TROPICS: TROPICAL GEOMETRY FOR STRUCTURED NEURAL NETWORK COMPRESSION THROUGH ZONOTOPE APPROXIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Structured compression of neural networks remains challenging when high accuracy must be preserved under severe parameter reduction. We introduce **T**ropical **R**epresentation for **O**ptimized **P**runing, **I**nformed **C**lustering, and **S**caling (**TROPICS**), a data-free, non-uniform, and single-pass framework based on the tropical geometry of ReLU neural networks. We compress layers in two phases. First, neurons are mapped to zonotope generators and clustered using K-means, merging inputs by mean and outputs by sum, while scale factors recover the signal energy lost during clustering. Second, to avoid forcing dissimilar generators to merge, we magnitude-prune the weakest remaining units. TROPICS requires neither training data nor fine-tuning, and it substantially outperforms existing baselines on VGG16-BN (CIFAR-10): our scaled Mixed strategy achieves an 84.7% parameter reduction with only a 0.44% accuracy drop, while at 92.5% parameter reduction the scale factors preserve up to 29% more accuracy than unscaled variants. On ResNet18 (ImageNet), TROPICS surpasses strong methods such as TropNNC, CUP, and Model Folding at comparable compression.

1 INTRODUCTION

Neural networks are typically heavily over-parameterized, leading to significant redundancy at the level of neurons and convolutional filters. This redundancy has motivated a broad literature on compression, including pruning (Luo et al., 2017; Duggal et al., 2021), low-rank factorization, quantization, and knowledge distillation. In practice, however, compression is often constrained: training data may be private or proprietary, there may be no compute budget for retraining, or the model may need to be deployed at the edge. Most structured methods need data or retraining, and the data-free ones have no principled way to predict how much accuracy they will lose.

Moreover, these approaches mostly focus on algebraic structure such as norms, singular values, or weight statistics. Fewer methods exploit the underlying geometry of the functions the network computes. ReLU activations produce piecewise-linear functions whose input-output behaviour is described by polyhedral regions in activation space. Each layer can be interpreted as a zonotope formed by the Minkowski sum of line segments, where each neuron or filter contributes one generator. This geometric view connects compression directly to the structure of the zonotope: redundant neurons correspond to nearly parallel generators, and weak neurons correspond to short ones.

TROPICS uses this view to guide compression in two steps. First, a generator is constructed for each neuron or filter by multiplying the Euclidean norm of its output weights with its input weight vector. This generator approximates how the neuron contributes to the layer's output range. Then the generators are clustered with K-means and each cluster is merged into a single compressed neuron. Scale factors are applied to recover the energy lost during averaging. Intuitively, after aggressive clustering, merging dissimilar neurons or filters is more harmful to accuracy than pruning a weak one. Therefore, in the second step, pruning by magnitude is applied: each compressed neuron is scored by the sum of its input and output norms, and the weakest neurons are pruned. Finally, we establish a theoretical, data-free error bound and show that it correlates with the accuracy drop, providing a reliable estimator of performance degradation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Contributions:

- **Tropical Compression Framework:** We introduce TROPICS, a data-free, non-uniform, single-pass compression method. By leveraging the max-plus algebraic structure of ReLU networks, we compute generators that map the layer to a zonotope using output-norm-scaled weight vectors, requiring zero training data or optimization.
- **Two-Phase Pipeline & Energy Scaling:** We propose a hybrid pipeline of K-means clustering followed by magnitude pruning to compress this zonotope into a lower-order approximation. We derive two scale factors that exactly preserve the ℓ_2 norm of each cluster in order to prevent signal collapse during this geometric reduction.
- **Analytical Batch-Norm Merging:** We handle Batch Normalization layers entirely without data by applying the law of total variance. This analytically computes the merged running statistics and averaged scale parameters, perfectly preserving the statistical behavior of the compressed layer without retraining.
- **Theoretical Error Bounds:** We provide theoretical error bounds for TROPICS compressions based on the Hausdorff distance between zonotopes and show that they correlate with the observed accuracy drop (peaking at 0.97 and 0.98 for Pearson and Spearman metrics, respectively), yielding a data-free performance estimator.

2 Related Work

Tropical geometry and neural networks. Zhang et al. (2018) and Charisopoulos & Maragos (2018) establish that ReLU networks can be expressed as tropical rational functions, connecting decision boundaries to tropical hypersurfaces and providing a framework for analyzing their expressiveness.
Structured pruning. ThiNet (Luo et al., 2017) identifies redundant filters by minimizing reconstruction error on the next layer's input. CUP (Duggal et al., 2021) clusters activations and prunes based on cluster assignments. Both methods require access to training data. TROPICS is data-free and requires no forward pass through the network. Misiakos et al. (2022) and TropNNC (Fotopoulos et al., 2024) apply tropical geometry to structured compression, for single-output networks and for multi-output networks with an iterative optimization phase, respectively. Strong baselines for structured filter compression that rely on training data, either for sparsity-regularized training or for fine-tuning, include Network Slimming (Liu et al., 2017), filter pruning for efficient ConvNets (Li et al., 2017), and FPGM (He et al., 2019). Model folding (Wang et al., 2025) is a fine-tuning-free method, with data-free variants, that merges similar neurons and repair activation statistics.

3 Tropical Geometry of ReLU Neural Networks

In this section, we present the theoretical connection between ReLU neural networks and tropical geometry. Tropical geometry is based on the max-plus algebra, which geometrically characterizes polyhedral complexes. Since ReLU activations compute the maximum of linear segments, their Minkowski sum forms a special case of polytopes called zonotopes.

3.1 Neural Layers as Zonotopes

Consider a single layer with input $\boldsymbol{x} \in \mathbb{R}^n$ and weight matrix W. The pre-activation output of neuron i is $\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i$. As the input varies over a bounded domain, the reachable pre-activations of each neuron form a line segment $S_i = \{\lambda[\boldsymbol{w}_i, b_i]^\top : \lambda \in [0,1]\}$. A zonotope is the Minkowski sum of such segments. The *layer zonotope* is formed by summing the segments of all n neurons:

$$Z_{\text{layer}} = \sum_{i=1}^{n} S_i = \Big\{ \sum_{i=1}^{n} \lambda_i [\boldsymbol{w}_i, b_i]^\top : \lambda_i \in [0,1] \Big\}. \tag{1}$$

Each segment S_i acts as a *generator*. Intuitively, compressing the layer corresponds to approximating Z_{layer} with a new zonotope constructed from fewer generators. Detailed derivations and proofs are provided in Appendix D and Appendix E.

Figure 1 (left) shows a concrete example with three generators. ReLU networks partition the input space into polyhedral regions corresponding to distinct activation patterns. Figure 1 (right) represents geometrically our compression pipeline.

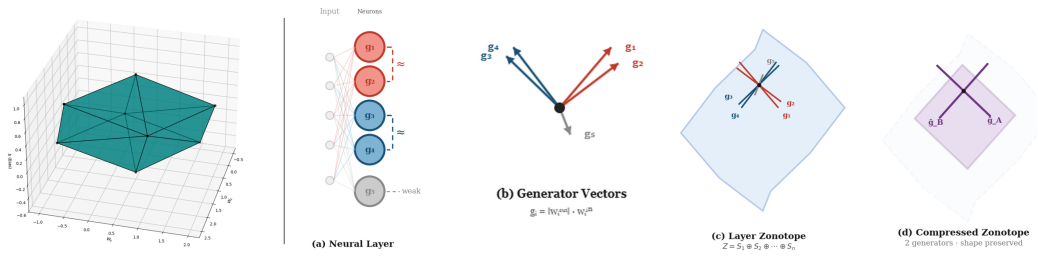


Figure 1: Zonotope geometry and the two-phase compression pipeline. **Left:** The 3D zonotope of $\mathcal{P}(\mathbf{x}) = \max(0,\, 1.0x_1{+}0.5x_2{+}1.0) + \max(0,\, -0.5x_1{+}1.5x_2{-}0.5) + \max(0,\, 1.5x_1{-}1.0x_2)$ in extended weight space $\mathbb{R}^3$, formed by three generators. **Right: (a)** Similar generators are clustered (red, blue) and weak generators targeted for pruning (grey). **(b)** The scaled generator vectors in 2D space. **(c)** The original layer zonotope $\mathcal{Z}$ formed as the Minkowski sum of all generator segments. **(d)** The compressed zonotope after K-means merging with energy-preserving scale factors.

3.2 Support Functions and Hausdorff Distance

Definition 3.1 (Support Function)**.** For a compact convex set $C \subset \mathbb{R}^n$, the support function $h_C : \mathbb{R}^n \to \mathbb{R}$ is defined as:

$$h_C(\boldsymbol{u}) = \sup_{\boldsymbol{x} \in C} \langle \boldsymbol{u}, \boldsymbol{x} \rangle \tag{2}$$

Theorem 3.2 (Hausdorff Distance Identity; proof in Appendix E)**.** *Let A and B be compact convex sets in $\mathbb{R}^n$. The Hausdorff distance $d_H(A, B)$ is given by the uniform norm of the difference of their support functions on the unit sphere:*

$$d_H(A, B) = \sup_{\|\boldsymbol{u}\|=1} |h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})| \tag{3}$$

4 Methodology

Each layer is compressed in two sequential phases: (1) *zonotope-guided clustering*, which merges groups of neurons or filters into single representatives while compensating for lost energy, and (2) *magnitude-based pruning*, which removes the weakest remaining neurons. The method is applied to both fully connected and convolutional layers. Intuitively, after aggressive clustering, merging dissimilar neurons is more harmful to accuracy than pruning a weak one. The generator definition in 4 is grounded in the tropical geometric structure of ReLU networks. This connection justifies both the form of the generators and the one-shot nature of the compression: no optimization step, no forward pass through the data, and no training samples are required.

One-shot compression and weight preparation: Generators are computed directly from the weight matrices. K-means clusters geometrically similar ones and merges each cluster into a single representative. Scale factors then restore the lost energy (6), all in a single pass, without training data and optimization. The entire compression pipeline is therefore executed in a single pass over the weight matrices, making TROPICS fast and data-free by construction. Before computing generators, layer weights are arranged into 2D. For fully connected layers, weights are natively 2D. For convolutional layers, the weights are reshaped into 2D by flattening each filter and permuting the output filters. After compression, the 2D weights are reshaped back to their original 4D shape.

Generator scaling for multiple outputs: In a single-output network, the generator of neuron i is exactly the extended vector $[\boldsymbol{w}_i, b_i]^\top$ scaled by the scalar output weight. In a multi-output network, where each neuron feeds into n_{out} downstream neurons the output weight is a vector $\boldsymbol{w}_i^{\text{out}} \in \mathbb{R}^{n_{\text{out}}}$. The total contribution of neuron i to the output space is captured by scaling the extended input vector by the Euclidean norm of the output weight vector:

$$\boldsymbol{g}_i = \|\boldsymbol{w}_i^{\text{out}}\|_2 \cdot [\boldsymbol{w}_i^{\text{in}},\, b_i]^\top \;\in\; \mathbb{R}^{n_{\text{in}}+1}. \tag{4}$$

The Euclidean norm aggregates the importance of neuron i across all output dimensions. Neurons with a large $\|\boldsymbol{w}_i^{\text{out}}\|_2$ have line segments that dominate the zonotope volume and should be preserved. Neurons with small norms contribute little to the output range and are candidates for removal.

4.1 COMPRESSION PIPELINE

4.1.1 PHASE 1: ZONOTOPE-GUIDED K-MEANS CLUSTERING

Before clustering, the generators are standardized and ℓ_2-normalized. This normalization makes the K-means objective equivalent to grouping by cosine similarity, so that directionally similar generators are clustered together regardless of their magnitude. The normalized generators $\{\tilde{\boldsymbol{g}}_i\}_{i=1}^{n}$ are then partitioned into K clusters using K-means, where $K = \lfloor r \cdot n \rfloor$ and $r \in (0, 1)$ is a per-layer compression ratio. Let $\mathcal{C}_1, \ldots, \mathcal{C}_K$ denote the resulting partition.

Merged weights. For each cluster $\mathcal{C}_k$, the compressed neuron is formed by

$$\bar{\boldsymbol{w}}_k^{\text{in}} = \frac{1}{|\mathcal{C}_k|} \sum_{i \in \mathcal{C}_k} \boldsymbol{w}_i^{\text{in}}, \qquad \bar{\boldsymbol{w}}_k^{\text{out}} = \sum_{i \in \mathcal{C}_k} \boldsymbol{w}_i^{\text{out}}. \tag{5}$$

The input weights are averaged (preserving the direction of the cluster centroid) and the output weights are summed (preserving the total output signal of the cluster).

Scale factors. To restore variance and energy lost during averaging and summing:

$$\alpha_k^{\text{in}} = \frac{\sigma\left(\{\boldsymbol{w}_i^{\text{in}}\}_{i \in \mathcal{C}_k}\right)}{\sigma(\bar{\boldsymbol{w}}_k^{\text{in}}) + \varepsilon}, \qquad \alpha_k^{\text{out}} = \frac{\sum_{i \in \mathcal{C}_k} \|\boldsymbol{w}_i^{\text{out}}\|_2}{\|\bar{\boldsymbol{w}}_k^{\text{out}}\|_2 + \varepsilon}. \tag{6}$$

The final compressed weights are $\hat{\boldsymbol{w}}_k^{\text{in}} = \alpha_k^{\text{in}} \cdot \bar{\boldsymbol{w}}_k^{\text{in}}$, $\hat{\boldsymbol{w}}_k^{\text{out}} = \alpha_k^{\text{out}} \cdot \bar{\boldsymbol{w}}_k^{\text{out}}$. Intuitively, α_k^{in} restores the spread of the input weights within the cluster, and α_k^{out} restores the total output energy. Together, the two scale factors amplify the compressed layer to recover what was lost during averaging.

The Error Bound is given by the Hausdorff distance between the original and the compressed zonotopes (proof in Appendix F):

$$\lim_{\varepsilon \to 0} d_H(Z, Z') \leq \sum_{k=1}^{K} \left[\sqrt{2S_k(S_k - M_k)} + (\alpha_k^{\text{in}} - 1)S_k \right], \quad S_k = \sum_{i \in \mathcal{C}_k} \|\boldsymbol{w}_i^{\text{out}}\|_2, \quad M_k = \left\| \sum_{i \in \mathcal{C}_k} \boldsymbol{w}_i^{\text{out}} \right\|_2$$

For ResNet experiments, we set the input scaling factor to $\alpha_k^{\text{in}} = 1$ to tighten the error bound. However, the sequential nature of VGG requires input signal amplification throughout deep layer to avoid signal vanishing. For ResNets, the residual connections deal with it without input scaling.

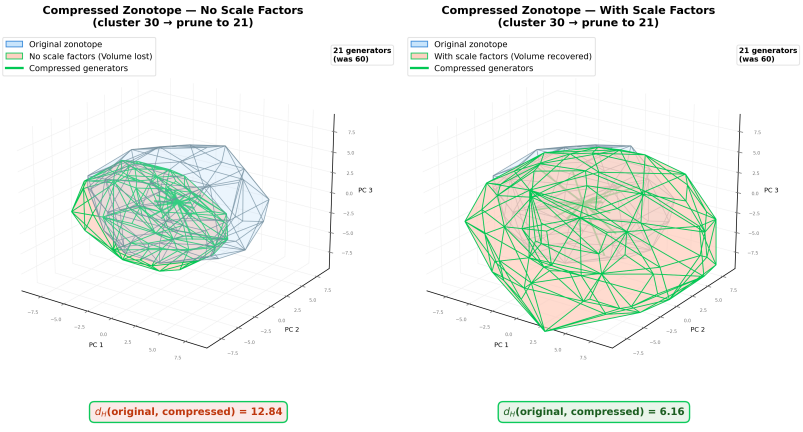


Figure 2: Hausdorff distance between the original and the compressed zonotopes (FC layer 0, FashionMNIST, 3D PCA projection). The uncompressed zonotope shrinks significantly.

4.1.2 PHASE 2: MAGNITUDE-BASED PRUNING

After clustering, each layer contains K compressed neurons. A second compression step is applied, retaining only the strongest $p\%$, where strength is measured by the product of input and output norms:

$$s_k = \left\| \|\hat{\boldsymbol{w}}_k^{\text{out}}\|_2 \cdot \hat{\boldsymbol{w}}_k^{\text{in}} \right\|_2 = \|\hat{\boldsymbol{w}}_k^{\text{in}}\|_2 \cdot \|\hat{\boldsymbol{w}}_k^{\text{out}}\|_2. \tag{7}$$

Algorithm 1: TROPICS Phase 1 — Zonotope-Guided Clustering

Input: Weights W, W^{next}, biases b, compression ratio r, layer type

`// Step 0: Prepare weights`

if *convolutional layer* **then**

  Flatten each filter: $\boldsymbol{w}_i^{\text{in}} \leftarrow \text{flatten}(W[i,:,:,:])$

  Permute & flatten W^{next} to get $\boldsymbol{w}_i^{\text{out}}$

else

  $\boldsymbol{w}_i^{\text{in}} \leftarrow$ row i of W; $\boldsymbol{w}_i^{\text{out}} \leftarrow$ col i of W^{next} `// FC: 2D, no reshape`

`// Step 1: Compute and normalized generators`

Compute $\boldsymbol{g}_i = [\|\boldsymbol{w}_i^{\text{out}}\|_2 \cdot \boldsymbol{w}_i^{\text{in}},\ \|\boldsymbol{w}_i^{\text{out}}\|_2 \cdot b_i]$ (4)

Standardize then ℓ_2-normalize: $\tilde{\boldsymbol{g}}_i$

`// Step 2: Cluster and merge`

Set $K \leftarrow \lfloor r \cdot n \rfloor$

Run K-means on $\{\tilde{\boldsymbol{g}}_i\}$ to get clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$

foreach *cluster* $\mathcal{C}_k$ **do**

  $\bar{\boldsymbol{w}}_k^{\text{in}} \leftarrow \text{mean}(\{\boldsymbol{w}_i^{\text{in}}\}_{i\in\mathcal{C}_k})$; $\bar{\boldsymbol{w}}_k^{\text{out}} \leftarrow \text{sum}(\{\boldsymbol{w}_i^{\text{out}}\}_{i\in\mathcal{C}_k})$

  Compute α_k^{in}; α_k^{out} via 6

  $\hat{\boldsymbol{w}}_k^{\text{in}} \leftarrow \alpha_k^{\text{in}} \cdot \bar{\boldsymbol{w}}_k^{\text{in}}$; $\hat{\boldsymbol{w}}_k^{\text{out}} \leftarrow \alpha_k^{\text{out}} \cdot \bar{\boldsymbol{w}}_k^{\text{out}}$

`// Step 3: Restore shape (conv only)`

if *convolutional layer* **then**

  Reshape $\hat{\boldsymbol{w}}_k^{\text{in}}$ back to $C_{\text{in}} \times K_h \times K_w$

  Permute $\hat{\boldsymbol{w}}_k^{\text{out}}$ back to $C_{\text{out}}^{\text{next}} \times C_{\text{out}} \times K_h' \times K_w'$

Output: Compressed weights $\{\hat{\boldsymbol{w}}_k^{\text{in}}, \hat{\boldsymbol{w}}_k^{\text{out}}\}_{k=1}^{K}$

Neurons are ranked by s_k in descending order and the bottom $(1-p)\%$ are discarded.
The Error Bound is given by the Hausdorff distance between the original and the compressed zonotopes for pruning:

$$d_H(Z, Z_{\setminus P}) \leq \sum_{i\in P} \|\boldsymbol{w}_i\|_2, \quad \text{where } P \subseteq \{1,\ldots,n\} \text{ is the set of pruned neurons or filters}$$

Theoretical Error Bound for the Mixed method is (proof in Appendix F):

$$\lim_{\varepsilon\to 0} d_H(Z, Z') \leq \underbrace{\sum_{i\in P} \|\boldsymbol{w}_i^{\text{out}}\|_2}_{\text{Pruning Error}} + \underbrace{\sum_{k=1}^{K} \left[\sqrt{2S_k(S_k - M_k)} + (\alpha_k^{\text{in}} - 1)S_k\right]}_{\text{Clustering / Merging Error}}$$

4.1.3 Batch Normalization

For convolutional layers followed by batch normalization, the compression is applied to the normalization parameters as part of the same one-shot pipeline. The learned scale and running mean are averaged across cluster members: $\hat{\gamma}_k = \frac{1}{|\mathcal{C}_k|}\sum_{i\in\mathcal{C}_k}\gamma_i$ and $\hat{\mu}_k = \frac{1}{|\mathcal{C}_k|}\sum_{i\in\mathcal{C}_k}\mu_i$. To compensate for the variance reduction introduced by averaging the means, the merged running variance is computed via the law of total variance:

$$\hat{\sigma}_k^2 = \underbrace{\frac{1}{|\mathcal{C}_k|}\sum_{i\in\mathcal{C}_k}\sigma_i^2}_{\text{mean of variances}} + \underbrace{\frac{1}{|\mathcal{C}_k|}\sum_{i\in\mathcal{C}_k}(\mu_i - \hat{\mu}_k)^2}_{\text{variance of means}}. \tag{8}$$

5 Experiments

In this section, we analyze the Setup that was used, we show comparisons of different variants of TROPICS, and finally we compare TROPICS with prior works.

5.1 Setup

TROPICS is evaluated on four architectures. A **custom fully connected network** is used for MNIST (LeCun et al. (1998)) and FashionMNIST (Xiao et al. (2017)). A **custom CNN** is evaluated on

MNIST, FashionMNIST, and CIFAR-10. **VGG16 with batch normalization** is evaluated on CIFAR-10 and CIFAR-100 (Krizhevsky (2009)). **ResNet18** is evaluated on ImageNet (Deng et al. (2009)). **No fine-tuning** is applied after compression. Results are reported as accuracy drop relative to the uncompressed baseline and as absolute accuracy.

Strategies & Protocol: TROPICS enables three data-free, one-shot modes: **Clustering** ($p = 1$, targeting directional redundancy), **Pruning** ($r = 1$, targeting magnitude redundancy), and **Mixed** ($r, p \in (0, 1)$, targeting both). Each layer has a distinct value of p and r, leading to a non-uniform compression. After the initial stage of removing duplicated filters without choosing ratios, a manual selection of p and r was set regarding our scope. For parameter reduction, we set higher late layers compression ratios (low p and r). For flops reduction, early layers or uniform compression ratios were chosen. All results are averaged across 5 K-means runs and evaluated on the test sets.

5.2 Results

In this section, we present the effect of the scale factors overcoming the accuracy drop, unlike the unscaled variant. In addition, we perform comparisons that show that the Mixed strategy with generators clustering is the optimal approach.

CIFAR-10. On VGG16-BN, the extensive compression finds our Mixed strategy achieves only a 0.58% drop at a massive 84.7% parameter reduction (2.33M), while Pruning after removing the duplicated filters by Clustering, achieves *zero* accuracy drop at 2.51M parameters. Importantly, for heavy Mixed compression (e.g., 92.5% reduction), TROPICS scale factors address the severe accuracy degradation, preserving over 30% accuracy. Full numbers in Table 2.

CIFAR-100. Pruning after removing the duplicated filters by Clustering with 1016 filters merged and 41 filters pruned matches the baseline accuracy with 0.02% drop while achieving a 45.8% parameter reduction (15.30M $\rightarrow$ 8.29M). Crucially, as the network is pushed into heavy Mixed compression (e.g., 75.6% reduction), unscaled methods suffer catastrophic collapse, while TROPICS scale factors solve this signal decay, preserving over 18% accuracy. Full numbers in Table 3.

ImageNet (ResNet18). The positive effect of scale factors holds for the pretrained `torchvision` ResNet18 architecture as well. Notably, our Clustering method with scaling, saves over 8% accuracy for a 35.3% parameter reduction. More details are presented in Table 4.

Figure 3 shows the effect of the scale factors on the accuracy for each dataset.

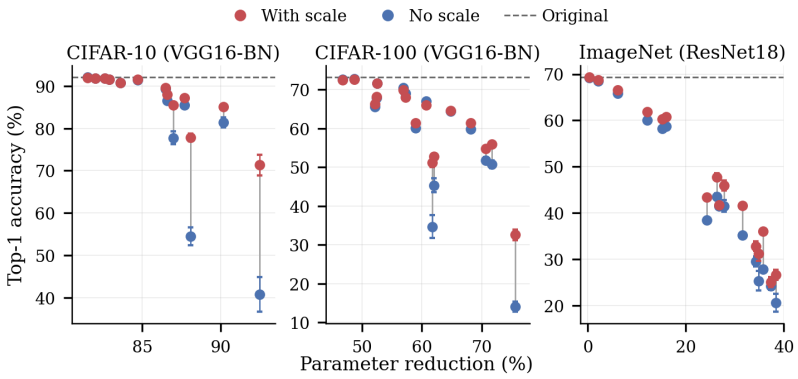


Figure 3: Accuracy of compressed models with and without TROPICS scale factors on CIFAR-10 (VGG16-BN), CIFAR-100 (VGG16-BN), and ImageNet (ResNet18).

Mixed versus pure clustering. Figure 4 isolates the contribution of the pruning phase on CIFAR-10, CIFAR-100 and ImageNet. The results are the average across 5 different seed executions. Full numbers are given in Table 7 (CIFAR-10), Table 8 (CIFAR-100) and Table 9 (ImageNet).

Generators versus filter clustering. Figure 5 compares generator clustering with filters-only clustering (without multiplying by the norm of the output weights). At matched initial compression ratios, merging generators reaches a larger parameter reduction, at equivalent or higher accuracies. That is because of the ability to find more duplicated filters than the set compression ratios require. The full numbers are given in Table 10.

Bound correlation with the observed accuracy drop. Figure 6 shows that the computed Hausdorff bound correlates with the measured accuracy drop across architectures, datasets, and compression

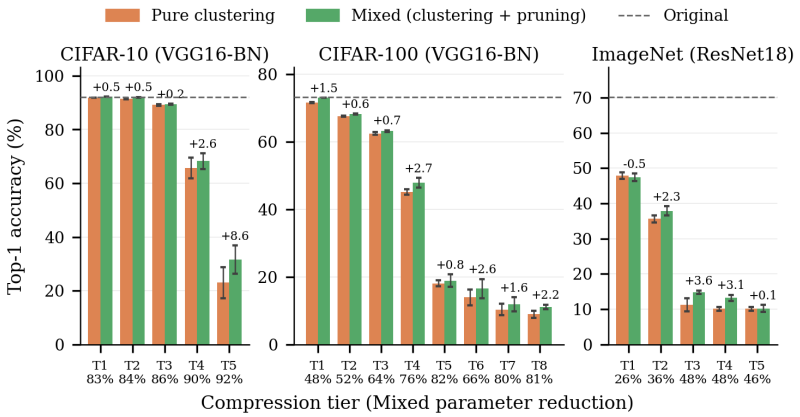


Figure 4: Mixed compression versus pure clustering under the scale variant, across compression tiers on CIFAR-10, CIFAR-100 and ImageNet. Annotations give the accuracy gained by the mixed strategy at each tier. Bars show the mean over five runs and error bars the standard deviation.

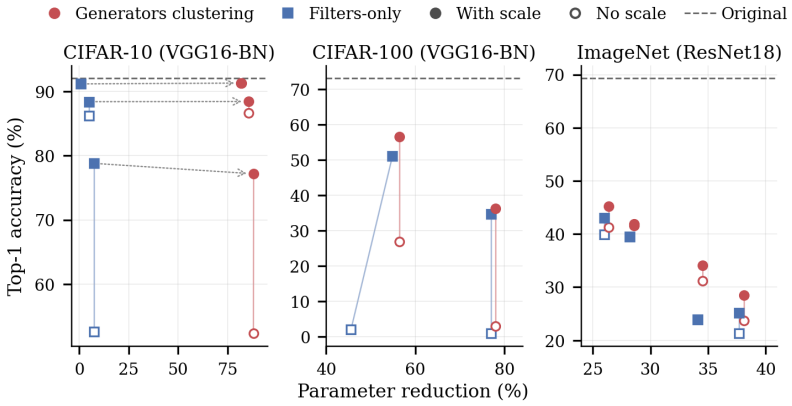


Figure 5: Generators clustering vs. filters-only clustering on VGG16-BN. Filled/hollow markers: with/without TROPICS scale factors. Arrows connect the same configurations.

strategies. Full per-configuration numbers and correlation statistics are given in Table 14, Table 15, Table 16 and Table 17 (Appendix A).

5.3 Comparison with Prior Work

Comparisons with TropNNC and CUP

The top row of Figure 7 compares TROPICS against TropNNC (Fotopoulos et al., 2024) and CUP (Duggal et al., 2021). The corresponding numbers are given in Table 5 and Table 6. Tropics scaling variant significantly outperforms TropNNC and CUP, proving that scaling impact is the main reason for preserving accuracy without the need for data (CUP) or data-free optimization (TropNNC). TROPICS scaled variant achieves an accuracy drop of only 3.69%, whereas without scaling, accuracy drops by 61.15%. At equivalent FLOPs and parameters, TropNNC with optimization suffers a 23.69% drop, and 58.75% without iterative optimization. On ImageNet (ResNet-18), TROPICS again outperforms TropNNC at equivalent FLOPs and parameters without requiring optimization. TROPICS drops by 8.05% at 9.82 million parameters, whereas TropNNC with optimization achieves a 10.6% drop. In addition, scaling improves accuracy for TROPICS under the higher-compression mixed strategy, retaining 1.3% accuracy.

Comparisons with Model Folding

Rows (b) and (c) of Figure 7 compare TROPICS with Model Folding (Wang et al., 2025) on ResNet18 (ImageNet-Val) at matched parameter counts and on VGG11-BN (CIFAR-10) at matched FLOPs reductions. TROPICS outperforms both data-free variants (DF-REPAIR and NO-REPAIR) at every

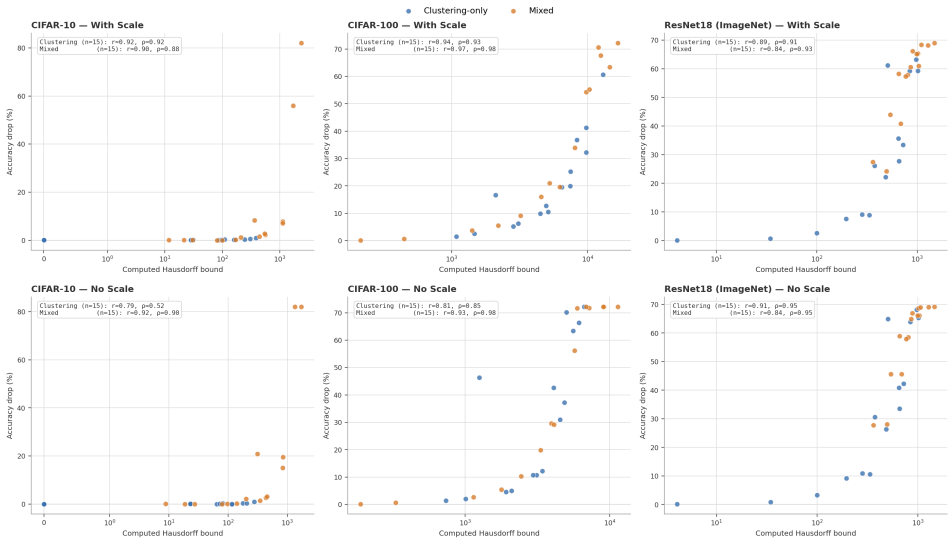


Figure 6: Pearson (r) and Spearman rank (ρ) correlations between the computed Hausdorff bound and the measured accuracy drop.

budget, with the gap widening as compression increases, and remains competitive with the data-dependent REPAIR variant (details in Table 11, Table 12, and Table 13).

TROPICS outperforms every data-free variant of Model Folding in both FLOPs and parameter matching for CIFAR-10 on the VGG-11 network. On ImageNet, TROPICS outperforms Model Folding at mild compression levels below 23% FLOPs reduction. The degradation of TROPICS under aggressive FLOPs reduction arises from the scope. For safety and simplicity reasons, TROPICS does not compress residual connections and focuses only on the first convolutional layers. This explains the higher accuracies achieved at mild compression levels. However, aggressively low FLOPs requires significant reductions to the initial convolutional layers, sacrificing useful information. Nevertheless, such severe FLOPs reduction is unnecessary in practical data-free applications, as accuracy collapses for both models at those levels, applying a fine-tuning stage becomes necessary. In terms of parameters, TROPICS outperforms Model Folding on data-free variants and it is competitive to the data-dependent. The scaling effect is once again crucial for restoring the accuracy lost during compression. The scaling gains reach an increase of 28% for VGG-11 on CIFAR-10 at an 80% parameter compression rate, and 3% for ResNet on ImageNet at a 23% parameter reduction.

Classic filter-pruning baselines.

Table 1 additionally compares TROPICS against three classic filter-pruning baselines on CIFAR-10, using each paper's own published results. TROPICS reaches a strictly smaller parameter and FLOP budget than every reported baseline configuration at zero accuracy drop, and it does so without access to training data. We can reasonably assume that they do not achieve a zero drop immediately after compression, which is precisely why they require a retraining phase.

6 Conclusion & Future Work

We introduced TROPICS, a data-free, layer-wise compression framework grounded in the zonotopic geometry of ReLU layers. TROPICS first clusters zonotope generators with K-means and merges each cluster, using two scale factors to recover the energy lost during averaging, and then prunes the weakest remaining neurons by magnitude. Intuitively, after aggressive clustering the remaining variation between neurons is meaningful, so pruning a weak neuron is safer than merging two dissimilar ones. Batch Normalization layers are merged analytically, and the Hausdorff-based error bound correlates strongly with the observed accuracy drop, providing a data-free estimate of performance degradation. Across four architectures and five datasets, from MNIST to ImageNet, TROPICS achieves an 84.7% parameter reduction on VGG16-BN (CIFAR-10) with only a 0.44% accuracy drop, and outperforms TropNNC, CUP, and Model Folding on ResNet18 (ImageNet). TROPICS has so far been tested only with ReLU activations. However, since the compression criteria

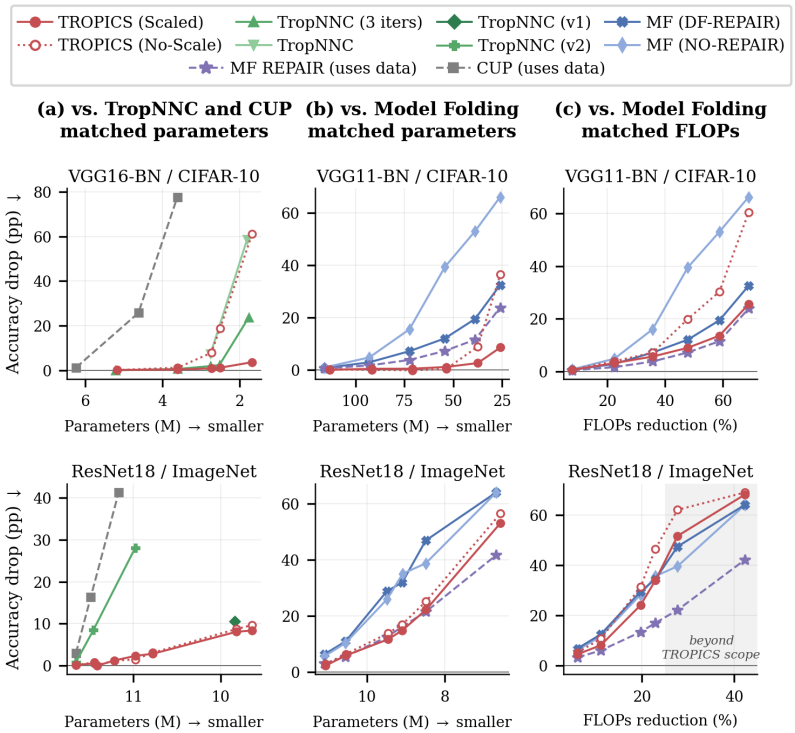


Figure 7: TROPICS against prior data-free and data-dependent compression methods at matched parameter counts. Column (a): comparison with TropNNC and CUP on VGG16-BN / CIFAR-10 and ResNet18 / ImageNet. Columns (b) and (c): comparison with Model Folding on ResNet18 / ImageNet and VGG11-BN / CIFAR-10.

Table 1: TROPICS vs. classic filter-pruning baselines on CIFAR-10. These baselines require data and retraining to recover accuracy (only FPGM reports immediate post-compression drop). Checkpoints were reproduced using the original papers' setups because they are not public available.

Method	Data-Free	Params (M)	FLOPs (M)	Acc (%) ↑	Drop ↓
Pruning Filters for Efficient ConvNets (Li et al., 2017), VGG-16, CIFAR-10					
VGG-16 (paper baseline)	–	15.0	626	93.25	–
VGG-16-pruned-A (paper)	No	5.4	412	93.40	−0.15
TROPICS (baseline)	–	14.99	629.14	94.27	–
TROPICS	Yes	1.11	271.85	94.27	0.00
Filter Pruning via Geometric Median (FPGM) (He et al., 2019), VGG-16, CIFAR-10					
FPGM (paper baseline)	–	14.99	629.14	93.58	–
FPGM (paper, Data-Free part)	Yes	2.94	413.97	80.38	13.20
TROPICS (baseline)	–	14.99	629.14	93.91	–
TROPICS	Yes	1.06	264.52	93.91	0.00
Network Slimming (Liu et al., 2017), VGG-19, CIFAR-10					
VGGNet (paper baseline)	–	20.04	797	93.66	–
VGGNet, 70% pruned (paper)	No	2.30	391	93.80	−0.14
TROPICS (baseline)	–	20.04	798.70	94.02	–
TROPICS	Yes	0.88	250.80	94.02	0.00

are weight-based, extending the method to other non-linearities is feasible. Extending the zonotopic analysis to Transformer architectures and LLMs are promising directions as well.

AI USE STATEMENT

In this work, we used generative AI tools for text polishing and identifying relevant related work. We also used AI tools for cross-validating mathematical claims and polishing formal notation.

ETHICS STATEMENT

This work focuses on foundational algorithmic methods for neural network weight compression. Our experiments rely exclusively on publicly available, standard benchmark datasets and pre-trained models, involving no human subjects, sensitive personal data, or crowdsourcing. Consequently, this study presents no direct privacy, security, or human-subject concerns.

REPRODUCIBILITY STATEMENT

The code and details about the implementation are available at `https://anonymous.4open.science/r/anonymous-D271/README.md`.

REFERENCES

V. Charisopoulos and P. Maragos. A Tropical Approach to Neural Networks with Piecewise Linear Activations. *arXiv:1805.08749*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009.

R. Duggal, C. Xiao, R. Vuduc, D. H. Chau, and J. Sun. Cup: Cluster pruning for compressing deep neural networks. In Yixin Chen, Heiko Ludwig, Yicheng Tu, Usama Fayyad, Xingquan Zhu, Xiaohua Tony Hu, Suren Byna, Xiong Liu, Jianping Zhang, Shirui Pan, Vagelis Papalexakis, Jianwu Wang, Alfredo Cuzzocrea, and Carlos Ordonez (eds.), *Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021*, Proceedings - 2021 IEEE International Conference on Big Data, Big Data 2021, pp. 5102–5106, United States, 2021. Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/BigData52589.2021.9671980. Publisher Copyright: © 2021 IEEE.; 2021 IEEE International Conference on Big Data, Big Data 2021 ; Conference date: 15-12-2021 Through 18-12-2021.

K Fotopoulos, P. Maragos, and P. Misiakos. Tropnnc: Structured neural network compression using tropical geometry. *arXiv preprint arXiv:2409.03945*, 2024.

S. Haddad and A. Halder. A note on the hausdorff distance between norm balls and their linear maps, 2023. URL https://arxiv.org/abs/2206.12012.

Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration, 2019. URL https://arxiv.org/abs/1811.00250.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

S. König. Computational aspects of the hausdorff distance in unbounded dimension, 2014. URL https://arxiv.org/abs/1401.1434.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets, 2017. URL https://arxiv.org/abs/1608.08710.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming, 2017. URL https://arxiv.org/abs/1708.06519.

J.H. Luo, J. Wu, and W. Lin. Thinet: A filter level pruning method for deep neural network compression, 2017. URL https://arxiv.org/abs/1707.06342.

Petros Maragos, Vasileios Charisopoulos, and Emmanouil Theodosis. Tropical geometry and machine learning. *Proceedings of the IEEE*, 109(5):728–755, 2021. doi: 10.1109/JPROC.2021.3065238.

P. Misiakos, G. Smyrnis, G. Retsinas, and P. Maragos. Neural network approximation based on hausdorff distance of tropical zonotopes. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=oiZJwC_fyS.

Dong Wang, Haris Šikić, Lothar Thiele, and Olga Saukh. Forget the data and fine-tuning! just fold the network to compress, 2025. URL https://arxiv.org/abs/2502.10216.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

L. Zhang, G. Naitzat, and L.H. Lim. Tropical geometry of deep neural networks, 2018. URL https://arxiv.org/abs/1805.07091.

A Detailed Numerical Results

This appendix reports the full numerical results summarized by the figures in the main text: the per-configuration compression sweeps (Table 2, Table 3, Table 4), the comparisons with prior work (Table 5, Table 6, Table 11, Table 12, Table 1, Table 13), the mixed-versus-clustering tiers (Table 7, Table 8, Table 9) and the clustering ablation (Table 10), and the Hausdorff bound versus accuracy drop correlation (Table 14, Table 15, Table 16, Table 17). Table numbering follows the order in which the results are discussed in the main text. M is the number of merged neurons, P is the pruned count. Cells with two values report *with scale / no scale*. Red. is parameter reduction.

Table 2: CIFAR-10 (VGG16-BN) compression. Original: 92.09%, $15.25\mathrm{M}$ params, $629.66\mathrm{M}$ FLOPs. Because VGG16 lacks residual connections, scale factors are strictly required to prevent signal collapse, preserving over 30% absolute accuracy at extreme compression. Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

						With Scale		No Scale	
Strategy	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
Original	–	–	15.25	629.66	–	92.09	–	92.09	–
Baseline merge	2382/2376	0	2.88	390.14	81.1	92.08 ± 0.00	0.01	92.09 ± 0.00	0.00
Pruning	2382/2376	70/71	2.78	389.41	81.8	92.18 ± 0.01	-0.09	92.15 ± 0.00	-0.06
Pruning	2382/2376	144/145	2.63	386.20	82.8	92.12 ± 0.00	-0.03	92.13 ± 0.00	-0.04
Pruning	2382/2376	217/220	2.51	383.20	83.5	92.09 ± 0.00	0.00	92.10 ± 0.00	-0.01
Pruning	2382/2376	255/258	2.37	366.93	84.5	91.76 ± 0.00	0.33	91.67 ± 0.00	0.42
Pruning	2382/2376	286/287	2.17	310.94	85.8	90.00 ± 0.00	2.09	89.97 ± 0.00	2.12
Pruning	2382/2376	382/385	2.11	349.44	86.2	87.14 ± 0.00	4.95	87.56 ± 0.00	4.53
Pruning	2382/2376	559/560	1.56	189.76	89.8	66.08 ± 0.00	26.01	66.79 ± 0.00	25.30
Clustering	2431	0	2.82	389.80	81.5	92.02 ± 0.03	0.07	92.04 ± 0.03	0.05
Clustering	2420/2414	0	2.74	372.59	82.0	91.77 ± 0.10	0.32	91.77 ± 0.08	0.32
Clustering	2456/2454	0	2.61	357.55	82.9	91.55 ± 0.16	0.54	91.55 ± 0.08	0.54
Clustering	2516/2510	0	2.50	297.37	83.6	90.72 ± 0.09	1.37	90.83 ± 0.04	1.26
Clustering	2688/2682	0	2.04	207.12	86.6	88.05 ± 0.43	4.04	86.49 ± 0.32	5.60
Clustering	2726/2720	0	1.99	179.74	87.0	85.47 ± 0.45	6.62	77.76 ± 1.57	14.33
Clustering	2816/2810	0	1.82	135.15	88.1	77.85 ± 0.95	14.24	54.52 ± 2.10	37.57
Mixed	2420/2414	53/54	2.66	372.03	82.6	91.83 ± 0.08	0.26	91.82 ± 0.06	0.27
Mixed	2456/2454	125	2.33	341.26	84.7	91.51 ± 0.10	0.58	91.39 ± 0.07	0.70
Mixed	2420/2414	340/343	2.06	336.72	86.5	89.55 ± 0.13	2.54	89.27 ± 0.12	2.82
Mixed	2516/2510	263/264	1.88	212.14	87.7	87.19 ± 0.43	4.90	85.46 ± 0.77	6.63
Mixed	2627/2621	343/346	1.49	176.69	90.2	85.04 ± 0.63	7.05	81.41 ± 1.21	10.68
Mixed	2688/2682	466/467	1.15	100.92	92.5	71.32 ± 2.43	20.77	40.74 ± 4.07	51.35

Table 3: CIFAR-100 (VGG16-BN) compression. Original: 73.21%, 15.30M params, 629.76M FLOPs. Scale factors buffer catastrophic signal collapse under heavy mixed compression, preserving over 18% absolute accuracy. Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

						With Scale		No Scale	
Strategy	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
Original	–	–	15.30	629.76	–	73.21	–	73.21	–
Baseline merge	1016/1015	0	8.44	541.08	44.8	73.21 ± 0.00	0.00	73.21 ± 0.00	0.00
Pruning	1016/1015	41	8.29	539.91	45.8	73.19 ± 0.01	0.02	73.21 ± 0.00	0.00
Pruning	1016/1015	158	7.88	536.84	48.5	73.05 ± 0.00	0.16	73.03 ± 0.00	0.18
Pruning	1016/1015	236	7.64	535.00	50.1	72.54 ± 0.00	0.67	72.53 ± 0.00	0.68
Pruning	1016/1015	364	6.84	510.70	55.3	70.65 ± 0.00	2.56	70.60 ± 0.00	2.61
Pruning	1016/1015	620	5.45	468.49	64.4	61.40 ± 0.00	11.81	61.38 ± 0.00	11.83
Pruning	1016/1015	894	4.65	429.37	69.6	38.49 ± 0.00	34.72	38.49 ± 0.00	34.72
Clustering	1064/1063	0	8.14	531.47	46.8	72.53 ± 0.17	0.68	72.69 ± 0.12	0.52
Clustering	1270/1269	0	7.31	412.21	52.2	66.26 ± 0.33	6.95	65.53 ± 0.55	7.68
Clustering	1414	0	7.26	532.32	52.5	71.69 ± 0.23	1.52	71.66 ± 0.25	1.55
Clustering	1420	0	6.53	491.71	57.3	68.09 ± 0.21	5.12	69.10 ± 0.07	4.11
Clustering	1596/1595	0	5.86	290.32	61.7	51.13 ± 1.03	22.08	34.70 ± 2.94	38.51
Clustering	1692	0	5.38	467.58	64.8	64.64 ± 0.38	8.57	64.45 ± 0.40	8.76
Clustering	1851	0	4.49	431.84	70.7	54.73 ± 0.81	18.48	51.82 ± 0.84	21.39
Mixed	1064/1063	80	7.85	529.22	48.7	72.66 ± 0.17	0.55	72.82 ± 0.16	0.39
Mixed	1099/1098	164	7.29	443.45	52.4	68.19 ± 0.17	5.02	67.88 ± 0.17	5.33
Mixed	1064/1063	359	6.59	502.63	56.9	69.94 ± 0.16	3.27	70.47 ± 0.12	2.74
Mixed	1301/1300	155	6.27	398.25	59.0	61.44 ± 0.39	11.77	60.17 ± 0.51	13.04
Mixed	1446	101	6.02	454.69	60.7	66.07 ± 0.13	7.14	67.02 ± 0.17	6.19
Mixed	1201/1200	421	5.81	322.01	62.0	52.81 ± 0.43	20.40	45.33 ± 1.84	27.88
Mixed	1614	201	4.86	453.90	68.2	61.45 ± 0.68	11.76	59.80 ± 0.48	13.41
Mixed	1731	207	4.33	389.82	71.7	55.98 ± 0.58	17.23	50.79 ± 0.98	22.42
Mixed	1483/1482	662/663	3.74	228.97	75.6	32.58 ± 1.40	40.63	14.03 ± 1.28	59.18

Table 4: ImageNet (ResNet18) compression. Original: 69.34%, 11.69M params, 3648M FLOPs. Scale factors preserve up to 6.0% absolute accuracy at extreme compression ($\leq 7.26\text{M}$ params). Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

						With Scale		No Scale	
Strategy	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
Original	–	–	11.69	3648.07	–	69.34	–	69.34	–
Pruning	0	154	10.27	3508.92	12.1	58.90 ± 0.00	10.44	58.90 ± 0.00	10.44
Pruning	0	206	9.91	3473.68	15.2	56.85 ± 0.00	12.49	56.85 ± 0.00	12.49
Pruning	0	308	8.85	3369.77	24.3	37.63 ± 0.00	31.71	37.63 ± 0.00	31.71
Pruning	0	359	8.62	3346.95	26.3	42.78 ± 0.00	26.56	42.78 ± 0.00	26.56
Pruning	0	475	8.00	3206.61	31.6	36.51 ± 0.00	32.83	36.51 ± 0.00	32.83
Pruning	0	487	7.56	3242.81	35.3	27.27 ± 0.00	42.07	27.27 ± 0.00	42.07
Pruning	0	512	7.56	3243.25	35.3	32.69 ± 0.00	36.65	32.69 ± 0.00	36.65
Clustering	3	0	11.66	3645.36	0.3	69.29 ± 0.00	0.05	69.23 ± 0.00	0.11
Clustering	26	0	11.45	3624.58	2.1	68.73 ± 0.03	0.61	68.50 ± 0.08	0.84
Clustering	77	0	10.98	3578.49	6.1	66.60 ± 0.10	2.74	65.84 ± 0.17	3.50
Clustering	154	0	10.27	3508.92	12.1	61.82 ± 0.23	7.52	60.05 ± 0.22	9.29
Clustering	206	0	9.91	3473.68	15.2	60.23 ± 0.22	9.11	58.28 ± 0.38	11.06
Clustering	231	0	9.82	3464.86	16.0	60.71 ± 0.21	8.63	58.77 ± 0.24	10.57
Clustering	308	0	8.85	3369.77	24.3	43.34 ± 0.62	26.00	38.48 ± 0.51	30.86
Clustering	359	0	8.62	3346.95	26.3	47.74 ± 0.78	21.60	43.50 ± 0.74	25.84
Clustering	475	0	8.00	3206.61	31.6	41.54 ± 0.44	27.80	35.18 ± 0.38	34.16
Clustering	487	0	7.56	3242.81	35.3	31.19 ± 1.57	38.15	25.32 ± 2.16	44.02
Clustering	512	0	7.56	3243.25	35.3	36.06 ± 0.36	33.28	27.86 ± 0.69	41.48
Mixed	359	13	8.50	3335.20	27.3	45.88 ± 1.09	23.46	41.50 ± 1.25	27.84
Mixed	26	346	8.50	3335.20	27.3	41.71 ± 0.20	27.63	41.46 ± 0.15	27.88
Mixed	282	189	7.68	3254.78	34.3	32.81 ± 1.03	36.53	29.52 ± 1.07	39.82
Mixed	487	36	7.26	3214.34	37.9	26.59 ± 1.12	42.75	20.56 ± 1.93	48.78
Mixed	78	445	7.26	3214.34	37.9	24.99 ± 1.10	44.35	24.22 ± 0.84	45.12

Table 5: Comparison with TropNNC and CUP on CIFAR-10 (VGG16-BN) at matched parameter counts. All methods' drops are relative to the uncompressed baseline (93.64%).

Method	Data-Free	Params (M)	FLOPs (G)	Acc (%) ↑	Drop ↓
CUP (Duggal et al., 2021)	No	6.24	0.46	92.41	1.23
TropNNC (3 iters) (Fotopoulos et al., 2024)	Yes	5.22	0.46	93.58	0.06
TROPICS (Mixed, Scaled)	Yes	5.18	0.46	93.45	0.19
TROPICS (Mixed, No-Scale)	Yes	**5.18**	**0.46**	**93.46**	**0.18**
CUP (Duggal et al., 2021)	No	4.62	0.42	67.91	25.73
TropNNC (3 iters) (Fotopoulos et al., 2024)	Yes	3.61	0.42	93.02	0.62
CUP (Duggal et al., 2021)	No	3.61	0.39	16.02	77.62
TROPICS (Mixed, Scaled)	Yes	**3.60**	**0.43**	**93.22**	**0.42**
TROPICS (Mixed, No-Scale)	Yes	3.60	0.43	92.43	1.21
TropNNC (3 iters) (Fotopoulos et al., 2024)	Yes	2.75	0.39	91.71	1.93
TropNNC (Fotopoulos et al., 2024)	Yes	2.75	0.39	86.20	7.44
TROPICS (Mixed, Scaled)	Yes	**2.73**	**0.40**	**92.73**	**0.91**
TROPICS (Mixed, No-Scale)	Yes	2.73	0.40	85.74	7.90
TropNNC (3 iters) (Fotopoulos et al., 2024)	Yes	2.52	0.38	91.20	2.44
TROPICS (Mixed, Scaled)	Yes	**2.50**	**0.40**	**92.43**	**1.21**
TROPICS (Mixed, No-Scale)	Yes	2.50	0.40	74.70	18.94
TropNNC (3 iters) (Fotopoulos et al., 2024)	Yes	1.76	0.33	69.95	23.69
TropNNC (Fotopoulos et al., 2024)	Yes	1.80	0.33	34.89	58.75
TROPICS (Mixed, Scaled)	Yes	**1.67**	**0.34**	**89.95**	**3.69**
TROPICS (Mixed, No-Scale)	Yes	1.67	0.34	32.49	61.15

Table 6: Comparison with TropNNC and CUP on ImageNet (ResNet18). Drops are relative to each method's uncompressed baseline (CUP/TropNNC: 69.75%; TROPICS: 69.40%, where minor baseline differences stem from evaluation on different unseeded random 2,000-image subsets of the test set while using identical model weights). Methods are grouped by comparable parameter counts.

Method	Data-Free	Params (M)	FLOPs (G)	Acc (%) ↑	Drop ↓
CUP (Duggal et al., 2021)	No	11.66	3.58	66.80	2.95
TropNNC (v2) (Fotopoulos et al., 2024)	Yes	11.66	3.48	68.95	0.80
TROPICS (Clustering, No-Scale)	Yes	**11.66**	**3.65**	**69.30**	**0.10**
TROPICS (Clustering, Scaled)	Yes	11.66	3.65	69.25	0.15
CUP (Duggal et al., 2021)	No	11.49	3.38	53.40	16.35
TropNNC (v2) (Fotopoulos et al., 2024)	Yes	11.46	3.25	61.30	8.45
TROPICS (Clustering, Scaled)	Yes	**11.45**	**3.62**	**68.90**	**0.50**
TROPICS (Clustering, No-Scale)	Yes	11.45	3.62	68.65	0.75
TROPICS (Mixed, Scaled)	Yes	**11.42**	**3.62**	**69.45**	**-0.05**
TROPICS (Mixed, No-Scale)	Yes	11.42	3.62	69.35	0.05
TROPICS (Mixed, Scaled)	Yes	11.22	3.60	68.20	1.20
TROPICS (Mixed, No-Scale)	Yes	**11.22**	**3.60**	**68.25**	**1.15**
CUP (Duggal et al., 2021)	No	11.17	3.15	28.45	41.30
TropNNC (v2) (Fotopoulos et al., 2024)	Yes	10.98	2.92	41.65	28.10
TROPICS (Clustering, No-Scale)	Yes	**10.98**	**3.58**	**67.95**	**1.45**
TROPICS (Clustering, Scaled)	Yes	10.98	3.58	67.10	2.30
TROPICS (Mixed, Scaled)	Yes	**10.78**	**3.56**	**66.60**	**2.80**
TROPICS (Mixed, No-Scale)	Yes	10.78	3.56	66.35	3.05
TropNNC (v1) (Fotopoulos et al., 2024)	Yes	9.84	3.46	59.15	10.60
TROPICS (Clustering, Scaled)	Yes	**9.82**	**3.46**	**61.35**	**8.05**
TROPICS (Clustering, No-Scale)	Yes	9.82	3.46	60.70	8.70
TROPICS (Mixed, Scaled)	Yes	**9.64**	**3.45**	**61.05**	**8.35**
TROPICS (Mixed, No-Scale)	Yes	9.64	3.45	59.75	9.65

Table 7: Comparison between the mixed and the clustering approach under the scale variant on CIFAR-10 (VGG16-BN, Original: 92.09%, 15.25M params, 629.66M FLOPs). Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

Compression Tier	Approach	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓
Tier 1 (Surpassing Baseline)	Pure clustering (Line 21)	2467	0	2.70	385.67	82.3	91.80 ± 0.04	0.29
	Mixed (Line 6)	**2382**	**129**	**2.62**	**384.56**	**82.8**	$\mathbf{92.25 \pm 0.00}$	$\mathbf{-0.16}$
Tier 2 (High Sparsity)	Pure clustering	2615	0	2.41	378.45	84.2	91.43 ± 0.17	0.66
	Mixed	**2503**	**143**	**2.37**	**378.41**	**84.5**	$\mathbf{91.98 \pm 0.12}$	**0.11**
Tier 3 (FLOP Reduction)	Pure clustering	2676	0	2.09	235.71	86.3	89.16 ± 0.30	2.93
	Mixed	**2580**	**97**	**2.09**	**231.57**	**86.3**	$\mathbf{89.34 \pm 0.21}$	**2.75**
Tier 4 (Aggressive 90%)	Pure clustering	2907	0	1.62	107.77	89.4	65.65 ± 3.88	26.44
	Mixed	**2802**	**148**	**1.53**	**103.88**	**90.0**	$\mathbf{68.25 \pm 2.92}$	**23.84**
Tier 5 (Extreme 92%)	Pure clustering	3091	0	1.32	57.06	91.3	22.98 ± 5.83	69.11
	Mixed	**2997**	**145**	**1.22**	**54.29**	**92.0**	$\mathbf{31.55 \pm 5.21}$	**60.54**

Table 8: Comparison between the mixed and the clustering approach under the scale variant on CIFAR-100 (VGG16-BN, Original: 73.21%, 15.30M params, 629.76M FLOPs) across low-to-high compression tiers. Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

Compression Tier	Approach	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓
Low Tier 1 ($\sim$ 534M FLOPs)	Pure clustering	1090	0	7.97	520.34	47.9	71.52 ± 0.16	1.69
	Mixed	**1016**	**113**	**7.95**	**534.16**	**48.0**	$\mathbf{73.03 \pm 0.00}$	**0.18**
Low Tier 2 ($\sim$ 443M FLOPs)	Pure clustering	1257	0	7.23	439.19	52.7	67.58 ± 0.16	5.63
	Mixed	**1099**	**164**	**7.29**	**443.45**	**52.4**	$\mathbf{68.19 \pm 0.17}$	**5.02**
Low Tier 3 ($\sim$ 458M FLOPs)	Pure clustering	1542	0	5.67	459.41	62.9	62.42 ± 0.43	10.79
	Mixed	**1434**	**147**	**5.50**	**458.06**	**64.1**	$\mathbf{63.11 \pm 0.30}$	**10.10**
Mid Tier 4 ($\sim$ 418M FLOPs)	Pure clustering	2088	0	3.73	417.26	75.6	45.10 ± 0.76	28.11
	Mixed	**1698**	**391**	**3.72**	**418.01**	**75.7**	$\mathbf{47.84 \pm 1.40}$	**25.37**
High Tier 5 ($\sim$ 396M FLOPs)	Pure clustering	2532	0	2.67	396.81	82.5	18.08 ± 0.94	55.13
	Mixed	**2160**	**372**	**2.67**	**396.81**	**82.5**	$\mathbf{18.84 \pm 1.87}$	**54.37**
High Tier 6 ($\sim$ 182M FLOPs)	Pure clustering	1833	0	5.33	187.23	65.2	13.91 ± 2.39	59.30
	Mixed	**1695**	**172**	**5.17**	**182.62**	**66.2**	$\mathbf{16.51 \pm 2.85}$	**56.70**
Deep Tier 7 ($\sim$ 124M FLOPs)	Pure clustering	2358	0	3.28	126.74	78.6	10.32 ± 1.74	62.89
	Mixed	**2236**	**160**	**3.14**	**124.77**	**79.5**	$\mathbf{11.88 \pm 2.03}$	**61.33**
Deep Tier 8 ($<$ 124M FLOPs)	Pure clustering	2421	0	2.99	125.54	80.5	8.93 ± 1.06	64.28
	Mixed	**1966**	**489**	**2.92**	**123.02**	**80.9**	$\mathbf{11.13 \pm 0.65}$	**62.08**

Table 9: Comparison between the mixed and the clustering approach under the scale variant on ImageNet (ResNet18, Original: 70.15% (random 2,000-image subsets of the test set while using the pre-trained torchvision model weights), 11.69M params, 3648.07M FLOPs). Accuracies are the mean $\pm$ standard deviation over five runs with different K-means seeds; drops are computed from the mean (their standard deviation equals that of the accuracy, since the baseline is fixed).

Compression Tier	Approach	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓
Tier 1 ($\sim$ 3345M FLOPs)	Pure clustering	359	0	8.62	3346.95	26.3	47.86 ± 0.96	22.29
	Mixed	**272**	**89**	**8.60**	**3345.36**	**26.4**	$\mathbf{47.37 \pm 1.05}$	**22.78**
Tier 2 ($\sim$ 3241M FLOPs)	Pure clustering	512	0	7.56	3243.25	35.3	35.58 ± 1.02	34.57
	Mixed	**416**	**98**	**7.54**	**3241.67**	**35.5**	$\mathbf{37.88 \pm 1.32}$	**32.27**
Tier 3 ($\sim$ 2936M FLOPs)	Pure clustering	744	0	6.08	2940.44	48.0	11.21 ± 1.83	58.94
	Mixed	**584**	**163**	**6.06**	**2936.60**	**48.2**	$\mathbf{14.80 \pm 0.48}$	**55.35**
Tier 4 ($\sim$ 2937M FLOPs)	Pure clustering	641	0	6.31	3121.04	46.0	10.13 ± 0.55	60.02
	Mixed	**695**	**51**	**6.06**	**2937.96**	**48.2**	$\mathbf{13.21 \pm 0.84}$	**56.94**
Tier 5 ($\sim$ 3121M FLOPs)	Pure clustering	641	0	6.31	3121.04	46.0	10.13 ± 0.55	60.02
	Mixed	**623**	**18**	**6.31**	**3121.04**	**46.0**	$\mathbf{10.20 \pm 0.95}$	**59.95**

Table 10: Comparison between generators clustering and only filters clustering across CIFAR-10, CIFAR-100 (VGG16-BN), and ImageNet (ResNet18).

							With Scale		No Scale	
Dataset	Method	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
CIFAR-10 (Original: 92.09%, 15.25M *params,* 629.66M *FLOPs)*										
CIFAR-10	Generators clustering	2442 / 2436	0	2.76 / 2.77	329.09 / 329.13	81.9 / 81.8	91.32	0.77	91.30	0.79
CIFAR-10	Filters-only	60 / 60	0	15.13	568.59	0.8	91.19	0.90	91.23	0.86
CIFAR-10	Generators clustering	2640 / 2634	0	2.19 / 2.20	219.50 / 219.54	85.6 / 85.6	88.45	3.64	86.65	5.44
CIFAR-10	Filters-only	272 / 272	0	14.48	448.79	5.0	88.42	3.67	86.24	5.85
CIFAR-10	Generators clustering	2816 / 2810	0	1.82 / 1.83	135.15 / 135.19	88.1 / 88.0	77.18	14.91	52.33	39.76
CIFAR-10	Filters-only	448 / 448	0	14.10	362.74	7.5	78.81	13.28	52.54	39.55
CIFAR-100 (Original: 73.21%, 15.30M *params,* 629.76M *FLOPs)*										
CIFAR-100	Generators clustering	1645	0	6.75	528.44	55.9	56.59	16.62	26.94	46.27
CIFAR-100	Filters-only	1633 / 1386	0	6.83 / 8.24	531.18 / 574.21	55.4 / 46.1	51.14	22.07	2.09	71.12
CIFAR-100	Generators clustering	2307	0	3.44	423.89	77.5	36.38	36.83	3.02	70.19
CIFAR-100	Filters-only	2307	0	3.44	423.89	77.5	34.81	38.40	1.00	72.21
ImageNet (ResNet18, Original: 69.34%, 11.69M *params,* 3648.07M *FLOPs)*										
ImageNet	Generators clustering	359	13	8.50	3335.20	27.3	45.17	24.17	41.32	28.02
ImageNet	Filters-only	359	13	8.50	3335.20	27.3	43.02	26.32	39.91	29.43
ImageNet	Generators clustering	26	346	8.50	3335.20	27.3	41.93	27.41	41.59	27.76
ImageNet	Filters-only	26	346	8.50	3335.20	27.3	39.46	29.89	39.46	29.89
ImageNet	Generators clustering	487	36	7.26	3214.34	37.9	28.51	40.84	23.72	45.63
ImageNet	Filters-only	487	36	7.26	3214.34	37.9	25.14	44.20	21.33	48.02
ImageNet	Generators clustering	282	189	7.68	3254.78	34.3	34.09	35.25	31.14	38.20
ImageNet	Filters-only	282	189	7.68	3254.78	34.3	23.90	45.44	23.90	45.44

Table 11: TROPICS vs. Model Folding (Wang et al., 2025) on ResNet18 (ImageNet-Val) at matched parameter counts; both methods share the same uncompressed baseline (69.34%). Model Folding is reported in its data-free variants (DF-REPAIR, NO-REPAIR) and its data-dependent variant (REPAIR). TROPICS is data-free throughout.

Method	Data-Free	Params (M)	Acc (%) ↑	Drop ↓
TROPICS (Scaled)	Yes	11.07	67.09	2.25
TROPICS (No-Scale)	Yes	11.07	66.48	2.86
Model Folding (DF-REPAIR)	Yes	11.10	63.05	6.29
Model Folding (NO-REPAIR)	Yes	11.10	63.78	5.56
Model Folding (REPAIR)	No	11.10	66.33	3.01
TROPICS (Scaled)	Yes	10.54	63.44	5.90
TROPICS (No-Scale)	Yes	10.54	62.94	6.41
Model Folding (DF-REPAIR)	Yes	10.55	58.44	10.90
Model Folding (NO-REPAIR)	Yes	10.55	58.99	10.35
Model Folding (REPAIR)	No	10.55	63.90	5.44
TROPICS (Scaled)	Yes	9.46	57.70	11.64
TROPICS (No-Scale)	Yes	9.46	55.55	13.79
Model Folding (DF-REPAIR)	Yes	9.49	40.59	28.75
Model Folding (NO-REPAIR)	Yes	9.48	43.37	25.97
Model Folding (REPAIR)	No	9.49	56.84	12.50
TROPICS (Scaled)	Yes	9.09	54.48	14.86
TROPICS (No-Scale)	Yes	9.09	52.35	16.99
Model Folding (DF-REPAIR)	Yes	9.09	37.38	31.96
Model Folding (NO-REPAIR)	Yes	9.08	34.29	35.05
Model Folding (REPAIR)	No	9.09	53.19	16.15
TROPICS (Scaled)	Yes	8.48	47.12	22.22
TROPICS (No-Scale)	Yes	8.48	44.13	25.21
Model Folding (DF-REPAIR)	Yes	8.49	22.50	46.84
Model Folding (NO-REPAIR)	Yes	8.49	30.62	38.72
Model Folding (REPAIR)	No	8.49	47.89	21.45
TROPICS (Scaled)	Yes	6.57	16.31	53.03
TROPICS (No-Scale)	Yes	6.57	12.88	56.46
Model Folding (DF-REPAIR)	Yes	6.68	5.36	63.98
Model Folding (NO-REPAIR)	Yes	6.68	5.42	63.92
Model Folding (REPAIR)	No	6.68	27.72	41.62

Table 12: TROPICS vs. Model Folding (Wang et al., 2025) on VGG11-BN (CIFAR-10) at matched parameter counts; both methods share the same uncompressed baseline (90.54%, 128.81M params). Model Folding is reported in its data-free variants (DF-REPAIR, NO-REPAIR) and its data-dependent variant (REPAIR). TROPICS is data-free throughout.

Method	Data-Free	Params (M)	FLOPs (M)	Acc (%) ↑	Drop ↓
TROPICS (Scaled)	Yes	113.41	513.07	90.62	-0.08
TROPICS (No-Scale)	Yes	113.41	513.07	90.46	0.08
Model Folding (DF-REPAIR)	Yes	116.16	503.10	89.96	0.58
Model Folding (NO-REPAIR)	Yes	116.16	503.10	89.74	0.80
Model Folding (REPAIR)	No	116.16	503.10	90.08	0.46
TROPICS (Scaled)	Yes	91.65	466.60	90.22	0.32
TROPICS (No-Scale)	Yes	91.65	466.60	90.64	-0.10
Model Folding (DF-REPAIR)	Yes	93.01	424.36	87.74	2.80
Model Folding (NO-REPAIR)	Yes	93.01	424.36	85.87	4.67
Model Folding (REPAIR)	No	93.01	424.36	88.93	1.61
TROPICS (Scaled)	Yes	70.91	422.32	90.18	0.36
TROPICS (No-Scale)	Yes	70.91	422.32	90.79	-0.25
Model Folding (DF-REPAIR)	Yes	72.47	352.86	83.48	7.06
Model Folding (NO-REPAIR)	Yes	72.47	352.86	75.11	15.43
Model Folding (REPAIR)	No	72.47	352.86	86.84	3.70
TROPICS (Scaled)	Yes	53.67	385.49	89.52	1.02
TROPICS (No-Scale)	Yes	53.67	385.49	90.33	0.21
Model Folding (DF-REPAIR)	Yes	54.30	285.33	78.57	11.97
Model Folding (NO-REPAIR)	Yes	54.30	285.33	51.23	39.31
Model Folding (REPAIR)	No	54.30	285.33	83.37	7.17
TROPICS (Scaled)	Yes	37.46	343.75	88.03	2.51
TROPICS (No-Scale)	Yes	37.46	343.75	81.72	8.82
Model Folding (DF-REPAIR)	Yes	38.88	224.75	71.16	19.38
Model Folding (NO-REPAIR)	Yes	38.88	224.75	37.58	52.96
Model Folding (REPAIR)	No	38.88	224.75	79.07	11.47
TROPICS (Scaled)	Yes	25.90	309.28	82.04	8.50
TROPICS (No-Scale)	Yes	25.90	309.28	54.01	36.53
Model Folding (DF-REPAIR)	Yes	26.05	170.37	58.16	32.38
Model Folding (NO-REPAIR)	Yes	26.05	170.37	24.48	66.06
Model Folding (REPAIR)	No	26.05	170.37	66.82	23.72

Table 13: Comparison on CIFAR-10 and ImageNet at matched FLOPs reductions.

FLOPs ↓	TROPICS		Model Folding (MF)		
	Scaled	**No Scale**	**No Repair**	**DF-REPAIR**	**REPAIR**[†]
VGG-11 / CIFAR-10 (Unpruned: 90.54%; Schedule: Clustering / MACs-weighted)					
7.9%	89.94	89.88	89.74	89.96	90.08
22.5%	87.35	86.64	85.67	87.66	88.89
35.7%	84.88	83.40	74.52	83.36	86.75
47.9%	81.54	70.62	51.02	78.45	83.30
58.8%	76.96	60.28	37.56	71.14	79.06
68.9%	65.03	30.23	24.35	58.00	66.68
ResNet-18 / ImageNet (Unpruned: 69.34%; Schedule: Mixed / Front-loaded)					
6.0%	64.79	64.12	63.21	62.50	66.04
11.1%	61.15	58.60	57.56	56.80	63.25
19.7%	45.17	37.94	41.01	39.76	55.89
22.9%	35.34	22.76	33.67	34.85	52.29
27.7%	17.63	7.21	29.68	21.86	47.14
42.3%	1.11	0.28	5.26	5.20	27.12

[†]Uses data during post-folding repair.

A.1 HAUSDORFF BOUND VERSUS ACCURACY DROP

Full per-configuration numbers underlying Figure 6 in the main text.

Table 14: Correlation between the computed Hausdorff bound and the measured accuracy drop across architectures, datasets, and compression strategies (clustering-only vs. mixed clustering+pruning), with and without the scale factor. Each cell reports Pearson's r / Spearman's ρ.

Architecture	Dataset	Strategy	n	With Scale (r / ρ)	No Scale (r / ρ)
VGG16-BN	CIFAR-10	Clustering-only	15	0.921 / 0.916	0.794 / 0.524
		Mixed	15	0.904 / 0.882	0.915 / 0.901
	CIFAR-100	Clustering-only	15	0.936 / 0.932	0.812 / 0.846
		Mixed	15	0.973 / 0.982	0.933 / 0.978
ResNet18	ImageNet	Clustering-only	15	0.895 / 0.914	0.915 / 0.950
		Mixed	15	0.841 / 0.932	0.844 / 0.951

Table 15: Per-configuration Hausdorff bound and measured accuracy drop for VGG16-BN on CIFAR-10, split by compression strategy (clustering-only vs. mixed clustering+pruning), with and without the scale factor.

Strategy	Config	Merged	Pruned	Bound (Scale)	Drop% (Scale)	Bound (No Scale)	Drop% (No Scale)
Clustering-only ($n = 15$)							
	C1	2431	0	28.52	0.07	23.00	0.04
	C2	2574	0	161.60	0.16	116.69	-0.06
	C3	2382 / 2376	0	0.00	0.01	0.00	0.00
	C4	2382 / 2380	0	0.00	0.01	0.00	0.00
	C5	2394 / 2388	0	31.32	0.17	23.24	0.14
	C6	2382 / 2376	0	0.00	0.01	0.00	0.00
	C7	2431	0	28.52	0.07	23.00	0.04
	C8	2444	0	109.95	0.28	81.12	0.36
	C9	2499	0	86.96	0.04	64.57	0.01
	C10	2382 / 2380	0	0.00	0.01	0.00	0.00
	C11	2574	0	161.60	0.16	116.69	-0.06
	C12	2531	0	245.83	0.36	178.26	0.36
	C13	2481	0	98.24	0.24	71.88	0.12
	C14	2612	0	387.26	0.90	277.11	0.89
	C15	2659	0	306.33	0.49	206.79	0.37
Mixed ($n = 15$)							
	C1	2382 / 2376	53 / 54	21.48	0.01	18.50	0.00
	C2	2382 / 2376	74 / 75	30.31	0.05	27.43	0.02
	C3	2420 / 2414	144 / 145	170.04	0.16	139.92	0.21
	C4	2531 / 2525	199 / 200	561.89	2.28	435.05	2.64
	C5	2382 / 2376	35 / 36	11.72	0.02	8.84	0.05
	C6	2382 / 2376	185 / 186	99.88	-0.03	96.74	0.05
	C7	2519	266	546.64	2.77	455.43	3.12
	C8	2382 / 2376	160 / 163	80.63	-0.08	78.69	-0.06
	C9	2506 / 2500	156 / 159	444.39	1.51	345.64	1.39
	C10	2431	276	210.58	1.21	201.09	2.10
	C11	2790	316	1122.97	7.78	852.05	19.53
	C12	2778 / 2772	598 / 601	1722.76	55.92	1346.49	82.03
	C13	2687 / 2681	348 / 349	1130.66	7.07	842.81	15.08
	C14	2541	299	363.48	8.33	312.23	20.86
	C15	2940 / 2934	648 / 651	2372.73	82.09	1725.37	82.09

B Network Architectures and Training Hyperparameters

The architectures and training configurations for all four network families are described below. No fine-tuning is applied after compression in any experiment.

B.1 Fully Connected Network (MNIST, FashionMNIST)

A custom fully connected network is used for MNIST and FashionMNIST. The architecture consists of three hidden layers with 60 neurons each, followed by a 10-class output layer. ReLU activations are applied after each hidden layer. The input is a flattened image of size $1 \times 28 \times 28 = 784$.

Table 16: Per-configuration Hausdorff bound and measured accuracy drop for VGG16-BN on CIFAR-100, split by compression strategy (clustering-only vs. mixed clustering+pruning), with and without the scale factor.

Strategy	Config	Merged	Pruned	Bound (Scale)	Drop% (Scale)	Bound (No Scale)	Drop% (No Scale)
Clustering-only ($n = 15$)							
	C1	1414	0	1084.75	1.52	736.09	1.43
	C2	1645	0	2106.92	16.62	1253.59	46.27
	C3	1237	0	1468.80	2.54	1007.97	2.06
	C4	1492	0	2836.64	5.21	1914.22	4.59
	C5	1459	0	3081.28	6.30	2094.65	4.96
	C6	1614	0	4946.54	12.75	3115.94	10.72
	C7	1767	0	4478.95	9.90	2947.12	10.77
	C8	1770	0	5124.84	10.52	3407.45	12.25
	C9	2076	0	6464.59	19.49	4081.85	42.60
	C10	1998	0	7468.77	25.24	4524.78	31.01
	C11	2307	0	8350.51	36.83	5015.33	70.19
	C12	2115	0	7459.34	19.92	4854.41	37.26
	C13	2304	0	9733.84	41.25	5569.03	63.33
	C14	2388	0	9757.18	32.21	6122.72	66.30
	C15	2616	0	12982.60	60.64	6643.96	72.21
Mixed ($n = 15$)							
	C1	1057	113	214.02	0.11	189.24	0.10
	C2	1134	197	1412.45	3.77	1139.43	2.68
	C3	1142 / 1141	359	2200.14	5.52	1782.97	5.39
	C4	1387 / 1386	372	5256.20	21.04	3932.36	29.61
	C5	1189	63	447.33	0.68	331.31	0.64
	C6	1410	418	4562.68	16.05	3320.53	19.82
	C7	1926	387	8047.17	33.97	5700.98	56.17
	C8	1721	302	6234.44	19.60	4101.23	29.14
	C9	1260 / 1259	353 / 354	3216.27	9.13	2430.82	10.28
	C10	2154	372	9763.04	54.31	5937.05	71.60
	C11	2694	345	14528.77	63.43	9085.45	72.21
	C12	1849	1065	12021.60	70.56	9032.02	72.21
	C13	1862	597	10303.93	55.27	7189.55	71.71
	C14	2460	324	12490.01	67.67	6870.17	72.21
	C15	2322	1018	16662.24	72.21	11348.85	72.21

B.2 CUSTOM CNN (MNIST, FASHIONMNIST, CIFAR-10)

A custom convolutional network is used for MNIST, FashionMNIST, and CIFAR-10. The feature extractor consists of three convolutional blocks with 32, 64, and 128 output channels respectively. Each block applies a 5×5 convolution with `padding=2`, followed by ReLU, 2×2 max pooling, and Dropout2d ($p = 0.25$). The classifier is a single linear layer preceded by standard Dropout ($p = 0.25$).

B.3 VGG16 WITH BATCH NORMALIZATION (CIFAR-10, CIFAR-100)

A VGG16-style network with batch normalization is trained from scratch on CIFAR-10 and CIFAR-100. The feature extractor follows the standard VGG16 channel progression (64–64–128–128–256–256–256–512–512–512–512–512–512), with five max-pooling stages. Each convolutional block is followed by batch normalization and ReLU. Conv layers use 3×3 kernels with `padding=1` and no bias. Dropout2d ($p = 0.25$ for CIFAR-10, $p = 0.10$ for CIFAR-100) is applied after each batch normalization. The classifier consists of two hidden linear layers of width 512 with ReLU and Dropout ($p = 0.5$ / $p = 0.2$), followed by a 10- or 100-class output layer. Weights are initialised with Kaiming normal for convolutional layers, constant initialisation for batch normalization, and normal initialisation ($\sigma = 0.01$) for linear layers.

B.4 RESNET18 (IMAGENET)

A pretrained ResNet18 is used for ImageNet evaluation, loaded with `ResNet18_Weights.DEFAULT` from `torchvision`. No further training is performed; the pretrained weights are used directly. The standard ImageNet preprocessing pipeline provided by the weights object is applied at inference time.

Table 17: Per-configuration Hausdorff bound and measured accuracy drop for the ResNet18 (ImageNet) configurations used in Table 14 and the corresponding figure. Clustering-only: pruned$= 0$. Mixed: merged> 0 and pruned> 0.

Strategy	Config	Merged	Pruned	Bound (Scale)	Drop% (Scale)	Bound (No Scale)	Drop% (No Scale)
Clustering-only ($n = 15$)							
	C1	231	0	334.89	8.86	334.89	10.56
	C2	3	0	4.07	0.06	4.07	0.11
	C3	26	0	34.55	0.64	34.55	0.86
	C4	77	0	99.88	2.64	99.88	3.27
	C5	154	0	195.77	7.58	195.77	9.16
	C6	206	0	282.32	9.08	282.32	10.88
	C7	308	0	375.31	26.10	375.31	30.58
	C8	359	0	484.14	22.16	484.14	26.35
	C9	436	0	506.35	61.22	506.35	64.89
	C10	512	0	720.95	33.38	720.95	42.28
	C11	487	0	652.09	35.67	652.09	40.80
	C12	475	0	660.51	27.76	660.51	33.54
	C13	641	0	844.68	59.28	844.68	63.88
	C14	718	0	975.14	63.21	975.14	68.11
	C15	744	0	1015.99	59.23	1015.99	65.32
Mixed ($n = 15$)							
	C1	359	13	495.33	24.17	495.33	28.02
	C2	26	346	362.41	27.41	362.41	27.76
	C3	487	36	687.55	40.83	689.12	45.63
	C4	78	445	538.97	43.95	540.36	45.63
	C5	641	11	862.26	60.53	859.56	64.95
	C6	52	599	655.50	58.27	657.53	58.90
	C7	718	31	1016.83	65.31	1012.16	68.62
	C8	103	646	802.64	57.80	805.81	58.48
	C9	744	16	1036.25	60.95	1034.58	66.12
	C10	52	707	767.15	57.30	768.53	57.95
	C11	769	32	1097.12	68.39	1064.98	68.98
	C12	155	645	899.90	66.18	880.50	66.92
	C13	923	25	1285.41	68.21	1279.80	69.02
	C14	103	844	989.14	65.11	992.51	66.15
	C15	1076	8	1482.85	69.01	1460.48	69.20

Table 18: FC network architecture.

Layer	Configuration
Input	Flatten 784
Hidden 1–3	Linear(60), ReLU
Output	Linear(10)

Compression is applied only to the first convolutional layer of each residual block (the `conv1` layers in `layer1`–`layer4`), as these are safe to prune without breaking residual connections. The `conv2` layers, downsampling projections, and the stem convolution are left unchanged.

C Small Datasets Results

On FC networks, Pruning is the dominant strategy because narrow layers lack directional redundancy, aggressive clustering collapses accuracy. On CNNs, all three strategies stay within 2% of the baseline at moderate compression, but accuracy drops by up to 5.2% at extreme reductions ($\approx 78\%$). Full results are in Appendix C.

CIFAR-10. Table 28 and Table 29 report results on CIFAR-10. On the custom CNN, Mixed achieves a 0.54% accuracy drop at mild compression (10.7% reduction), while pure Pruning yields a 1.04% drop at 17.9% reduction.

Table 19: FC network training hyperparameters.

Hyperparameter	Value
Optimizer	Adam
Learning rate	1×10^{-3}
Loss	Cross-entropy
Batch size	128
Epochs	60
Normalization	Mean 0.5, Std 0.5

Table 20: Custom CNN architecture.

Layer	Configuration
Conv Block 1	Conv2d(1/3, 32, 5×5, pad=2), ReLU, MaxPool(2), Dropout2d(0.25)
Conv Block 2	Conv2d(32, 64, 5×5, pad=2), ReLU, MaxPool(2), Dropout2d(0.25)
Conv Block 3	Conv2d(64, 128, 5×5, pad=2), ReLU, MaxPool(2), Dropout2d(0.25)
Classifier	Dropout(0.25), Linear(d, 10)

Table 21: Custom CNN training hyperparameters.

Hyperparameter	Value
Optimizer	Adam
Learning rate	1×10^{-3}
Loss	Cross-entropy
Batch size	128

Table 22: VGG16 training hyperparameters.

Hyperparameter	CIFAR-10	CIFAR-100
Optimizer	Adam	Adam
Learning rate	1×10^{-3}	1×10^{-3}
Weight decay	1×10^{-4}	1×10^{-5}
LR scheduler	CosineAnnealingLR	CosineAnnealingLR
Batch size	128	128
Epochs	200	300
Dropout (conv)	0.25	0.10
Dropout (classifier)	0.50	0.20
Augmentation	RandomCrop(32, pad=4), RandomHorizontalFlip, RandomErasing($p = 0.5$)	RandomCrop(32, pad=4), RandomHorizontalFlip, RandomErasing($p = 0.5$)
Normalization	(0.4914, 0.4822, 0.4465) / (0.2023, 0.1994, 0.2010)	(0.5071, 0.4867, 0.4408) / (0.2675, 0.2565, 0.2761)

Table 23: ResNet18 inference configuration.

Hyperparameter	Value
Pretrained weights	`ResNet18_Weights.DEFAULT`
Input resolution	224×224
Batch size	128
Preprocessing	`ResNet18_Weights.DEFAULT.transforms()`

Table 24: MNIST (custom FC) compression. Original: 97.21%, 44.86K params. Shows scaled vs. unscaled accuracy drops across critical configurations. Results averaged across five runs $\pm$ standard deviation (omitted when deterministic).

					With Scale		No Scale	
Strategy	M	P	Params (K)	Red. (%)	Acc (%)	Drop	Acc (%)	Drop
Original	–	–	44.86	–	97.21	–	97.21	–
Pruning	0	3	44.56	0.7	97.21	0.00	97.21	0.00
Pruning	0	10	36.51	18.6	97.21	0.00	97.21	0.00
Clustering	3	0	42.35	5.6	93.85 ± 3.77	3.36 ± 3.77	96.92 ± 0.14	0.29 ± 0.14
Clustering	6	0	42.06	6.2	93.56 ± 3.35	3.65 ± 3.35	96.70 ± 0.21	0.51 ± 0.21
Clustering	24	0	37.01	17.5	85.32 ± 2.00	11.89 ± 2.00	94.64 ± 0.31	2.57 ± 0.31
Mixed	5	14	32.73	27.0	87.11	10.10	96.67	0.54
Mixed	21	14	29.42	34.4	79.24 ± 9.50	17.97 ± 9.50	94.90 ± 0.60	2.31 ± 0.60
Mixed	15	26	25.15	43.9	72.21 ± 0.53	25.00 ± 0.53	78.74 ± 0.60	18.47 ± 0.60
Mixed	45	32	18.00	59.9	27.41 ± 6.86	69.80 ± 6.86	44.84 ± 1.31	52.37 ± 1.31

Table 25: FashionMNIST (custom FC) compression. Original: 87.56%, 55.03K params. Results averaged across five runs $\pm$ standard deviation (omitted when deterministic).

					With Scale		No Scale	
Strategy	M	P	Params (K)	Red. (%)	Acc (%)	Drop	Acc (%)	Drop
Original	–	–	55.03	–	87.56	–	87.56	–
Pruning	0	3	52.49	4.6	87.56	0.00	87.56	0.00
Pruning	0	12	44.89	18.4	82.41	5.15	82.41	5.15
Clustering	3	0	52.49	4.6	87.39	0.17	87.56	0.00
Clustering	6	0	52.14	5.3	87.53	0.03	87.45	0.11
Clustering	24	0	41.52	24.5	80.98 ± 1.14	6.58 ± 1.14	84.27 ± 1.09	3.29 ± 1.09
Clustering	54	0	37.01	32.7	68.64 ± 3.68	18.92 ± 3.68	75.36 ± 1.98	12.20 ± 1.98
Mixed	27	29	41.49	24.6	75.79 ± 1.34	11.77 ± 1.34	76.87 ± 1.73	10.69 ± 1.73
Mixed	6	17	40.04	27.2	76.94 ± 0.32	10.62 ± 0.32	77.81 ± 0.11	9.75 ± 0.11
Mixed	18	23	36.39	33.9	71.67 ± 0.84	15.89 ± 0.84	71.48 ± 1.31	16.08 ± 1.31
Mixed	54	39	21.71	60.5	42.30 ± 4.46	45.26 ± 4.46	51.98 ± 3.50	35.58 ± 3.50

Table 26: MNIST (custom CNN) compression. Original: 99.43%, 0.27M params. In convolutional architectures, the unscaled clustering reliably matches or outperforms the scaled version, particularly at extreme mixed compression levels. Results averaged across five runs $\pm$ standard deviation (omitted when deterministic).

					With Scale		No Scale	
Strategy	M	P	Params (M)	Red. (%)	Acc (%)	Drop	Acc (%)	Drop
Original	–	–	0.27	–	99.43	–	99.43	–
Pruning	0	7	0.26	3.7	99.43	0.00	99.43	0.00
Pruning	0	4	0.25	7.4	99.43	0.00	99.43	0.00
Pruning	0	13	0.22	18.5	99.40	0.03	99.40	0.03
Pruning	0	26	0.22	18.5	99.44	-0.01	99.44	-0.01
Clustering	7	0	0.26	3.7	99.43	0.00	99.43	0.00
Clustering	6	0	0.25	7.4	99.36	0.07	99.37	0.06
Clustering	13	0	0.22	18.5	99.42 ± 0.01	0.01 ± 0.01	99.41 ± 0.02	0.02 ± 0.02
Clustering	26	0	0.22	18.5	99.39 ± 0.03	0.04 ± 0.03	99.42 ± 0.02	0.01 ± 0.02
Clustering	35	0	0.19	29.6	99.35 ± 0.04	0.08 ± 0.04	99.37 ± 0.04	0.06 ± 0.04
Mixed	24	19	0.18	33.3	99.37 ± 0.08	0.06 ± 0.08	99.38 ± 0.06	0.05 ± 0.06
Mixed	25	23	0.17	37.0	99.40 ± 0.02	0.03 ± 0.02	99.44 ± 0.01	-0.01 ± 0.01
Mixed	35	26	0.14	48.1	99.29 ± 0.04	0.14 ± 0.04	99.33 ± 0.05	0.10 ± 0.05
Mixed	40	40	0.12	55.6	99.21 ± 0.02	0.22 ± 0.02	99.31 ± 0.03	0.12 ± 0.03
Mixed	69	48	0.06	77.8	97.87 ± 0.51	1.56 ± 0.51	98.40 ± 0.21	1.03 ± 0.21

Table 27: FashionMNIST (custom CNN) compression. Original: 92.88%, 0.27M params. The trend continues: removing the scale factors prevents the distortion of convolutional feature maps, preserving nearly 1% of accuracy in the most heavily compressed state. Results averaged across five runs $\pm$ standard deviation (omitted when deterministic).

					With Scale		No Scale	
Strategy	M	P	Params (M)	Red. (%)	Acc (%)	Drop	Acc (%)	Drop
Original	–	–	0.27	–	92.88	–	92.88	–
Pruning	0	7	0.26	3.7	92.88	0.00	92.88	0.00
Pruning	0	4	0.25	7.4	92.74	0.14	92.74	0.14
Pruning	0	13	0.22	18.5	92.53	0.35	92.53	0.35
Pruning	0	26	0.22	18.5	92.58	0.30	92.58	0.30
Clustering	7	0	0.26	3.7	92.80	0.08	92.80	0.08
Clustering	6	0	0.25	7.4	92.58	0.30	92.60	0.28
Clustering	13	0	0.22	18.5	92.26 ± 0.09	0.62 ± 0.09	92.33 ± 0.13	0.55 ± 0.13
Clustering	26	0	0.22	18.5	92.54	0.34	92.54	0.34
Clustering	35	0	0.19	29.6	92.02 ± 0.04	0.86 ± 0.04	92.28 ± 0.07	0.60 ± 0.07
Mixed	24	19	0.18	33.3	91.70	1.18	91.97	0.91
Mixed	25	23	0.17	37.0	91.61	1.27	92.23	0.65
Mixed	35	26	0.14	48.1	91.09 ± 0.04	1.79 ± 0.04	91.45 ± 0.13	1.43 ± 0.13
Mixed	40	40	0.12	55.6	91.05 ± 0.09	1.83 ± 0.09	91.25 ± 0.04	1.63 ± 0.04
Mixed	69	48	0.06	77.8	86.97 ± 0.96	5.91 ± 0.96	87.63 ± 0.69	5.25 ± 0.69

Table 28: CIFAR-10 (custom CNN) compression. Original: 80.76%, 0.28M params. Scale factors provide a massive accuracy preservation as mixed compression becomes aggressive, recovering over 16% absolute accuracy at extreme reductions. Results averaged across five runs $\pm$ standard deviation (omitted for deterministic pure Pruning).

					With Scale		No Scale	
Strategy	M	P	Params (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
Original	–	–	0.28	–	80.76	–	80.76	–
Pruning	0	4	0.26	7.1	80.70	0.06	80.70	0.06
Pruning	0	13	0.23	17.9	79.74	1.02	79.74	1.02
Clustering	6	0	0.26	7.1	80.33 ± 0.00	0.43 ± 0.00	80.60 ± 0.00	0.16 ± 0.00
Clustering	35	0	0.20	28.6	77.89 ± 0.07	2.87 ± 0.07	77.48 ± 0.23	3.28 ± 0.23
Mixed	7	7	0.25	10.7	80.21 ± 0.03	0.55 ± 0.03	80.09 ± 0.00	0.67 ± 0.00
Mixed	25	23	0.18	35.7	75.63 ± 0.19	5.13 ± 0.19	73.93 ± 0.07	6.83 ± 0.07
Mixed	35	26	0.15	46.4	71.63 ± 0.65	9.13 ± 0.65	68.63 ± 0.80	12.13 ± 0.80
Mixed	69	48	0.07	75.0	54.82 ± 1.57	25.94 ± 1.57	38.46 ± 3.59	42.30 ± 3.59

Table 29: CIFAR-10 (VGG16-BN) compression. Original: 92.11%, 15.25M params, 629.66M FLOPs. Because VGG16 lacks residual connections, scale factors are strictly required to prevent signal collapse, preserving over 38% absolute accuracy at extreme compression. Results averaged across five runs $\pm$ standard deviation (omitted for deterministic pure Pruning).

						With Scale		No Scale	
Strategy	M	P	Params (M)	FLOPs (M)	Red. (%)	Acc (%) ↑	Drop ↓	Acc (%) ↑	Drop ↓
Original	–	–	15.25	629.66	–	92.11	–	92.11	–
Pruning	0	217	13.78	564.71	9.6	91.44	0.67	91.44	0.67
Pruning	0	1536	5.68	454.69	62.8	92.11	0.00	92.11	0.00
Clustering	2397	0	2.74	333.27	82.0	91.03 ± 0.03	1.08 ± 0.03	91.13 ± 0.12	0.98 ± 0.12
Clustering	2613	0	2.18	210.40	85.7	87.62 ± 0.30	4.49 ± 0.30	83.83 ± 0.96	8.28 ± 0.96
Mixed	406	768	9.19	244.20	39.7	73.89 ± 1.79	18.22 ± 1.79	34.88 ± 4.33	57.23 ± 4.33
Mixed	2331	102	2.69	329.07	82.4	91.30 ± 0.10	0.81 ± 0.10	91.23 ± 0.08	0.88 ± 0.08
Mixed	2438	277	1.98	214.52	87.0	85.14 ± 1.15	6.97 ± 1.15	82.87 ± 1.27	9.24 ± 1.27
Mixed	2596	508	1.24	108.16	91.9	66.94 ± 1.89	25.17 ± 1.89	29.74 ± 3.89	62.37 ± 3.89
Mixed	2650	475	1.22	112.87	92.0	70.46 ± 1.67	21.65 ± 1.67	32.06 ± 3.67	60.05 ± 3.67

D THEORETICAL PRELIMINARIES OF TROPICAL GEOMETRY

In this section we present some fundamental properties of tropical geometry that were derived from (Maragos et al., 2021). We choose some of the theoretical insights that have been used in our proposed algorithms. In addition, we present some key definitions and proofs related to the Hausdorff distance König (2014); Haddad & Halder (2023) that has been used in our approach as well.

D.1 TROPICAL SEMIRINGS, ZONOTOPES, AND POLYTOPES

TROPICAL SEMIRINGS AND MAX-PLUS ALGEBRA

The **tropical semiring** is an algebraic structure $(\mathbb{R} \cup \{-\infty\}, \oplus, \otimes)$ where the operations are defined as follows:

- Tropical addition: $a \oplus b = \max(a, b)$
- Tropical multiplication: $a \otimes b = a + b$

This is also known as the **max-plus semiring**. There is a dual **min-plus semiring** where addition is replaced by $\min$.

Tropical algebra is idempotent: $a \oplus a = a$, and has applications in optimization, automata theory, and dynamic programming. The term "tropical" was coined to honor Imre Simon, a pioneer of min-plus algebra.

TROPICAL POLYNOMIALS AND TROPICAL GEOMETRY

A **tropical polynomial** in variables $x_1, \ldots, x_n$ is an expression of the form

$$p(x) = \max_{k=1}^{K} \left(\langle a_k, x \rangle + b_k \right)$$

where $a_k \in \mathbb{R}^n$, $b_k \in \mathbb{R}$, and $\langle \cdot, \cdot \rangle$ denotes the usual inner product. The graph of a tropical polynomial is a piecewise-linear, convex function.

The **tropical hypersurface** (or curve) associated with $p(x)$ is the set of points where the maximum is attained by at least two terms. This set forms the "decision boundary" in tropical geometry, and corresponds to the regions where the function is not differentiable.

NEWTON POLYTOPES AND TROPICAL DUALITY

The **Newton polytope** of a tropical polynomial $p(x)$ is the convex hull of the exponent vectors a_k appearing in the terms of $p(x)$. The **extended Newton polytope** includes the biases b_k as an extra coordinate:

$$\mathrm{ENewt}(p) = \mathrm{conv}\left\{(a_k, b_k) : k = 1, \ldots, K\right\}$$

There is a duality between the tropical curve (the set where the max is attained at least twice) and the upper faces of the Newton polytope.

TROPICAL HALFSPACES, POLYHEDRA, AND POLYTOPES

A **tropical halfspace** is defined as the set of points $x \in \mathbb{R}^n$ satisfying

$$\max_{i \in S}(x_i + a_i) \geq \max_{j \in T}(x_j + a_j)$$

for some partition $S \cup T$ of $\{1, \ldots, n\}$ and constants a_i. A **tropical polyhedron** is the intersection of finitely many tropical halfspaces. A **tropical polytope** is a bounded tropical polyhedron.

ZONOTOPES

A **zonotope** is a special type of a polytope that can be written as the Minkowski sum of line segments (generators). In $\mathbb{R}^n$, a zonotope Z generated by vectors $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m$ is

$$Z = \left\{ \sum_{i=1}^{m} \lambda_i \boldsymbol{g}_i : \lambda_i \in [0, 1] \right\}$$

Zonotopes arise naturally in tropical geometry as Newton polytopes of tropical polynomials, and are important in neural network analysis, reachability, and approximation.

Minkowski Sum and Polytope Approximation

The **Minkowski sum** of two sets $A, B \subset \mathbb{R}^n$ is

$$A \oplus B = \{a + b : a \in A, b \in B\}$$

Zonotopes are closed under Minkowski addition. Approximating general polytopes by zonotopes is a key problem in tropical neural network compression and geometric approximation.

E Mathematical Preliminaries

E.1 Support Functions and Hausdorff Distance

Definition E.1 (Support Function)**.** For a compact convex set $C \subset \mathbb{R}^n$, the support function $h_C : \mathbb{R}^n \to \mathbb{R}$ is defined as:

$$h_C(\boldsymbol{u}) = \sup_{\boldsymbol{x} \in C} \langle \boldsymbol{u}, \boldsymbol{x} \rangle$$

Theorem E.2 (Hausdorff Distance Identity)**.** *Let A and B be compact convex sets in $\mathbb{R}^n$. The Hausdorff distance $d_H(A, B)$ is given by the uniform norm of the difference of their support functions on the unit sphere:*

$$d_H(A, B) = \sup_{\|\boldsymbol{u}\|=1} |h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})|$$

Proof. Hausdorff distance can be written in terms of their support functions König (2014).

Let $A, B \subset \mathbb{R}^n$ be compact convex sets. The Hausdorff distance is

$$d_H(A, B) = \max\{h_+(A, B),\ h_+(B, A)\}$$

where

$$h_+(A, B) = \inf\{\, \varepsilon > 0 : A \subseteq B + \mathbb{B}_\varepsilon \,\}$$

and $\mathbb{B}_\varepsilon$ is the closed ball of radius ε centered at the origin.

Support Function: For any convex set K,

$$h_K(\boldsymbol{u}) = \sup_{\boldsymbol{x} \in K} \langle \boldsymbol{u}, \boldsymbol{x} \rangle$$

for all $\boldsymbol{u} \in \mathbb{R}^n$.

Minkowski Sum: The support function of a Minkowski sum satisfies:

$$h_{B+\mathbb{B}_\varepsilon}(\boldsymbol{u}) = h_B(\boldsymbol{u}) + \varepsilon\|\boldsymbol{u}\|$$

If $\|\boldsymbol{u}\| = 1$, this becomes $h_B(\boldsymbol{u}) + \varepsilon$.

Inclusion Condition: $A \subseteq B + \mathbb{B}_\varepsilon$ if and only if

$$h_A(\boldsymbol{u}) \leq h_B(\boldsymbol{u}) + \varepsilon \qquad \forall \|\boldsymbol{u}\| = 1$$

Therefore,

$$h_+(A, B) = \sup_{\|\boldsymbol{u}\|=1} [h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})]$$

and similarly,

$$h_+(B, A) = \sup_{\|\boldsymbol{u}\|=1} [h_B(\boldsymbol{u}) - h_A(\boldsymbol{u})]$$

Final Formula:

$$\begin{aligned} d_H(A, B) &= \max\left\{ \sup_{\|\boldsymbol{u}\|=1} [h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})],\ \sup_{\|\boldsymbol{u}\|=1} [h_B(\boldsymbol{u}) - h_A(\boldsymbol{u})] \right\} \\ &= \sup_{\|\boldsymbol{u}\|=1} |h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})| \end{aligned}$$

Thus, for compact convex sets, the Hausdorff distance is

$$d_H(A, B) = \sup_{\|\boldsymbol{u}\|=1} |h_A(\boldsymbol{u}) - h_B(\boldsymbol{u})|$$

□

1. Homogeneity of the Support Function

For any convex compact set $P \subset \mathbb{R}^n$, the support function is

$$h_P(\boldsymbol{u}) = \sup_{\boldsymbol{x} \in P} \langle \boldsymbol{u}, \boldsymbol{x} \rangle$$

and for any scalar $\lambda \geq 0$,

$$h_P(\lambda \boldsymbol{u}) = \lambda h_P(\boldsymbol{u})$$

This is called *positive homogeneity of degree 1*.

2. Implication: Only Direction Matters

Suppose we want to maximize $|h_P(\boldsymbol{u}) - h_Q(\boldsymbol{u})|$ over all nonzero $\boldsymbol{u}$:

Let $\boldsymbol{u} \neq 0$ and write $\boldsymbol{u} = \|\boldsymbol{u}\| \cdot \hat{\boldsymbol{u}}$ with $\|\hat{\boldsymbol{u}}\| = 1$. Then,

$$h_P(\boldsymbol{u}) = h_P(\|\boldsymbol{u}\|\hat{\boldsymbol{u}}) = \|\boldsymbol{u}\| h_P(\hat{\boldsymbol{u}})$$

and similarly for Q.

So,

$$|h_P(\boldsymbol{u}) - h_Q(\boldsymbol{u})| = \|\boldsymbol{u}\| \cdot |h_P(\hat{\boldsymbol{u}}) - h_Q(\hat{\boldsymbol{u}})|$$

If we let $\|\boldsymbol{u}\| \to \infty$, this can be made arbitrarily large unless we restrict $\|\boldsymbol{u}\| = 1$. Thus, maximizing over all $\boldsymbol{u} \neq 0$ is equivalent to maximizing over the unit sphere Haddad & Halder (2023).

3. Unit Sphere Contains All Directions

The unit sphere $S^{n-1} = \{\boldsymbol{u} \in \mathbb{R}^n : \|\boldsymbol{u}\| = 1\}$ contains every possible direction in $\mathbb{R}^n$, just scaled to length 1.

4. Conclusion

$$\sup_{\boldsymbol{u} \neq 0} \frac{|h_P(\boldsymbol{u}) - h_Q(\boldsymbol{u})|}{\|\boldsymbol{u}\|} = \sup_{\|\boldsymbol{u}\|=1} |h_P(\boldsymbol{u}) - h_Q(\boldsymbol{u})|$$

So, for Hausdorff distance and support functions, maximizing over the unit sphere is both necessary and sufficient.

Summary: Because the support function is positively homogeneous, only the direction of $\boldsymbol{u}$ matters. The unit sphere S^{n-1} covers all directions, so we restrict to $\|\boldsymbol{u}\| = 1$ without loss of generality.

E.2 Zonotopes on the Interval $[0, 1]$

Line Segments

Let us define a line segment generated by a vector $\boldsymbol{w} \in \mathbb{R}^n$ based on the interval $[0, 1]$.

$$S = \{\lambda \boldsymbol{w} : \lambda \in [0, 1]\} = [0, \boldsymbol{w}]$$

Support Function of a Segment on $[0, 1]$

The support function of the segment $S = [0, \boldsymbol{w}]$ is defined as:

$$h_S(\boldsymbol{u}) = \max(0, \langle \boldsymbol{u}, \boldsymbol{w} \rangle)$$

Proof. By definition:

$$h_S(\boldsymbol{u}) = \sup_{\lambda \in [0,1]} \langle \boldsymbol{u}, \lambda \boldsymbol{w} \rangle = \sup_{\lambda \in [0,1]} \lambda \langle \boldsymbol{u}, \boldsymbol{w} \rangle$$

We consider two cases for the scalar value $c = \langle \boldsymbol{u}, \boldsymbol{w} \rangle$:

- **Case 1 ($c > 0$):** The function $f(\lambda) = \lambda c$ is increasing. The maximum is at $\lambda = 1$. Thus, $h_S(\boldsymbol{u}) = 1 \cdot c = \langle \boldsymbol{u}, \boldsymbol{w} \rangle$.
- **Case 2 ($c \leq 0$):** The function $f(\lambda) = \lambda c$ is non-increasing. The maximum is at $\lambda = 0$. Thus, $h_S(\boldsymbol{u}) = 0 \cdot c = 0$.

Combining these, $h_S(\boldsymbol{u}) = \max(0, \langle \boldsymbol{u}, \boldsymbol{w} \rangle)$. □

F Application to Neural Pruning

F.1 The Neural Layer as a Zonotope

Consider a layer with weight matrix W (rows $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$) receiving input $\boldsymbol{x}$ where $x_i \in [0, 1]$ (e.g., outputs of a Sigmoid or bounded ReLU function). The set of reachable outputs is the zonotope:

$$Z_{\text{layer}} = \sum_{i=1}^{k} [0, \boldsymbol{w}_i]$$

Using the additivity proposition, the support function of the layer is:

$$h_{Z_{\text{layer}}}(\boldsymbol{u}) = \sum_{i=1}^{k} h_{[0,\boldsymbol{w}_i]}(\boldsymbol{u}) = \sum_{i=1}^{k} \max(0, \langle \boldsymbol{u}, \boldsymbol{w}_i \rangle)$$

F.2 Compression Error Bound

Definitions:

$$Z = \bigoplus_{i=1}^{n} [0, \mathbf{w}_i^{\text{out}}] = \bigoplus_{k=1}^{K} Z_k, \qquad Z_k = \bigoplus_{i \in C_k} [0, \mathbf{w}_i^{\text{out}}]$$

$$\rho_i = \|\mathbf{w}_i^{\text{out}}\|_2, \qquad \mathbf{v}_i = \frac{\mathbf{w}_i^{\text{out}}}{\rho_i}, \qquad S_k = \sum_{i \in C_k} \rho_i$$

$$\bar{\mathbf{w}}_k^{\text{out}} = \sum_{i \in C_k} \mathbf{w}_i^{\text{out}}, \qquad M_k = \|\bar{\mathbf{w}}_k^{\text{out}}\|_2, \qquad \mathbf{v}_k^\star = \frac{\bar{\mathbf{w}}_k^{\text{out}}}{M_k}$$

$$\alpha_k^{\text{out}} = \frac{S_k}{M_k + \varepsilon}, \qquad \hat{\mathbf{w}}_k^{\text{out}} = \alpha_k^{\text{out}} \bar{\mathbf{w}}_k^{\text{out}} = \frac{S_k M_k}{M_k + \varepsilon} \mathbf{v}_k^\star =: \tilde{S}_k \mathbf{v}_k^\star$$

$$Z' = \bigoplus_{k=1}^{K} \left[0,\, \alpha_k^{\text{in}} \hat{\mathbf{w}}_k^{\text{out}}\right] = \bigoplus_{k=1}^{K} \left[0,\, \alpha_k^{\text{in}} \tilde{S}_k \mathbf{v}_k^\star\right]$$

Step 1: Global Cluster Decomposition

$$h_Z(\mathbf{u}) - h_{Z'}(\mathbf{u}) = \sum_{k=1}^{K} \left(h_{Z_k}(\mathbf{u}) - h_{\left[0, \alpha_k^{\text{in}} \tilde{S}_k \mathbf{v}_k^\star\right]}(\mathbf{u}) \right)$$

$$|h_Z(\mathbf{u}) - h_{Z'}(\mathbf{u})| \leq \sum_{k=1}^{K} \left| h_{Z_k}(\mathbf{u}) - h_{\left[0, \alpha_k^{\text{in}} \tilde{S}_k \mathbf{v}_k^\star\right]}(\mathbf{u}) \right|$$

$$\sup_{\|\mathbf{u}\|_2 = 1} |h_Z(\mathbf{u}) - h_{Z'}(\mathbf{u})| \leq \sum_{k=1}^{K} \sup_{\|\mathbf{u}\|_2 = 1} \left| h_{Z_k}(\mathbf{u}) - h_{\left[0, \alpha_k^{\text{in}} \tilde{S}_k \mathbf{v}_k^\star\right]}(\mathbf{u}) \right|$$

$$d_H(Z, Z') \leq \sum_{k=1}^{K} d_H \left(Z_k, \left[0, \alpha_k^{\text{in}} \tilde{S}_k \mathbf{v}_k^\star\right] \right)$$

Step 2: Triangle Inequality on Clusters

$$d_H\left(Z_k, \left[0, \alpha_k^{\text{in}}\tilde{S}_k\mathbf{v}_k^\star\right]\right) \le \underbrace{d_H\left(Z_k, [0, S_k\mathbf{v}_k^\star]\right)}_{\text{(I)}} + \underbrace{d_H\left([0, S_k\mathbf{v}_k^\star], [0, \tilde{S}_k\mathbf{v}_k^\star]\right)}_{\text{(II)}} + \underbrace{d_H\left([0, \tilde{S}_k\mathbf{v}_k^\star], \left[0, \alpha_k^{\text{in}}\tilde{S}_k\mathbf{v}_k^\star\right]\right)}_{\text{(III)}}$$

Step 3: Evaluation of Term (II)

$$h_{[0,S_k\mathbf{v}_k^\star]}(\mathbf{u}) - h_{[0,\tilde{S}_k\mathbf{v}_k^\star]}(\mathbf{u}) = \max(0, \langle\mathbf{u}, S_k\mathbf{v}_k^\star\rangle) - \max(0, \langle\mathbf{u}, \tilde{S}_k\mathbf{v}_k^\star\rangle) = (S_k - \tilde{S}_k)\max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)$$

$$\text{(II)} = \sup_{\|\mathbf{u}\|_2=1} |(S_k - \tilde{S}_k)\max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)| = S_k - \tilde{S}_k$$

$$S_k - \tilde{S}_k = S_k - \frac{S_k M_k}{M_k + \varepsilon} = \frac{\varepsilon S_k}{M_k + \varepsilon}$$

Step 4: Evaluation of Term (III)

$$h_{\left[0,\alpha_k^{\text{in}}\tilde{S}_k\mathbf{v}_k^\star\right]}(\mathbf{u}) - h_{[0,\tilde{S}_k\mathbf{v}_k^\star]}(\mathbf{u}) = (\alpha_k^{\text{in}} - 1)\tilde{S}_k \max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)$$

$$\text{(III)} = \sup_{\|\mathbf{u}\|_2=1} |(\alpha_k^{\text{in}} - 1)\tilde{S}_k \max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)| = (\alpha_k^{\text{in}} - 1)\tilde{S}_k$$

Step 5: Evaluation of Term (I)

$$S_k = \sum_{i\in C_k}\rho_i \implies h_{[0,S_k\mathbf{v}_k^\star]}(\mathbf{u}) = \sum_{i\in C_k}\rho_i \max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)$$

$$h_{Z_k}(\mathbf{u}) - h_{[0,S_k\mathbf{v}_k^\star]}(\mathbf{u}) = \sum_{i\in C_k}\rho_i\Big[\max(0, \langle\mathbf{u}, \mathbf{v}_i\rangle) - \max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)\Big]$$

$$|\max(0, \langle\mathbf{u}, \mathbf{v}_i\rangle) - \max(0, \langle\mathbf{u}, \mathbf{v}_k^\star\rangle)| \le |\langle\mathbf{u}, \mathbf{v}_i - \mathbf{v}_k^\star\rangle| \le \|\mathbf{u}\|_2\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2$$

$$\|\mathbf{u}\|_2 = 1 \implies \text{(I)} \le \sum_{i\in C_k}\rho_i\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2$$

$$\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2^2 = \|\mathbf{v}_i\|_2^2 - 2\langle\mathbf{v}_i, \mathbf{v}_k^\star\rangle + \|\mathbf{v}_k^\star\|_2^2 = 2 - 2\langle\mathbf{v}_i, \mathbf{v}_k^\star\rangle$$

$$\sum_{i\in C_k}\rho_i\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2^2 = 2\sum_{i\in C_k}\rho_i - 2\left\langle\sum_{i\in C_k}\rho_i\mathbf{v}_i, \mathbf{v}_k^\star\right\rangle = 2S_k - 2\langle\bar{\mathbf{w}}_k^{\text{out}}, \mathbf{v}_k^\star\rangle$$

$$\langle\bar{\mathbf{w}}_k^{\text{out}}, \mathbf{v}_k^\star\rangle = \left\langle\bar{\mathbf{w}}_k^{\text{out}}, \frac{\bar{\mathbf{w}}_k^{\text{out}}}{M_k}\right\rangle = \frac{\|\bar{\mathbf{w}}_k^{\text{out}}\|_2^2}{M_k} = \frac{M_k^2}{M_k} = M_k$$

$$\sum_{i\in C_k}\rho_i\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2^2 = 2(S_k - M_k)$$

$$\sum_{i\in C_k}\rho_i\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2 = \sum_{i\in C_k}\sqrt{\rho_i}\left(\sqrt{\rho_i}\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2\right) \le \sqrt{\sum_{i\in C_k}\rho_i}\sqrt{\sum_{i\in C_k}\rho_i\|\mathbf{v}_i - \mathbf{v}_k^\star\|_2^2}$$

$$\text{(I)} \le \sqrt{S_k}\sqrt{2(S_k - M_k)} = S_k\sqrt{2\left(1 - \frac{M_k}{S_k}\right)}$$

$$\alpha_k^{\text{out}} = \frac{S_k}{M_k + \varepsilon} \implies \frac{M_k}{S_k} = \frac{1}{\alpha_k^{\text{out}}} - \frac{\varepsilon}{S_k}$$

$$\text{(I)} \le S_k\sqrt{2\left(1 - \frac{1}{\alpha_k^{\text{out}}} + \frac{\varepsilon}{S_k}\right)}$$

Final Bound:

$$d_H(Z, Z') \le \sum_{k=1}^{K}\left[S_k\sqrt{2\left(1 - \frac{1}{\alpha_k^{\text{out}}} + \frac{\varepsilon}{S_k}\right)} + \frac{\varepsilon S_k}{M_k + \varepsilon} + (\alpha_k^{\text{in}} - 1)\tilde{S}_k\right]$$

$$\lim_{\varepsilon\to 0} d_H(Z,Z') \le \sum_{k=1}^{K}\left[S_k\sqrt{2\left(1-\frac{1}{\alpha_k^{\text{out}}}\right)}+(\alpha_k^{\text{in}}-1)S_k\right] = \sum_{k=1}^{K}\left[\sqrt{2S_k(S_k-M_k)}+(\alpha_k^{\text{in}}-1)S_k\right]$$

Pruning Error Bound

Pruning a set $P \subseteq \{1,\dots,n\}$ of neurons removes the segments $S_i = [0, \boldsymbol{w}_i]$. Let Z be the original zonotope $Z_P = \bigoplus_{i\in P}[0,\boldsymbol{w}_i]$ and $Z_{\setminus i}$ be the pruned zonotope.By additivity of the support function and the fact that, for any compact convex set K, $\sup_{\|\boldsymbol{u}\|=1} h_K(\boldsymbol{u}) = \max_{\boldsymbol{z}\in K}\|\boldsymbol{z}\|_2$.

$$d_H(Z,Z_{\setminus P}) = \sup_{\|\boldsymbol{u}\|=1}\left|h_Z(\boldsymbol{u})-h_{Z_{\setminus P}}(\boldsymbol{u})\right| = \sup_{\|\boldsymbol{u}\|=1}\sum_{i\in P}\max(0,\langle\boldsymbol{u},\boldsymbol{w}_i\rangle)$$

$$= \max_{\boldsymbol{z}\in Z_P}\|\boldsymbol{z}\|_2 = \max_{\boldsymbol{\varepsilon}\in\{0,1\}^{|P|}}\left\|\sum_{i\in P}\varepsilon_i\boldsymbol{w}_i\right\|_2 \le \sum_{i\in P}\|\boldsymbol{w}_i\|_2.$$

$$d_H(Z,Z_{\setminus P}) \le \sum_{i\in P}\|\boldsymbol{w}_i\|_2$$

Pruning a single neuron i removes the segment $S_i = [0,\boldsymbol{w}_i]$. Let Z be the original zonotope and $Z_{\setminus i}$ be the pruned zonotope. Using the Hausdorff identity derived in Theorem 1:

$$d_H(Z,Z_{\setminus i}) = \sup_{\|\boldsymbol{u}\|=1}|h_Z(\boldsymbol{u})-h_{Z_{\setminus i}}(\boldsymbol{u})| = \sup_{\|\boldsymbol{u}\|=1}\max(0,\langle\boldsymbol{u},\boldsymbol{w}_i\rangle) = \|\boldsymbol{w}_i\|_2.$$

The pruning error equals the Euclidean norm of the removed generator.

G Scale Factor Properties

Two scale factors are applied after merging: a variance-preserving factor α_k^{in} for the input weights and an energy-preserving factor α_k^{out} for the output weights.

Input scale factor (variance preservation). Averaging the input weights of a cluster reduces their variance. The factor α_k^{in} (Eq. 6) restores the standard deviation of the merged vector to match that of the original cluster members:

$$\sigma\big(\alpha_k^{\text{in}}\cdot\bar{\boldsymbol{w}}_k^{\text{in}}\big) = \alpha_k^{\text{in}}\cdot\sigma(\bar{\boldsymbol{w}}_k^{\text{in}}) = \frac{\sigma(\{\boldsymbol{w}_i^{\text{in}}\}_{i\in\mathcal{C}_k})}{\sigma(\bar{\boldsymbol{w}}_k^{\text{in}})+\varepsilon}\cdot\sigma(\bar{\boldsymbol{w}}_k^{\text{in}}) \approx \sigma\big(\{\boldsymbol{w}_i^{\text{in}}\}_{i\in\mathcal{C}_k}\big).$$

The approximation becomes exact as $\varepsilon\to 0$.

Output scale factor (energy preservation). The factor α_k^{out} (Eq. 6) ensures that the Euclidean norm of the merged output weight vector equals the sum of norms of the original cluster members:

Proposition G.1. *Let* $\bar{\boldsymbol{w}}_k^{\text{out}} = \sum_{i\in\mathcal{C}_k}\boldsymbol{w}_i^{\text{out}}$. *Then* $\hat{\boldsymbol{w}}_k^{\text{out}} = \alpha_k^{\text{out}}\cdot\bar{\boldsymbol{w}}_k^{\text{out}}$ *satisfies* $\|\hat{\boldsymbol{w}}_k^{\text{out}}\|_2 = \sum_{i\in\mathcal{C}_k}\|\boldsymbol{w}_i^{\text{out}}\|_2$.

Proof. Direct substitution of α_k^{out}:

$$\|\hat{\boldsymbol{w}}_k^{\text{out}}\|_2 = \alpha_k^{\text{out}}\cdot\|\bar{\boldsymbol{w}}_k^{\text{out}}\|_2 = \frac{\sum_{i\in\mathcal{C}_k}\|\boldsymbol{w}_i^{\text{out}}\|_2}{\|\bar{\boldsymbol{w}}_k^{\text{out}}\|_2+\varepsilon}\cdot\|\bar{\boldsymbol{w}}_k^{\text{out}}\|_2 \approx \sum_{i\in\mathcal{C}_k}\|\boldsymbol{w}_i^{\text{out}}\|_2. \qquad \square$$

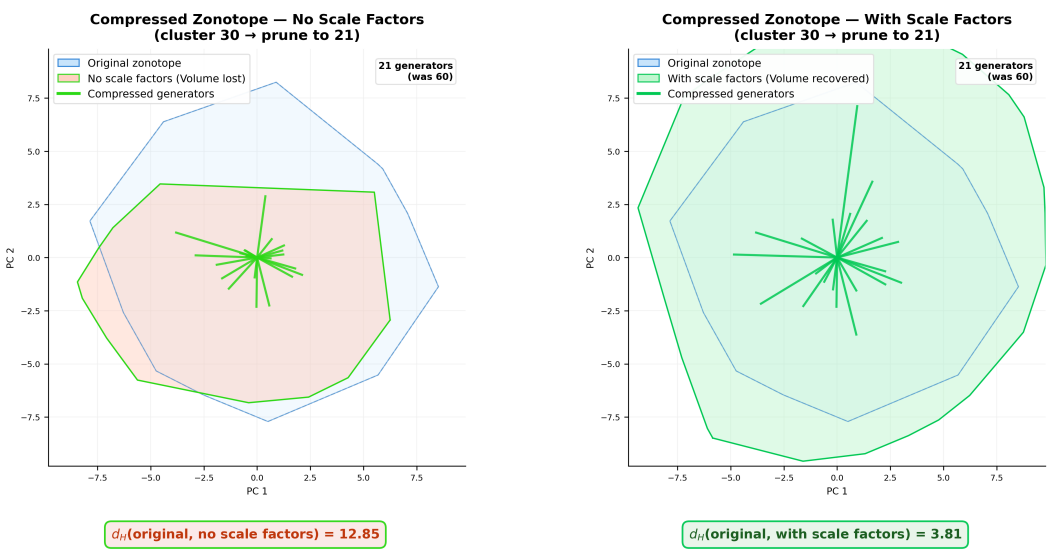


Figure 8: Effect of the scale factors on the compressed layer zonotope (FC layer 0, FashionMNIST, 60 generators clustered to 30 and pruned to 21, projected to 2D via PCA). The original zonotope is shown as a blue wireframe in both panels. **Left:** without scale factors ($\alpha_{\text{in}} = \alpha_{\text{out}} = 1$); the compressed zonotope (orange) is significantly smaller than the original ($d_H = 12.85$). **Right:** with both scale factors applied; the compressed zonotope (green) closely matches the original volume ($d_H = 3.81$), demonstrating that the scale factors effectively compensate for the energy lost during averaging.

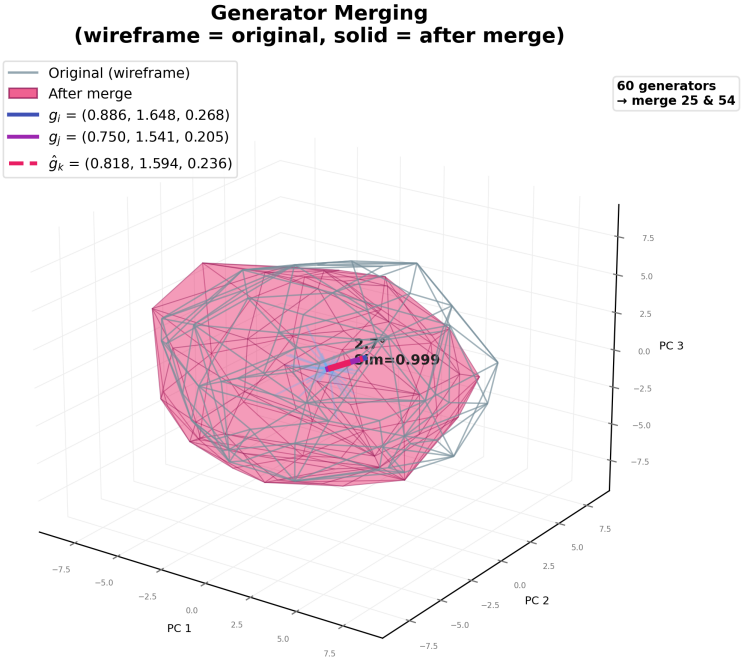


Figure 9: Generator merging illustrated on the layer zonotope of FC layer 0 (FashionMNIST, generators projected to 3D by PCA). The two most similar generators g_i and g_j (angle 2.7°, cosine similarity 0.999) are merged into $\hat{g}_k$ (dashed). The minimal shape change confirms that merging nearly parallel generators introduces negligible geometric error.