Quantization and the Bottom of the Loss Landscape

Luca Di Carlo♢ LUCADC@PRINCETON.EDU
Joseph Henry Laboratories of Physics, Princeton University, Princeton, NJ, USA
Lewis-Sigler Institute, Princeton University, Princeton, NJ, USA
Center for the Physics of Biological Function, Princeton University, Princeton, NJ, USA

Daniel T. Bernstein♢ DB8682@PRINCETON.EDU
Lewis-Sigler Institute, Princeton University, Princeton, NJ, USA
Center for the Physics of Biological Function, Princeton University, Princeton, NJ, USA

David Schwab DAVIDJSCHWAB@GMAIL.COM
Initiative for the Theoretical Sciences, The Graduate Center, CUNY, New York, NY, USA

♢ *Equal contribution*

Abstract

We introduce two physics-inspired methods for the compression of neural networks that encourage weight clustering, in anticipation of model quantization, by adding attractive interactions between parameters to the loss. Our two methods implement interactions either directly or via an intermediary set of centroids. By applying these methods to pre-trained neural networks, we investigate the existence of compressible configurations near the bottom of the loss landscape. The direct interaction approach suggests the existence of multiple, qualitatively distinct compressed configurations close to pre-trained models, and the centroid-mediated approach provides a pipeline for quantization that is competitive with extant quantization methods.

1. Introduction

The surprising effectiveness of relatively simple optimization algorithms [16] in training deep neural networks is often attributed to the accessibility and abundance of "good" minima in the loss landscape. A growing body of work suggests that, near the bottom of this landscape, most directions are relatively flat [18, 21, 23], and seemingly distinct minima can be connected by low-loss, non-linear paths [8]. Moreover, empirical studies indicate that many parameters are functionally redundant, although identifying which ones a priori remains challenging [9].

Model quantization aims to reduce the number of bits used to represent network weights, forcing them to take values from a small, discrete set. This is important for deploying large-scale models on resource-constrained hardware [6, 7]. The two most common quantization strategy paradigms are Post-Training Quantization (PTQ) [19] and Quantization-Aware Training (QAT) [2, 15, 26].

This work ties the above concepts together via the following question: given a pre–trained model, can we exploit the abundance of flat directions and redundant parameters near a minimum to direct the model towards a nearby, compressible solution—one that admits quantization with minimal loss in performance?

Guided by analogies with statistical physics, we introduce two conceptually related compression schemes. These schemes allow us to investigate whether pre-trained models can be steered via small local adjustments into quantizable regions of the loss landscape.

2. Related Work

The idea of combining the training loss with a clustering prior dates back to the work of Nowlan and Hinton [20], which motivated this prior from the perspective of regularization rather than compression. More recently, Ullrich et al. [24] applied this idea to model compression via "soft weight-sharing," where a mixture-of-Gaussians prior softly clusters the weights. Both approaches frame clustering as a probabilistic prior over the parameters. Related to this, several methods have incorporated explicit K-means clustering into the quantization process. Existing works [10, 25] optimize cluster centers after an initial K-means clustering step. Other works [4, 5] propose differentiable variants of K-means, using attention mechanisms to allow end-to-end training of both cluster assignments and centroids.

In parallel, a number of QAT techniques have been explored. Notable examples for our work include BinaryRelax [26], which explores relaxed binary representations, and PARQ [15], which revisits and improves the straight-through estimator commonly used in QAT. These methods train quantized models from scratch, often achieving strong performance even at low bit-widths, though typically at the cost of increased training time and complexity.

3. Methods

Given a target loss function $\mathcal{L}_{\text{task}}(\boldsymbol{\theta})$ defined over input-output pairs X, Y, we consider the following modification of the training objective:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{\text{task}}(\boldsymbol{\theta}; X, Y) + \mathcal{L}_{\text{C}}(\boldsymbol{\theta}; \mathbf{P}), \tag{1}$$

where the compression term $\mathcal{L}_{\text{C}}$ biases the optimization toward regions of parameter space where weights are organized into a small number of distinct clusters, thereby facilitating quantization. Minimizing $\mathcal{L}(\boldsymbol{\theta})$ corresponds to solving the original learning problem under an additional structural constraint favoring clustered solutions. In the following, we introduce two classes of compression losses, each implementing a different mechanism to steer the optimization toward clustered solutions.

3.1. Compression Via Pairwise Interactions

We consider a pairwise interaction potential between weights within the same layer:

$$\mathcal{L}_{\text{C}}(\boldsymbol{\theta}; \boldsymbol{h}, \boldsymbol{w}) = \sum_l h_l \sum_{i \neq j} U_{w_l}\left(\theta_i^{(l)} - \theta_j^{(l)}\right), \tag{2}$$

where $U_w(x)$ denotes an attractive potential defined by a triangular well of width w ($U_w(x) = |x|\theta(w^2 - x^2) + w\theta(x^2 - w^2)$, with $\theta(\cdot)$ the Heaviside step function). In other words, this potential looks like $|x|$ for $|x| < w$ and becomes flat for $|x| > w$. This potential induces attractive forces between weights which lie within a distance w of each other, thereby promoting local clustering of weights within each layer. Computing the interaction potential for a single layer requires $\mathcal{O}(N_l^2)$ operations, where N_l is the number of parameters in the l-th layer. At each training step we

compute the compression loss (2) by randomly subsampling a fraction $\mathcal{O}\left(\sqrt{N_l}\right)$ of weights per layer. This reduces the computational cost to $\mathcal{O}(N)$ overall, and additionally serves as a regularizer that discourages overly aggressive clustering. We refer to this approach as the weight–weight compression scheme. Unlike traditional quantization methods, we do not need to specify the bit width, nor do different layers need to take on the same bit-widths. We do not know a priori to what degree the model can be quantized, if at all. Instead of specifying a bit-width, we specify a strength and range for the local interactions between weights.

3.2. Compression Via Centroids

In this case, an effective attractive interaction between the weights is mediated by a set of K additional learnable parameters per layer, namely a set of centroids $C_\alpha^{(l)}$ with $\alpha = 1, \ldots, K$. In a way this approach is reminiscent of previously studied soft-weigh sharing methods [11, 24]. These centroids act as attractors that encourage weight clustering during training, as described by the following loss,

$$\mathcal{L}_{\mathrm{C}}(\boldsymbol{\theta}, \boldsymbol{C}) = \sum_l \sum_i \min_\alpha \left| C_\alpha^{(l)} - \theta_i^{(l)} \right|^\rho , \tag{3}$$

where ρ determines the shape of the potential. We refer to this approach as the centroids–weight compression scheme. The full training dynamics are then governed by stochastic gradient descent:

$$\delta C_\alpha^{(l)} = -\eta_2 \nabla_{C_\alpha^{(l)}} \mathcal{L}_{\mathrm{C}}(\boldsymbol{\theta}, \boldsymbol{C}), \qquad \delta \theta_i^{(l)} = -\eta_1 \nabla_{\theta_i^{(l)}} \mathcal{L}_{\mathrm{C}}(\boldsymbol{\theta}, \boldsymbol{C}) - \nabla_{\theta_i^{(l)}} \mathcal{L}_{\mathrm{task}}(\boldsymbol{\theta}), \tag{4}$$

In practice, gradients can be efficiently computed without relying on automatic differentiation tools. To evaluate the true effect of quantization, we assess the final model performance using **clamped parameters**, obtained by replacing each weight with the value of its nearest centroid. We allow for distinct learning rates η_1 and η_2. This asymmetry gives rise to a form of off-equilibrium dynamics, in which the centroids evolve on a potentially different timescale than the weights they influence.

4. Experiments

We conduct compression experiments using the two schemes described above, starting from pre–trained models. We apply both compression schemes to two widely used benchmarks: CIFAR-10 [17] with the Wide ResNet-16 (WRN-16) architecture [27], and ImageNet-1K [22] with the ResNet-50 (RN50) architecture [12].

4.1. Weight-Weight Compression

Our results suggest that weight-weight compression results in the formation of sharp clusters in the weight distribution. The number, positions, and sharpness of the clusters depend on the hyperparameters of the pairwise interactions: see Figure 1 (A). Moreover, even under fixed hyperparameters, different random seeds might result in different clustering, as shown in Figure 1 (B). Compared to the pre-trained accuracy, the ResNet-50 models highlighted in Figure 1 experience approximately 5% validation degradation on ImageNet-1K. These results suggest that more than one distinct clustering configuration is reachable from the same pre-trained configuration via weight-weight compression, and that stochastic dynamics contribute to which configuration the model approaches. This raises the prospect that, given a fuller understanding of how stochastic dynamics drive the model into

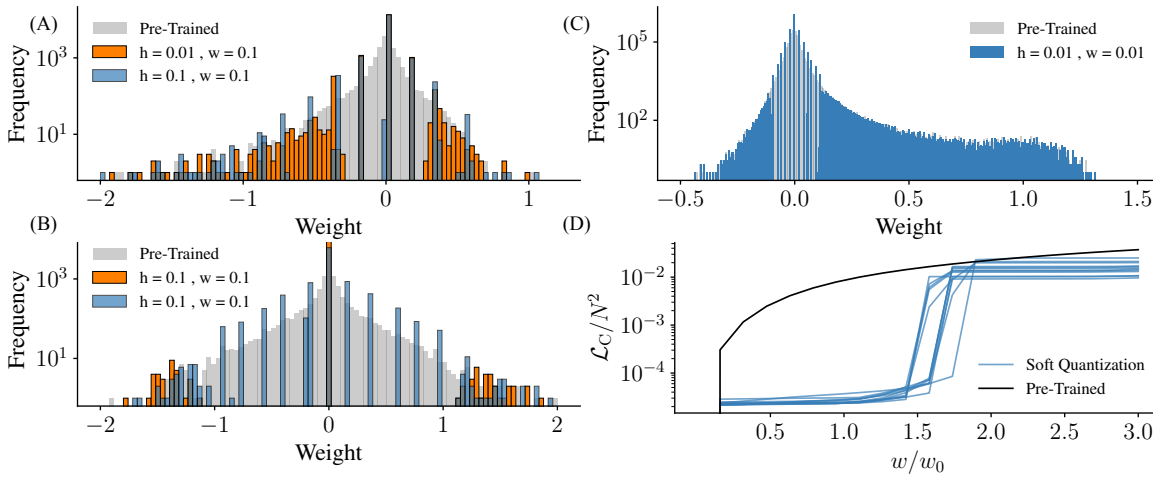

Figure 1: ResNet-50 on ImageNet-1K, weight distributions of convolutional kernels for the compressed and the pre–trained model for different weight-weight interaction strengths (A) and for different random seeds (B). Weight distribution for the fully connected output layer (C). The compression loss in a single layer as a function of the well width w (D) for the first layers of the compressed model (blue curves). As a reference we show the analogue (black) curve computed on the first layer of the pre-trained model.

qualitatively distinct yet similarly performant minima, it may be possible to steer a model into minima which are more amenable to explicit quantization.

We caution about directly relating this method to quantization. Retraining ResNet-50 with weight-weight compression yields models whose weight distributions are not perfectly clustered, with tails that often mimic the weight distribution of the pre-trained model. This is particularly evident in the last fully connected layer, see Figure 1 (C), which is known to be more challenging to quantize [1]. One hypothesis for why tails occur is a sampling problem: pairwise interactions are subsampled at each training step, and interactions between nearby weights at the distribution tails are much less likely to be sampled than interactions between nearby weights in the middle of the distribution.

Figure 1 (D) shows the single-layer compression loss (normalized by number of interactions) for a retrained model as a function of the interaction range. The compression loss does not spike until we reach an interaction range almost twice the value of the range upon which the model was retrained. Likewise, clusters form in a manner that is roughly symmetric around zero. Both the spacing and symmetry of clustering configurations suggest emergent structure in the weight distribution at a length scale larger than the interaction range itself.

4.2. Centroids-Weight Compression

We compare our method against several state-of-the-art quantization techniques, including QAT approaches and PTQ with histogram equalization [14]. The results are summarized in Table 1. Notably, on WRN-16 our method performs marginally better than PTQ. On the larger ResNet-50 model, it significantly outperforms PTQ. QAT methods achieve high accuracy even for low bit-width, but they required training from scratch [15]. At higher bit-widths, our method achieves competitive

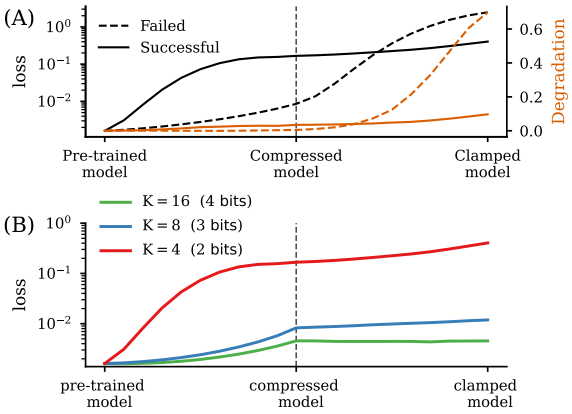

Figure 2: WRN-16 on CIFAR10: Task loss and performance degradation along linear paths. (A) Comparison between successful and failed compression experiments. (B) Comparison between compressions for different number of centroids.

	WRN-16	(92.73)	
Method	**2 bits**	**3 bits**	**4 bits**
Naive PTQ	73.91	84.17	90.74
CWC (ours)	72.92	90.93	92.41
	ResNet50	(75.50)	
Method	**2 bits**	**3 bits**	**4 bits**
Naive PTQ	4.87	28.17	49.38
CWC (ours)	50.78	73.50	75.55
PARQ	72.43	73.91	74.52
BinaryRelax	72.64	74.02	74.58

Table 1: Top-1 test accuracy (%) under different quantization schemes and bit-widths for CIFAR-10 (WRN-16) and ImageNet-1K (ResNet-50). Full-precision accuracy is shown in parentheses.

performance with substantially lower computational overhead, requiring only a few fine-tuning epochs starting from a pre-trained model, 10 epochs for WRN-16 and 3 epochs for ResNet50.

After compressing the model by following (4), we evaluate the task loss and validation accuracy along two linear interpolation paths: one from the pre-trained model to the compressed model, and another from the compressed model to the clamped model. In Figure 2(A), we show representative examples of both successful and failed compression experiments. In the failed case, the compressed model's loss remains close to that of the pre-trained model, and most of the performance degradation occurs during the clamping stage. This suggests that, in order to obtain a configuration whose parameters can be clamped/quantized without a significant loss in performance, the optimization process must sufficiently climb the energy landscape.

Figure 2 (B) illustrates this behavior for successful compression experiments, using the same interpolation scheme but across different numbers of centroids $K = 4, 8, 16$, corresponding to 2, 3, and 4 bits compression, respectively (results in Table 1). The emerging picture is that, near the bottom of the loss landscape, slightly increasing the loss—i.e., ascending the energy landscape—can provide access to compressible configurations, defined as parameter configurations whose performance is preserved under clamping or quantization. The energy barrier that must be overcome to reach such configurations decreases as the number of centroids increases.

5. Conclusion

We proposed two methods that encourage the formation of clustered weight configurations while maintaining proximity to pre-trained configurations via a compression-aware loss. Our results support the hypothesis that the abundance of flat directions and parameter redundancy in deep networks make it possible to find compressible solutions that do not lie far from the original minima. While the clustering induced by the compression losses is not always exact, one of the proposed methods—based on centroid-mediated interactions—naturally yields a quantized model by clamping each weight to its nearest centroid. This provides a direct path from soft clustering to hard quantization. Preliminary

experiments indicate that our approach achieves performance competitive with state-of-the-art QAT schemes.

References

[1] AmirAli Abdolrashidi, Lisa Wang, Shivani Agrawal, Jonathan Malmaud, Oleg Rybakov, Chas Leichner, and Lukasz Lew. Pareto-optimal quantized resnet is mostly 4-bit. 2021 ieee. In *CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3085–3093, 2021.

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

[4] Minsik Cho, Keivan Alizadeh-Vahid, Saurabh Adya, and Mohammad Rastegari. Dkm: Differentiable k-means clustering layer for neural network compression. In *International Conference on Learning Representations*, 2022.

[5] Minsik Cho, Keivan A Vahid, Qichen Fu, Saurabh Adya, Carlo C Del Mundo, Mohammad Rastegari, Devang Naik, and Peter Zatloukal. edkm: An efficient and accurate train-time weight clustering for large language models. *IEEE Computer Architecture Letters*, 23(1):37–40, 2024.

[6] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

[7] Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. *9th International Conference on Learning Representations, ICLR*, 2022.

[8] Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. Essentially no barriers in neural network energy landscape. In *International conference on machine learning*, pages 1309–1318. PMLR, 2018.

[9] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

[10] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*, 2016. URL http://arxiv.org/abs/1510.00149.

[11] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*, 2016. URL http://arxiv.org/abs/1510.00149.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2024. URL http://github.com/google/flax.

[14] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[15] Lisa Jin, Jianhao Ma, Zechun Liu, Andrey Gromov, Aaron Defazio, and Lin Xiao. PARQ: Piecewise-Affine Regularized Quantization, March 2025. URL http://arxiv.org/abs/2503.15748. arXiv:2503.15748 [cs].

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images.(2009), 2009.

[18] Zhenyu Liao and Michael W Mahoney. Hessian eigenspectra of more realistic nonlinear models. *Advances in Neural Information Processing Systems*, 34:20104–20117, 2021.

[19] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A White Paper on Neural Network Quantization, June 2021. URL http://arxiv.org/abs/2106.08295. arXiv:2106.08295 [cs].

[20] Steven J. Nowlan and Geoffrey E. Hinton. Simplifying neural networks by soft weight-sharing. *Neural Computation*, 4(4):473–493, 1992. doi: 10.1162/neco.1992.4.4.473.

[21] Vardan Papyan. The full spectrum of deepnet hessians at scale: Dynamics with sgd training and sample size. *arXiv preprint arXiv:1811.07062*, 2018.

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[23] Levent Sagun, Utku Evci, V. Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks, 2018. URL https://openreview.net/forum?id=rJrTwxbCb.

[24] Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. In *International Conference on Learning Representations*, 2017.

[25] Junru Wu, Yue Wang, Zhenyu Wu, Zhangyang Wang, Ashok Veeraraghavan, and Yingyan Lin. Deep k-means: Re-training and parameter sharing with harder cluster assignments for compressing deep convolutions. In *International Conference on Machine Learning*, pages 5363–5372. PMLR, 2018.

[26] Penghang Yin, Shuai Zhang, Jiancheng Lyu, Stanley Osher, Yingyong Qi, and Jack Xin. Binaryrelax: A relaxation approach for training deep neural networks with quantized weights. *SIAM Journal on Imaging Sciences*, 11(4):2205–2223, 2018. doi: 10.1137/18M1166134. URL https://doi.org/10.1137/18M1166134.

[27] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Procedings of the British Machine Vision Conference 2016*, pages 87–1. British Machine Vision Association, 2016.

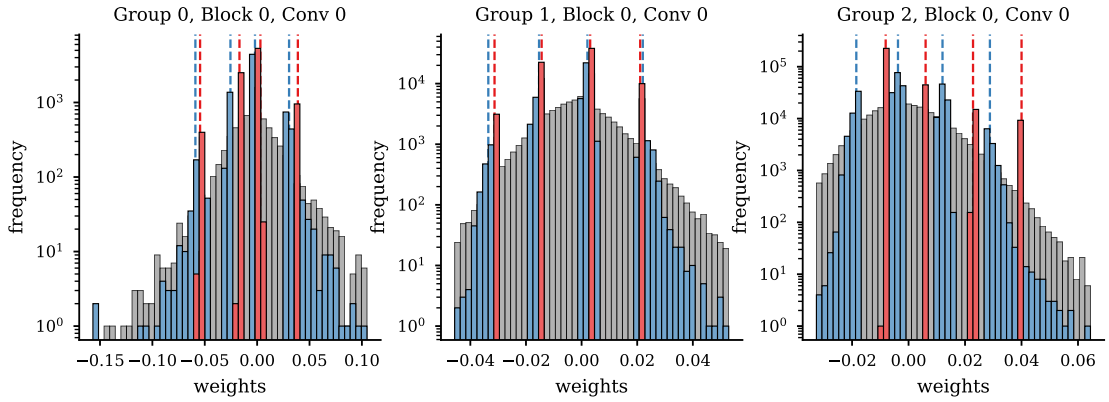

Figure 3: Weight distribution of WRN-16 convolutional layers for the pre–trained model (gray), a failed (blue) experiment and a successful (red) experiment, as described in Figure 2.

Appendix A. Details of the experiments

In this appendix we discuss the details of the experiments presented in the main text.

A.1. Weight-Weight Compression Experiments

We conducted compression experiments using the weight-weight interaction potential, defined in Equation (1), in PyTorch. We use Stochastic gradient descent with a batch size of 512, learning rate of 10^{-3}, and (Nesterov) momentum of 0.9 and no weight decay. We use data augmentation consisting of horizontal mirroring and random cropping, with the ratio between the cropped and original image areas randomly chosen from 0.08 to 1. We train all model parameters on the task, and subject all the parameters to the pairwise interaction excluding the biases and the BatchNorm layers. For each layer with N_l weights, we subsample N_l of the N_l^2 pairwise interactions at each training step.

To mitigate a prohibitively large parameter search in $\boldsymbol{h}$ and $\boldsymbol{w}$, we choose $h_l = h$, and we choose a rough scaling between the order of magnitude of w_l and the order of magnitude of weights in a layer: for the models corresponding to information in Figure 1, we choose $w_l = 0.1$ for all layers with less than 10^5 weights, and $w_l = 0.01$ for all layers with greater than 10^5 weights.

A.2. Centroid-Weight Compression Experiments

We conducted compression experiments using the centroid-weight interaction potential implemented in the JAX/Flax framework [3, 13]. On WRN-16, compression was performed for 10 epochs with a batch size of 256 using stochastic gradient descent (SGD) with momentum 0.9 and weight decay 5×10^{-4}. We have used data augmentation consisting of random 4-pixel crop-resize and horizontal mirroring. For ResNet-50, we used 3 epochs and a batch size of 256, also with SGD, momentum 0.9, and weight decay 5×10^{-4}. We used data augmentation consisting of random 4-pixel crop-resize and random rotation in a $[-\frac{\pi}{36}, +\frac{\pi}{36}]$ interval. In either cases we do not compress the BatchNorm layers.

In both settings, we randomly explored the space of hyperparameters including the exponent ρ defining the interaction potential of Eq. (3), the coupling strengths g_1 and g_2, and the learning rate

λ. We performed experiments across different numbers of centroids K. While our method does not require K to be a power of two, we chose $K = 4, 8, 16$—corresponding to 2, 3, and 4 bits of precision, respectively—for consistency with common benchmarks in the literature.

In Figure 3 we show the weight distribution for three different convolutional layers of WRN-16 on CIFAR10, for a failed (blue) and a successful (red) compression experiment. As shown in Figure 2, if the strength of the compression is not too strong, the weights are widely spread around the centroids and the performance of the model significantly drops when clamping the parameters to value of the closest centroid.