values which we show empirically in Figure 4. This phenomenon makes sense since a large fraction of the data fitting is moved into fitting a good prior, and particularly when the dataset fraction used to train the prior is large. When the prior is already fitted on the data, the final solution can have a very low complexity with respect to that prior and is encouraged to do so.

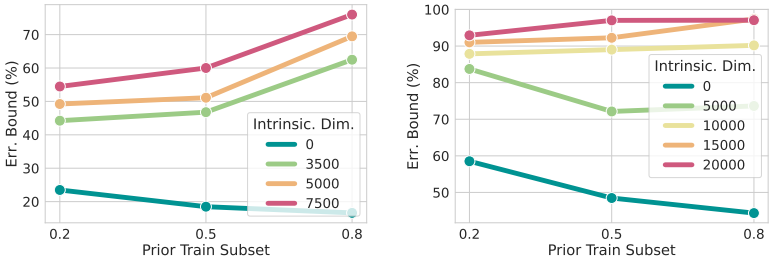
E Experimental Details

In this section we provide experimental details to reproduce our results.

E.1 Model Training Details

We use a standard small convolutional architecture for our experiments, which we find produces better bounds than its ResNet counterparts. The architecture is detailed in Table 5, and we use $k = 16$ for experiments, but this value is ablated in Figure 6.

Stochastic training: All models were trained for 500 epochs using the Adam optimizer with learning rate 0.001, except for ImageNet which was trained for 80 epochs with SGD using learning rate of 0.05 and weight decay of 0.00002. The model architectures for each dataset are listed below:



(a) CIFAR-10 (b) CIFAR-100

Figure 4: Our bounds using data dependent priors trained using varying fractions of the training dataset. We see that when using data dependent priors, lower intrinsic dimensionalities and lower KL models are favored by the bound.

- MNIST [43] (+ SVHN [54] Transfer): LeNet-5 [43].
- FashionMNIST [67] (+ CIFAR-10 [37] Transfer): ResNet20 [28].
- SVHN: ConvNet (Table 5).
- SVHN + ImageNet Transfer: EfficientNet-B0 [61].
- CIFAR-10: ConvNet.
- CIFAR-10 + ImageNet Transfer: EfficientNet-B0 [61].
- CIFAR-100 [37]: ConvNet.
- CIFAR-100 + ImageNet [15] Transfer: EfficientNet-B0.

Table 5: Simple convolutional architecture we use to compute our bounds.

ConvNet Architecture
Conv(3,k), BN, ReLU
Conv(k,k), BN, ReLU
Conv(k,2k), BN, ReLU
MaxPool2d(2)
Conv(2k,2k), BN, ReLU
Conv(2k,2k), BN, ReLU
Conv(2k,2k), BN, ReLU
MaxPool2d(2)
Conv(2k,2k), BN, ReLU
Conv(2k,2k), BN, ReLU
Conv(2k,2k), BN, ReLU
GlobalAveragePool2d
Linear(2k,C)

Full-batch training: We train all models for 3000 epochs, use learning rates equal to 0.1 (MNIST + LeNet-5 and CIFAR-10 + ResNet-18) and 0.5 (CIFAR-10 + ConvNet), and a cosine learning rate scheduler that we warm-up for 10 epochs. We also clip the full gradient to have an L_2-norm of at most 0.25 before performing parameter updates in each epoch [24].

Transfer Learning All previous training details remain the same, except that θ_0 from Eq. (3) is initialized from a pre-trained checkpoint instead of a random initialization. As typically done in literature, the final classification layer is replaced with a randomly initialized fully-connected layer to account for the number of classes in the downstream task.

E.2 Bound Hyperparameter Optimization

As explained in Appendix A, we optimize the bound hyperparameters by considering that the hypothesis of interest h specifies the hyperparameters in addition to the weights. Therefore, we pay

bits back for the combination of hyperparemeters that we select. For example, if we are doing a grid search over 2 values of the quantization-aware training learning rate, 2 values of the intrinsic dimensionality values, 2 values of the quantization levels, and use k-means by default, then the number of bits that we pay is $\log_2(2 \times 2 \times 2) = 3$ bits.

Optimizing PAC-Bayes bounds for data-independent priors: Our PAC-Bayesian subspace compression bounds for data-independent priors have 4 hyperparameters that we list here-under alongside the possible values that we consider for each hyperparameter:

- The learning rate for the quantization-aware training, possible values: $\{0.001, 0.003, 0.005, 0.0001\}$.
- The intrinsic dimensionality, possible values: $\{0, 1000, 2500, 3000, 3500, 4000, 5000, 7500, 8000, 10000, 12000, 15000, 20000, 25000, 50000, 100000, 250000, 500000\}$, except for the ImageNet transfer learning which was conducted on the more limited range: $\{500, 1000, 2000, 3000, 4000, 6000, 8000\}$
- The number of quantization levels, possible values: $\{0, 7, 11, 30, 50\}$.
- The quantization initialization, possible values: $\{\text{uniform}, \text{k-means}\}$.

Note that we only use a subset of these hyperparameter values for some datasets, depending on the dataset size and other considerations. For all bound computations, we use arithmetic encoding and 30 epochs of quantization-aware training.

In Table 6, we summarize the hyperparameters corresponding to the data-independent bounds that we report in Table 2.

Table 6: Hyperparameters corresponding to our PAC-Bayesian Subspace Compression Bounds reported in Table 2 as well as SVHN and ImageNet to SVHN transfer learning with **data-independent priors**. All bound results are obtained with 95% confidence, i.e. $\delta = .05$.

	Err. Bound (%)	Quant. Learning Rate	Intrinsic Dimensionality	Levels	Quant. Init.
MNIST	**11.6**	0.005	1000	7	Uniform
+ SVHN Transfer	**9.0**	0.005	1000	7	Uniform
FashionMNIST	**32.8**	0.005	2500	7	Uniform
+ CIFAR-10 Transfer	**28.2**	0.005	2500	7	Uniform
SVHN	**36.1**	0.0001	3500	11	Uniform
+ ImageNet Transfer	**29.1**	0.003	4000	7	Uniform
CIFAR-10	**58.2**	0.0001	3500	7	k-Means
+ ImageNet Transfer	**35.1**	0.003	3000	7	Uniform
CIFAR-100	**94.6**	0.0001	10000	11	k-Means
+ ImageNet Transfer	**81.3**	0.003	8000	7	Uniform

Optimizing PAC-Bayes bounds for data-dependent priors: In addition to the hyperparameters listed above, we also tune the hyperparameter corresponding to the subset of the training dataset that we use to train the prior on. We consider the following values for the subset of the training dataset: $\{20\%, 50\%, 80\%\}$.

In Table 7, we summarize the the hyperparameters corresponding to the data-dependent bounds that we report in Table 2. The best bounds are obtained for intrinsic dimensionality equal to 0, therefore no quantization is performed.

E.3 Computational Infrastructure & Resources

Our computational hardware involved a mix of NVIDIA GeForce RTX 2080 Ti (12GB), NVIDIA TITAN RTX (24GB), NVIDIA V100 (32GB), and NVIDIA RTX8000 (48GB). The experiments were managed via W&B [5]. The total computational cost of all experiments (including the ones that do not appear in this work) amounts to ≈ 8000 GPU hours.

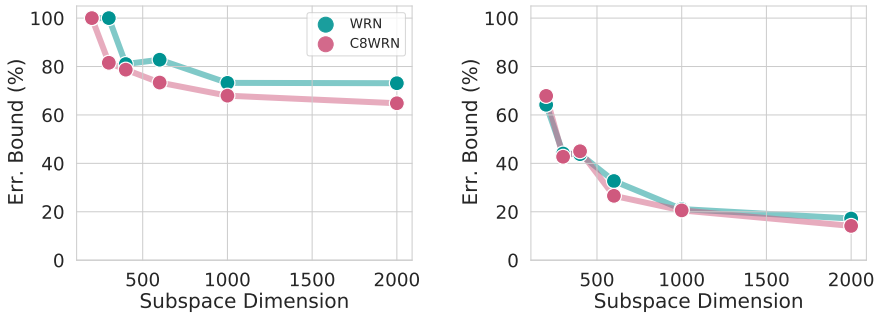
F Equivariance

We conduct a simple experiment to evaluate the extent to which model equivariance has on the compressibility of deep learning models and the tightness of our generalization bounds. We use

Table 7: Hyperparameters corresponding to our PAC-Bayes bounds reported in Table 2 as well as SVHN and ImageNet with **data-dependent priors**. The best bounds are obtained for intrinsic dimensionality equal to 0, therefore no quantization is performed. All bound results are obtained with 95% confidence, i.e. $\delta = .05$.

	Err. Bound (%)	Training Subset (%)
MNIST	**1.4**	50
FashionMNIST	**10.1**	80
SVHN	**8.7**	50
CIFAR-10	**16.6**	80
CIFAR-100	**44.4**	80
ImageNet	**40.9**	50

the rotationally equivariant C_8 WideResNet model from Weiler and Cesa [64] which has an 8-fold rotational symmetry, and we also use a non equivariant version of this model. The equivariant model has a depth of 10 and a widen factor of 4 yielding 1.451M parameters. We control for the number of parameters by adjusting the widen factor of the non equivariant model to 4.67 yielding 1.447M parameters.



(a) RotMNIST (12k labels) (b) MNIST (60k labels)

Figure 5: Comparison of rotationally equivariant C_8 WideResNet vs ordinary WideResNet with the same number of parameters on (a) the rotationally equivariant RotMNIST dataset [41] and (b) the ordinary MNIST dataset. Both models are capable of fitting the data, but the equivariant model yields a more compressible solution when fitting the rotationally equivariant data than the non equivariant model, and hence yields a better generalization bound. (Note the difference in dataset size, RotMNIST has only 12K data points unlike MNIST)

We evaluate these models both on MNIST and the RotMNIST dataset [41] consisting of 12K training examples of rotated MNIST digits. As shown in Figure 5 (a), when paired with the rotationally symmetric RotMNIST dataset, the rotationally equivariant model achieves better bounds and is more compressible than it's non equivariant counterpart despite having the same number of parameters. However, when this symmetry of the dataset is removed by considering standard MNIST, we see that the benefits of equivariance to the generalization bound and compressibility vs the WRN model dissapear.

G Full-Batch vs. Stochastic Training (SGD)

To further expand on the results that we present in Section 6, we study the impact of hyperparameters, namely the weight decay and the architecture choice, on the bounds obtained through full-batch (F-B) training. Table 8 summarizes these results and we provide the training and bound computation details in Appendix E. Our PAC-Bayes subspace compression bounds provide similar theoretical guarantees for both full-batch and stochastic training, suggesting that the implicit biases of SGD are not necessary to guarantee good generalization. Moreover, we see that the results are consistent for different configurations, which result in comparable bounds overall.

Transfer learning using full-batch training: We perform full batch training for transfer learning from SVHN to MNIST using LeNet-5 and the same experimental setup described in Appendix E.

Our best PAC-Bayes subspace compression bounds for SVHN to MNIST transfer are 8.7% and 9.0% for full-batch and SGD training, respectively. This finding provides further evidence that good generalization of neural networks, and the success of transfer learning in particular, does not necessarily require stochasticity or additional flatness-inducing procedures to be achieved.

Finally, we note that we optimize over the same set of hyperparameters for the bound computation for both full-batch and stochastic training.

Table 8: Our PAC-Bayes subspace compression bounds obtained through full-batch (F-B) training for different configurations and datasets.

Dataset	Architecture	Stochastic Err. Bound (%)	F-B Weight Decay	F-B Err. Bound (%)
MNIST	LeNet-5	11.6	0.01	12.5
			0.001	**11.2**
			0.005	12.0
			0.0001	11.7
CIFAR-10	ResNet-18	74.7	0.01	77.8
			0.001	76.3
			0.005	76.1
			0.0001	**75.3**
CIFAR-10	ConvNet	58.2	0.01	65.8
			0.001	63.6
			0.0001	**61.4**

H Model Size vs. Compressibility

We perform an ablation to determine how the size of the model affects our generalization bounds. Using the fixed model architecture Table 5, we vary the width k from 4 to 192. Using our subspace compression scheme, we find that the compression ratio of the model does increase with model size, however the total compressed size still increases slowly making our bounds less strong for larger models. For this paper, we find the sweet spot $k = 16$ is just above the point with equal number of parameters and data points.

We note that this finding leaves room for an improved compression scheme and generalization bounds which are able to explain why even larger models still generalize better. Curiously, when plotting to the total compressed dataset size ($K(h|P) + \text{NLL}$) using the model as a compression scheme, we find that the MDL principle which favors shorter description lengths of the data actually prefers larger models than our PAC-Bayes generalization bounds selects.

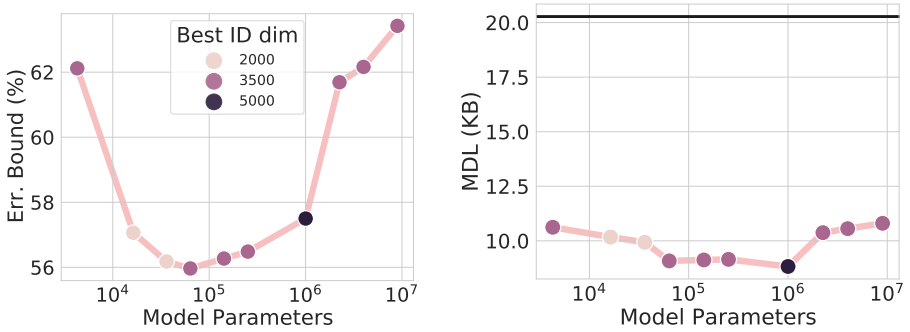


Figure 6: **Left:** Generalization error bound as a function of model size on the CIFAR10 dataset. The ID subspace dimension that achieves the best bound is shown by the color. In terms of our bound computation, the optimal number of parameters of the network is only slightly above the number of data points. **Right:** The total compressed size ($K(h|P) + \text{NLL}$) of the training dataset using our model as a compression scheme. While the raw labels have size 20.3KB (shown by the black line), the best model compresses the labels down to 8.6KB. Curiously, the compressed dataset size and hence the MDL principle favors larger models than our generalization bounds. [S L]

I Double Descent

Under select conditions, we are able to reproduce the double descent phenomenon in our generalization bounds. In Figure 7 (right), we show that our bound exhibits a double descent similar to what we see in terms of the test error Figure 7 (left). The results we show in Figure 7 (right) are obtained for a fixed intrinsic dimensionality of 35000, but we observed that this middle descent consistently appears in our bounds plots for a given (fixed) intrinsic dimensionality where we select the best bound for each base width. However, we expect that extending the plot out to larger model widths the bound gets worse again as explained in Appendix H.

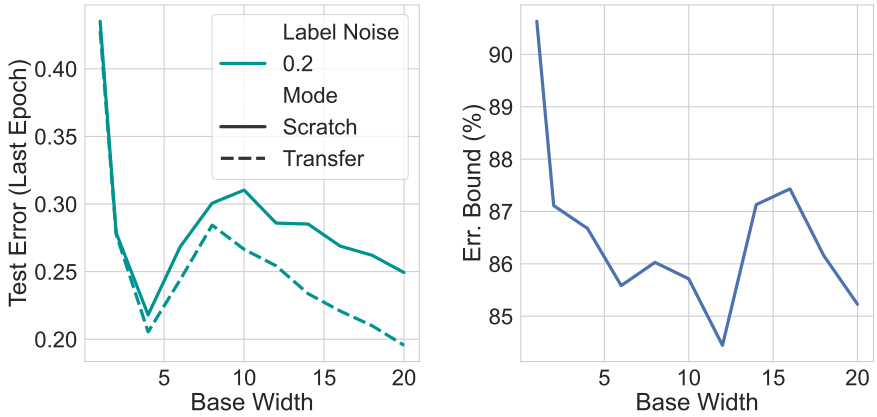


Figure 7: **Left:** Double descent (in terms of the test error of the last epoch model) observed when varying the width of a ResNets-18 model to fit the CIFAR-10 dataset with label noise equal to 0.2. **Right:** Our bounds showing a similar *double descent* behaviour where the bound starts to worsen only to become better again at a later width. Here we can fix the intrinsic dimensionality to be equal to 35000 and we choose the best subspace compression bound for each base width.

J PAC-Bayes Bounds

J.1 Catoni PAC-Bayes Bound

In our case, since neural networks achieve low training error, we focus on a bound like Catoni [7] which becomes tighter when $\mathbb{KL}(Q, P)$ is large. This is the same bound used in Zhou et al. [71].

Theorem 1 (Catoni [7])**.** *Given a* 0-1 *loss* ℓ*, a fixed* $\alpha > 1$ *and a confidence level* $\delta \in (0, 1)$ *then*

$$\underset{\theta \sim Q}{\mathbb{E}}[R(f_\theta)] \leq \inf_{\lambda > 1} \Phi^{-1}_{\lambda/N}\left[\underset{\theta \sim Q}{\mathbb{E}}[\hat{R}(f_\theta)] + \frac{\alpha}{\lambda}\left[\mathbb{KL}(Q, P) + \log\frac{1}{\delta} + 2\log\left(\frac{\log(\alpha^2\lambda)}{\log\alpha}\right)\right]\right]$$

holds with probability higher than $1 - \delta$ *and where*

$$\Phi^{-1}_{\gamma}(x) = \frac{1 - e^{\gamma x}}{1 - e^{\gamma}}.$$

J.2 Variable Length Encoding and Robustness Adjustment

In Zhou et al. [71], the authors assume a fixed length encoding for the weights. Given that the distribution over quantization levels is highly nonuniform, using a variable length encoding (such as Huffman encoding or arithmetic encoding) can represent the same information using fewer bits. While this choice gives significant benefits, it means that we cannot immediately make use of robustness adjustment from Zhou et al. [71], where the robustness adjustment comes from considering neighboring models that result from perturbing the weights slightly.

Revisiting the prior derivation in Zhou et al. [71], we show why the method used for bounding the KL does not transfer over to variable length encodings. In Zhou et al. [71], the prior used is

$$P = \frac{1}{Z} \sum_{S,Q,C} 2^{-(|S|+|C|+d\lceil \log L \rceil)} \mathcal{N}\left(\hat{w}\left(S,Q,C\right), \tau^2\right)$$

where S denotes the encoding of the position of the pruned weights, C denotes the codebook, Q the codebook value that the weight take, d the number of nonzero weights, L the number of clusters, $\hat{w}$ the quantized weight and τ^2 the prior variance. Note that $\hat{w}$ changes depending on S, Q, C and also note that the fixed-length encoding can be seen in how we sum over $d\lceil \log L \rceil$ options. This prior is a mixture of Gaussians centered at the quantized values. With this choice of prior Zhou et al. [71] and setting the posterior to be also Gaussian centered at a quantized value, one can upper bound the KL with a computationally tractable term involving the sum over dimensions. Crucially, for their decomposition they use the fact that the size of the encoding $|Q|$ is $d\lceil \log L \rceil$, which is independent of the coding Q and only dependent on the codebook C. Therefore they are able to upper bound the KL.

$$\mathbb{KL}\left(\mathcal{N}\left(\hat{w}, \sigma^2 I_d\right), \sum_{Q} \mathcal{N}(\hat{w}(\hat{S}, Q, \hat{C}), \tau^2)\right) = \sum_{i=1}^{d} \mathbb{KL}\left(\mathcal{N}\left(\hat{w}_i, \sigma^2\right), \sum_{j=1}^{L} \mathcal{N}\left(\hat{w}_j, \tau^2\right)\right)$$

due to the independence of the fixed length encoding to that of the values that each quantized value takes, see appendix in Zhou et al. [71]. This independence is broken for variable length encoding as the cluster centers and the values that each weight can take are interlinked. Thus, we cannot express and satisfactorily approximate the first high-dimensional KL term as a sum of one-dimensional elements that can be estimated through quadrature or Monte Carlo.

K Licences

MNIST [4] [43] is made available under the terms of the Creative Commons Attribution-Share Alike 3.0 license. FashionMNIST [5] [67] is available under the MIT license. CIFAR-10 and CIFAR-100 [37] are made available under the MIT license. ImageNet [6] [15] is the copyright of Stanford Vision Lab, Stanford University, Princeton University. SVHN [7] [54] is released for non-commericial use only.

[4] http://yann.lecun.com/exdb/mnist/
[5] https://github.com/zalandoresearch/fashion-mnist
[6] https://www.image-net.org/download
[7] http://ufldl.stanford.edu/housenumbers/