Integrating Independent Layer-Wise Rank Selection with Low-Rank SVD Training for Model Compression: A Theory-Driven Approach

Yifan Guo[1] and Alyssa Yu[2]

[1] Towson University, Towson, Maryland, USA

[2] Poolesville High School, Poolesville, Maryland, USA

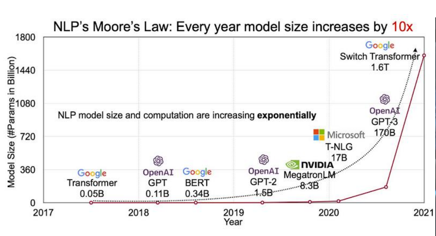
Introduction

Need for Model Compression

- In recent years, neural networks have grown drastically in complexity with billions or trillions of parameters.
- Reducing model size is crucial for scenarios demanding fast re-training and inference on resource-constrained edge devices.

Uses of Model Compression

- Large language models require substantial computing resources to accommodate for their large model sizes
- Model compression allows for fast real-time decision making in self-driving cars.



Harishdatalab. (2023, June 15). The Increasing Size of LLM Models. https://medium.com/@harishdatalab/unveiling-the-power-of-large-language-models-llms-e235c4eba8a9.

Edmonds, E. (2019, March 14). https://newsroom.aaa.com/wp-content/uploads/2019/03/av-report-i692832436.jpg

Low-Rank Factorization and Truncation

Given $\mathbf{W}$ an $m \times n$ matrix, its Singular Value Decomposition (SVD) is:

$$\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top,$$

- $\boldsymbol{U}$ is an orthogonal $m \times m$ matrix whose columns are the left singular vectors.
- $\boldsymbol{\Sigma}$ is an $m \times n$ diagonal matrix with singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$ on the diagonal (denoted as singular values and r is the rank of $\boldsymbol{W}$ and $r \leq \min\{m, n\}$.) and the rest entries are all zero.
- $\boldsymbol{V}^\top$ is the transpose of an orthogonal $n \times n$ matrix whose rows are the right singular vectors.

$$\boldsymbol{W} = \underbrace{\begin{bmatrix} | & & | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_k & \cdots & \mathbf{u}_r \\ | & & | & & | \end{bmatrix}}_{\mathbf{U} \in \mathbb{R}^{m\times r}} \underbrace{\begin{bmatrix} \sigma_1 & & & & \\ & \ddots & & & \\ & & \sigma_k & & \\ & & & \ddots & \\ & & & & \sigma_r \end{bmatrix}}_{\boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}} \underbrace{\begin{bmatrix} - & \mathbf{v}_1^\top & - \\ & \vdots & \\ - & \mathbf{v}_k^\top & - \\ & \vdots & \\ - & \mathbf{v}_r^\top & - \end{bmatrix}}_{\boldsymbol{V}^\top \in \mathbb{R}^{r\times n}}$$

Truncation: The full-rank matrix $\boldsymbol{W}$ has rank $r = \min\{m, n\}$. With a chosen $k \leq r$, we obtain the corresponding low-rank matrix $\boldsymbol{W}_k = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^T$, where $\boldsymbol{U}_k, \boldsymbol{\Sigma}_k, \boldsymbol{V}_k$ are the top-k components truncated from $\boldsymbol{U}, \boldsymbol{\Sigma}$, and $\boldsymbol{V}$.

$$\boldsymbol{W}_k = \underbrace{\begin{bmatrix} | & & | \\ \mathbf{u}_1 & \cdots & \mathbf{u}_k \\ | & & | \end{bmatrix}}_{\boldsymbol{U}_k \in \mathbb{R}^{m\times k}} \underbrace{\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix}}_{\boldsymbol{\Sigma}_k \in \mathbb{R}^{k\times k}} \underbrace{\begin{bmatrix} - & \mathbf{v}_1^\top & - \\ & \vdots & \\ - & \mathbf{v}_k^\top & - \end{bmatrix}}_{\boldsymbol{V}_k^\top \in \mathbb{R}^{k\times n}}$$

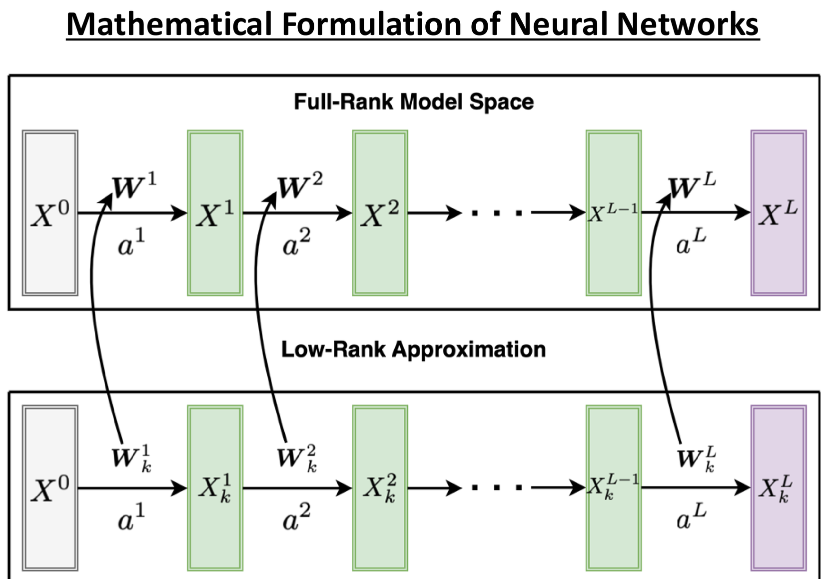
Framework

Uniqueness of Our Approach

- Low-rank training incorporates various penalty terms in the loss function to reduce the rank of weight matrices while preserving high accuracy.
- Three aspects considered in optimal rank selection:
  - **Independent** vs. Dependent Layer-Wise Rank Selection
  - **Theory-Driven** vs. Heuristic Rank Search Strategy
  - **During-Training** vs. Post-Training Rank Selection.
- Our approach includes independent layer-wise rank selection, is theory-driven, and conducts rank selection during training; no previous works include *all* three optimal aspects.

Approach	Layer-Wise Rank Selection (Independent vs. Dependent)	Search Strategy (Theory-Driven vs. Heuristic)	Rank Selection Timing (During- vs. Post-Training)
[Cheng *et al.*, 2020]	Dependent	Heuristic	Post-training
[Kim *et al.*, 2020]	Dependent	Theory-Driven	Post-Training
[Eo *et al.*, 2021]	Independent	Heuristic	Post-Training
[Sobolev *et al.*, 2022]	Dependent	Heuristic	Post-Training
[Xiao *et al.*, 2023]	Dependent	Heuristic	Post-Training
[Wang *et al.*, 2023]	Independent	Heuristic	During-Training
[Cao *et al.*, 2024]	Dependent	Heuristic	Post-Training
Ours	Independent	Theory-Driven	During-Training

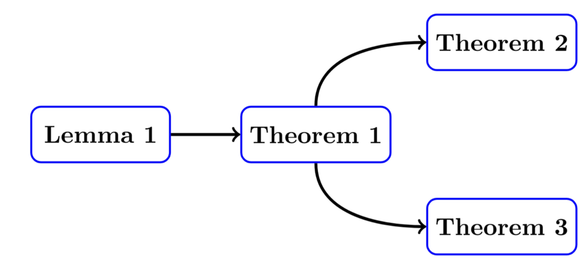
Mathematical Formulation of Neural Networks



- Neural network of L layers denoted by $f_{\mathcal{W}}(\cdot)$ and parameterized by $\mathcal{W} = \{\boldsymbol{W}^1, \boldsymbol{W}^2, \ldots, \boldsymbol{W}^L\}$.
- $X^l = a^l(\boldsymbol{W}^l X^{l-1})$ for $l = 1, 2, \ldots, L$, where $\boldsymbol{W}^l$, a^l, and X^l represent the weight matrix, non-linear activation function, and output of the l-th layer.

Our Theocratical Findings

Research Problem: Design a theory-driven approach to search for truncations $(W_k^1, \ldots, W_k^L)$ of $(W^1, \ldots, W^L)$ with layer wise ranks $(k^1, k^2, \ldots, k^L)$ to achieve optimal compression and accuracy concurrently.



Lemma 1 (Eckart–Young–Mirsky Theorem [Golub *et al.*, 1987])**.** *Let $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ be a matrix with rank r and let $\|\cdot\|_2$ denote the spectral norm. Following the same definitions in Proposition 1, we define $\boldsymbol{W}_k$ to be the best rank-k approximation of $\boldsymbol{W}$ in the spectral norm, i.e., $\boldsymbol{W}_k = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^T = \sum_{i=1}^k \sigma_i \boldsymbol{U}_{:,i}\boldsymbol{V}_{i,:}^T$, where $\boldsymbol{U}_k, \boldsymbol{\Sigma}_k$, and $\boldsymbol{V}_k$ are the top-k components truncated from $\boldsymbol{U}, \boldsymbol{\Sigma}$, and $\boldsymbol{V}$, respectively. Then, we have $\|\boldsymbol{W} - \boldsymbol{W}_k\|_2 = \sigma_{k+1}$.*

How Low-Rank Truncation Impact the Output

Theorem 1 (The output difference bound under rank-k approximation for L-layer neural networks)**.** *Let a^l be the activation function for the l-th layer, where a^l is ρ_l-Lipschitz and satisfies $a^l(0) = 0$ for all $l \in [1, L]$. Let $\boldsymbol{W}_l$ be the full-rank matrix at layer l, and let $\boldsymbol{W}_l$ be its rank k^l approximation from keeping only the top k^l singular values in the SVD decomposition of $\boldsymbol{W}_k^l$. Let σ_i^l denote the ith singular value of $\boldsymbol{W}_l$, X^0 be the initial input vector, and X^l and X_k^l be the output vectors at the l-th layer after applying $\boldsymbol{W}^l$ and $\boldsymbol{W}_k^l$, respectively. Then, the output difference $\left\|X^L - X_k^L\right\|_2$ from a rank-k approximation over an L-layer feed-forward network is upper-bounded by $\left(\prod_{l=1}^L \rho_l\sigma_1^l\right)\left(\sum_{l=1}^L \frac{\sigma_{k^l+1}^l}{\sigma_1^l}\right)\left\|X^0\right\|_2$.*

How Low-Rank Truncation Impact the Loss Error

- Let $X_i^L = f_{\mathcal{W}}(X_i^0) \in \mathbb{R}^C$ and $X_{i,k}^L = f_{\mathcal{W}_{\boldsymbol{k}}}(X_{i,k}^0) \in \mathbb{R}^C$ be the output logits after feeding an input X_i^0 sampled from the training dataset $\mathcal{D}^{tr} = \{X_i^0, y_i\}_{i=1}^R$, from the full-rank parameter space $\mathcal{W} = \{\boldsymbol{W}^1, \boldsymbol{W}^2, \ldots, \boldsymbol{W}^L\}$ and low-rank parameter space $\mathcal{W}_{\boldsymbol{k}} = \{\boldsymbol{W}_k^1, \boldsymbol{W}_k^2, \ldots, \boldsymbol{W}_k^L\}$, respectively.
- Let B be a constant such that $\left\|X_i^0\right\|_2 \leq B$ for all $i \in [1, R]$.

Theorem 2 (The loss error bound under rank-k truncation in classification problems)**.** *Following the definitions in Theorem 1, we consider a C-class classification problem. We let $z_i = \mathtt{softmax}\left(X_i^L\right)$ and $z_{i,k} = \mathtt{softmax}\left(X_{i,k}^L\right)$, where $\mathtt{softmax}$ is the softmax function. Consider the cross-entropy function $g(z, y) = -y^T \log(z)$ and let the loss functions be $L\left(\mathcal{W}; X_i^0\right) = g(z_i, y_i)$ and $L\left(\mathcal{W}_{\boldsymbol{k}}; X_i^0\right) = g(z_{i,k}, y_i)$. We set $\sigma_{k^l+1}^l$ and δ such that $\frac{\sigma_{k^l+1}^l}{\sigma_1^l} < \delta$, $\forall\, l \in [L]$. Then, for $\forall\, \epsilon > 0$, $\exists\, \delta = \frac{\epsilon}{\sqrt{2}BL(\prod_{l=1}^L \rho_l\sigma_1^l)}$ such that $|L(\mathcal{W}; \mathcal{D}^{tr}) - L\left(\mathcal{W}_{\boldsymbol{k}}; \mathcal{D}^{tr}\right)| < \epsilon$.*

Theorem 3 (The loss error bound under rank-k truncation in regression problems)**.** *Following the definitions in Theorem 1, we consider a regression problem with loss function $g(z, y) = \|z - y\|_2$. Let $L\left(\mathcal{W}; X_i^0\right) = g\left(X_i^L, y_i\right)$ and $L\left(\mathcal{W}_{\boldsymbol{k}}; X_i^0\right) = g\left(X_{i,k}^L, y_i\right)$. We set $\sigma_{k^l+1}^l$ and δ such that $\frac{\sigma_{k^l+1}^l}{\sigma_1^l} < \delta$, $\forall\, l \in [1, L]$. Then, for $\forall\, \epsilon > 0$, $\exists\, \delta = \frac{\epsilon}{BL(\prod_{l=1}^L \rho_l\sigma_1^l)}$ such that $|L\left(\mathcal{W}; \mathcal{D}^{tr}\right) - L\left(\mathcal{W}_{\boldsymbol{k}}; \mathcal{D}^{tr}\right)| < \epsilon$.*

Our Algorithms

- Calculating the Lipschitz constant of layer-wise parameters is computationally expensive in many CNN models, preventing the extension of our theoretical findings from simple feed-forward networks to complex CNNs.
- The rank selection is integrated in low-rank SVD training.
- The low-rank SVD training loss function is

$$L(\mathcal{U}, \Sigma, \mathcal{V}; D^{tr}) = \underbrace{L_T(\mathcal{U}, \Sigma, \mathcal{V})}_{\textit{Training Loss}} + \lambda_O \underbrace{L_O(\mathcal{U}, \mathcal{V})}_{\textit{Orthogonality Loss}} + \lambda_R \underbrace{L_R(\Sigma)}_{\textit{Regularization Loss}}$$

- For the low-rank regularization loss, we consider the Nuclear norm $||\boldsymbol{W}^l||_1$ and the Hoyer norm $||\boldsymbol{W}^l||_1 / ||\boldsymbol{W}^l||_F$.

Algorithm 1: Proposed Rank Selection Algorithm

Input: full-rank parameters $\mathcal{W}$, loss error tolerance ϵ, stop searching precision $\Delta\delta$
Output: low-rank parameters $\mathcal{W}_{\boldsymbol{k}}$

```
Function RankSelection(W, ε, Δδ):
    floss ← L_T(W; D^tr);
    l ← 0; u ← 1; δ ← (l + u)/2;
    while |l − u| ≥ Δδ or |floss − loss| ≥ ε do
        W_k ← SVDLowRankApprox(W, δ);
        loss ← L_T(W_k; D^tr);
        if |floss − loss| < ε then
            l ← δ; δ ← (l + u)/2;
        else
            u ← δ; δ ← (l + u)/2;
    return W_k ;
```

Algorithm 2: Standard SVD Low-Rank Approximation

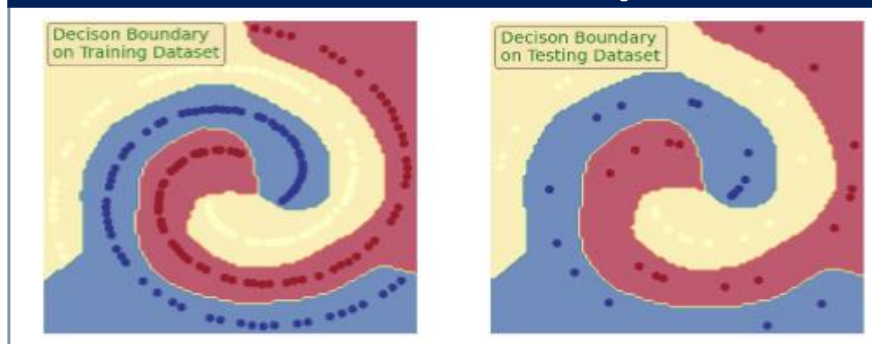
```
Function SVDLowRankApprox(W, δ):
    W_k ← ∅;
    for each W^l ∈ W do
        U^l, Σ^l, V^l ← SVD(W^l);
        k^l ← argmax_k {k | σ_k^l / σ_1^l ≥ δ};
        W_k^l ← U_k^l Σ_k^l V_k^l; W_k ← W_k ∪ W_k^l;
        // U_k^l, Σ_k^l, V_k^l are top-k^l vectors
           truncated from U^l, Σ^l, V^l
    return W_k
```

Algorithm 3: Proposed Rank Selection Enabled Low-Rank SVD Training Algorithm

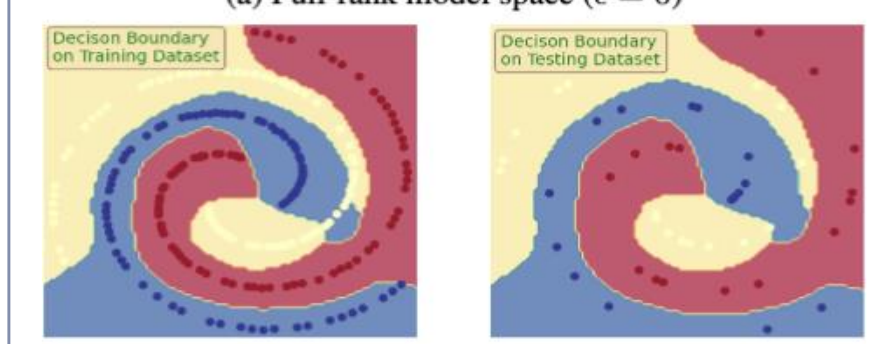
Input: full-rank parameters $\mathcal{U}, \Sigma, \mathcal{V}$, loss error tolerance ϵ, stop searching precision $\Delta\delta$, training epoch E
Output: low-rank parameters $\mathcal{U}_k, \Sigma_k, \mathcal{V}_k$

```
Initialize parameters U, Σ, V ;
e ← 1 ;
while e ≤ E do
    Update U, Σ, V based on loss function L(U, Σ, V) with an appropriate optimizer and extract the learning loss
        L_T(U, Σ, V); // L_O(U, V) and L_R(Σ) are not used in the next-step rank
        selection
    U_k, Σ_k, V_k ← RankSelection(U, Σ, V, ε, Δδ, L_T(U, Σ, V));
    U, Σ, V ← U_k, Σ_k, V_k; // Use truncated models for the next round of training
    e ← e + 1;
return U_k, Σ_k, V_k ;
```

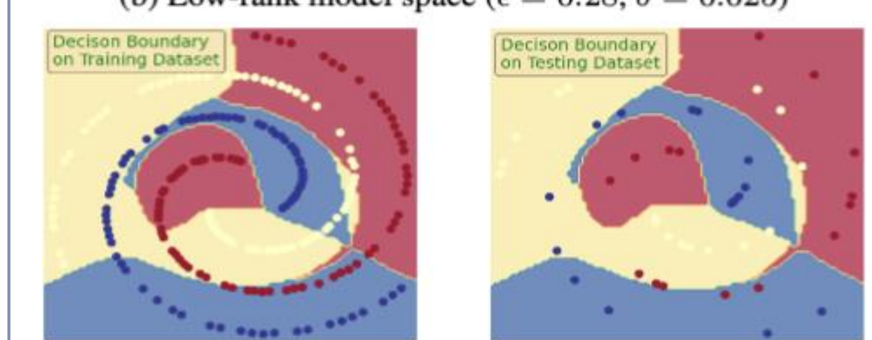
Proof-of-Concept



(a) Full-rank model space ($\epsilon = 0$)



(b) Low-rank model space ($\epsilon = 0.28$, $\delta = 0.025$)

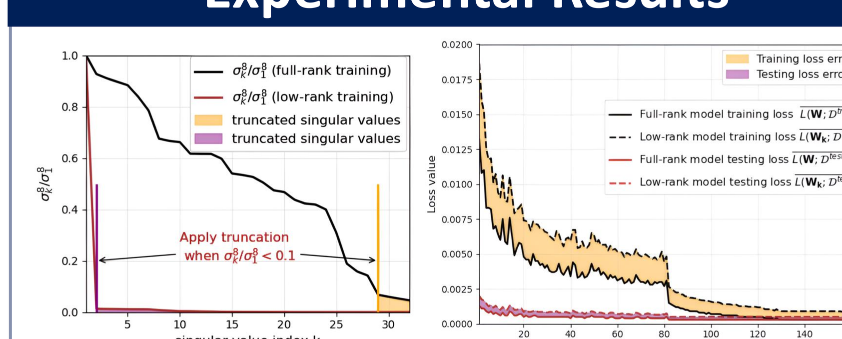


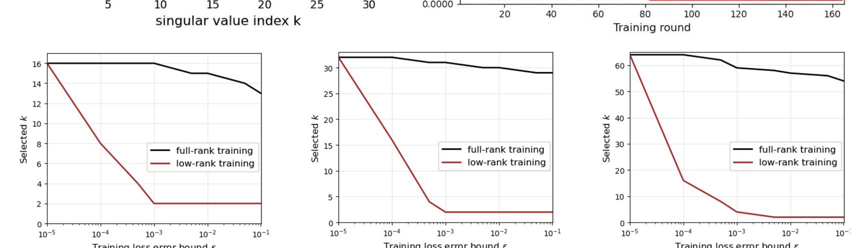
(c) Low-rank model space ($\epsilon = 0.33$, $\delta = 0.03$)

A visualization of the decision boundaries on the training dataset (left column) and testing dataset (right column) for different rank selections.

- We conduct a pilot study using a simple 3-layer feedforward neural network for a ternary classification problem.
- Validate the feasibility of identifying δ based on our derived $\epsilon - \delta$ and determining the optimal $(k^1, k^2, \ldots, k^L)$

Experimental Results





(a) The 1^{st} layer (b) The 8^{th} layer (c) The 15^{th} layer

Correlation between training loss error and rank selection on ResNet-20 model

Approach	ResNet-20		ResNet-32		ResNet-56		ResNet-110	
	Test Acc ↑	CR ↓	Test Acc ↑	CR ↓	Test Acc ↑	CR ↓	Test Acc ↑	CR ↓
[Yang *et al.*, 2020] (Channel, Squared Hoyer)	0.887	0.202	0.893	0.367	0.924	0.485	0.924	0.511
[Yang *et al.*, 2020] (Spatial, Squared Hoyer)	0.866	0.242	0.889	0.413	0.918	0.522	0.917	0.543
[Wang *et al.*, 2023]	0.822	0.337	0.834	0.398	0.844	0.441	0.849	0.462
Ours (Channel, Nuclear)	0.870	**0.155**	0.879	**0.312**	0.892	**0.422**	0.892	**0.445**
Ours (Spatial, Nuclear)	0.867	0.221	0.873	0.319	0.899	0.438	0.898	0.465
Ours (Channel, Squared Hoyer)	**0.892**	**0.155**	**0.899**	**0.312**	**0.929**	**0.422**	**0.928**	**0.445**
Ours (Spatial, Squared Hoyer)	0.873	0.221	0.881	0.319	0.912	0.438	0.912	0.465

Overall performance comparisons on CIFAR-10 dataset

Approach	ResNet-18		ResNet-50	
	Test Acc ↑	CR ↓	Test Acc ↑	CR ↓
[Yang *et al.*, 2020] (Channel, Squared Hoyer)	0.684	0.204	0.691	0.392
[Yang *et al.*, 2020] (Spatial, Squared Hoyer)	0.670	0.221	0.678	0.411
Ours (Channel, Squared Hoyer)	**0.690**	**0.181**	**0.696**	**0.366**
Ours (Spatial, Squared Hoyer)	0.672	0.207	0.681	0.398

Overall performance comparisons on ImageNet dataset

Conclusions

- In-depth theoretical analysis that quantitatively measures how low-rank approximation affects training losses.
- Rank selection enabled low-rank training inspired by our theoretical findings.
- Our algorithm, paired with channel decomposition and Hoyer regularization, achieves better results than the previous state-of-the-art studies.

Future Work

- A stricter bound of results from Theorems 1-3 can give us better control over the balance between deep learning model compression and accuracy.
- The rank-selection algorithm could be based off a generalization bound for the predicted accuracy on unseen data.
- Rank-selection-based model compression can be implemented in other models other than ResNet.

Key References

- [Idelbayev et al., 2020] Y. Idelbayev and M. Á. Carreira-Perpiñán, Low-rank compression of neural nets: Learning the rank of each layer, in Proc. of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 8046-8056.
- [Wang et al., 2023] Hongyi Wang, Saurabh Agarwal, Yoshiki Tanaka, Eric Xing, Dimitris Papailiopoulos, et al. Cuttlefish: Low-rank model training without all the tuning. In Proc. of Machine Learning and Systems, vol. 5, pp: 578–605, 2023.
- [Yang et al., 2020] H. Yang, M. Tang, W. Wen, F. Yan, D. Hu, A. Li, H. Li, Y. Chen. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp: 2899–2908.