BOLT: BUDGET-AWARE ORDERED LOW-RANK TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

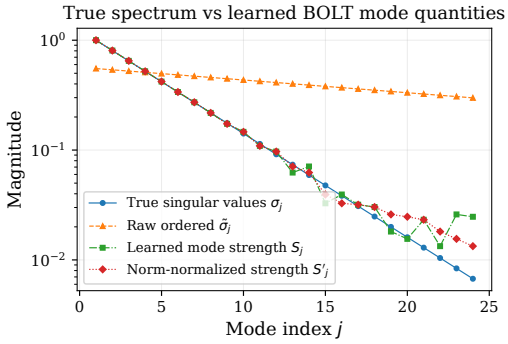
Deep neural networks (DNNs) are a key driving force behind many modern artificial intelligence (AI) systems due to their powerful learning capabilities. However, the growing scale and complexity of DNNs demand substantial memory, computation, and infrastructure. This limits deployment on edge devices and restricts experimentation to well-resourced settings. Existing compression methods commonly use sparsity, reduced precision, or low-rank factorization, but often lack direct control over the final model size under a prescribed global parameter budget. We address this limitation by formulating model compression as a budget-constrained optimization problem and presenting Budget-aware Ordered Low-rank Training (BOLT), a training-time compression method that learns data-driven ordered low-rank parameterizations under an explicit parameter budget. BOLT factorizes eligible layers into ordered low-rank factors and uses supervised learning and soft structural self-transfer to concentrate information learned from the data into leading modes. It then applies a scale-invariant mode-strength criterion for ordering and estimates the loss impact of complete suffix removal using a first-order Taylor approximation and directional Gauss–Newton (GN) curvature. Sparse Pareto dynamic programming (DP) then selects layer specific prefix ranks under the global budget followed by replay buffer based output distribution drift validation. This allows BOLT to determine suitable ranks for each layer without predefined rank schedules or manual pruning thresholds. Experiments across convolutional and Transformer architectures show that BOLT performs competitively against seven baselines, including the dense model. BOLT achieves near-dense performance with only 10% of the original parameters on select architectures, enabling up to 90% model reduction for resource-constrained deployment.

1 INTRODUCTION

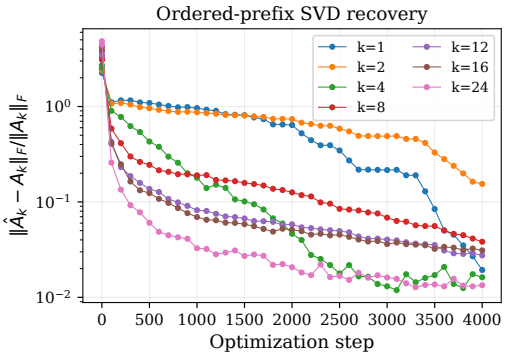
DNNs have driven major advances in AI by learning complex functional mappings directly from data. However, this representational power often requires substantial memory, computation, energy, and infrastructure. As architectures, datasets, and task demands have grown in complexity many modern DNNs based AI such as foundational models, visual transformer have scaled to billions of parameters (Xu et al., 2025). Training and fine-tuning such large-scale DNN models typically require substantial graphical processing unit (GPU) memory, computational throughput, energy consumption, and financial investment in infrastructure, which restricts meaningful experimentation to well-resourced researchers, institutions, and organizations (Strubell et al., 2019). Reducing model size is therefore important for improving research accessibility, lowering deployment and fine-tuning costs, and enabling practical use on resource-constrained platforms such as mobile devices, embedded systems, drones, and Internet-of-Things (IoT) systems (Meuser et al., 2024).

Prior studies show that DNNs often contain substantial redundancy, indicating that only a subset of parameters or structural components may be essential for achieving the learning objective (Nanda et al., 2023). Model compression addresses this by reducing model size while preserving accuracy, commonly through pruning, quantization, and low-rank factorization (Cheng et al., 2024; Cai et al., 2022). Pruning removes redundant weights or structures (Frankle & Carbin, 2018), while quantization lowers numerical precision (Jacob et al., 2018). However, their practical speedups often require specialized sparse kernels, quantization-aware operators, accelerator support, or architecture-specific constraints (Yang et al., 2020). Low-rank factorization instead represents dense weights using lower-

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.



(a) True SVD spectrum σ_j, learned raw singular coefficient $\tilde{\sigma}_j$, learned mode strength S_j, and norm-normalized mode strength S'_j.



(b) Ordered-prefix recovery error for oracle truncated-SVD maps.

Figure 1: Verification of BOLT's structural hypothesis on a generated matrix $A \in \mathbb{R}^{64\times64}$, constructed as $A = U\,\mathrm{diag}(s)V^\top$ with rank 24. (a) The scale-aware mode strength $S_j = \tilde{\sigma}j|N{:},j|2|Cj,{:}|_2$ tracks the true spectrum σ_j more accurately than the raw coefficient $\tilde{\sigma}_j$, while the normalized strength S'_j closely preserves the spectral ordering. (b) The relative Frobenius prefix error $\|\hat{A}_k - A_k\|_F / \|A_k\|_F$ shows that BOLT's learned rank-k prefixes closely recover the SVD maps.

rank factors, reducing parameters and computation while preserving the layer input-output interface (Guo & Yu, 2025; Horváth et al., 2023). A key challenge, however, is selecting suitable layer-wise ranks, since poor rank allocation can either retain unnecessary parameters or degrade predictive performance (Wang et al., 2023).

To address this challenge, we propose BOLT, a two stage compression framework that learns compact low rank DNNs under an explicit global parameter budget. BOLT uses an singular value decomposition (SVD) based representation and learns ordered prefixes that concentrate information learned from the data in leading modes. Since factor rescaling can make spectral coefficients alone unreliable indicators of mode importance, BOLT uses a scale aware mode strength that incorporates both the coefficients and learned factors, as illustrated in Fig. 1. It then estimates the supervised loss change from complete suffix removal using first order Taylor approximation and directional GN curvature and formulates layer wise prefix rank allocation as a multiple choice knapsack problem under the target budget. Sparse Pareto DP solves this allocation, and low damage budget feasible candidates are validated on replay data to control output-distribution drift before physical compaction. Our contributions are:

1. We introduce BOLT, a data-driven ordered low-rank compression framework that formulates compression as an explicit budget-constrained optimization problem.
2. We develop a scale-aware mode-strength criterion together with a suffix removal loss change that combines a first-order Taylor approximation with directional GN curvature, followed by replay-based output distribution drift validation.
3. We formulate the theoretical basis of BOLT as a budget-constrained ordered prefix selection problem and characterize how feasible rank selection can satisfy a target budget while bounding the increase in supervised loss.
4. We empirically evaluate BOLT across representative convolutional neural network (CNN) and Transformer architectures on diverse benchmarks, demonstrating competitive performance–capacity trade-offs against established baselines.

2 Related Work

Representing a DNN weight matrix as the product of two lower-rank factor matrices is a common compression strategy that has been extensively explored in prior works (Yaguchi et al., 2019). However, because DNNs consist of heterogeneous layers with different dimensionalities, sensitivities, and redundancy patterns, determining suitable layer-wise ranks while preserving model performance during training remains challenging (Idelbayev & Carreira-Perpinán, 2020). A common direction is low-rank training, where structural regularizers are added to the objective to optimize supervised loss while encouraging weight matrices to become low-rank. Alvarez & Salzmann (2017) use a nuclear-norm regularizer with proximal stochastic gradient descent, whose update involves singular-value

soft-thresholding, while Xu et al. (2019) propose Trained Rank Pruning (TRP), alternating low-rank approximation with network training and nuclear regularization. These methods, however, rely on repeated low-rank approximation or singular-value operations during optimization. Yang et al. (2020) reduce this overhead through SVD training, which decomposes each layer into a full-rank SVD form and directly optimizes the factors using orthogonality and sparsity-inducing regularizers, followed by singular-value pruning.

Meanwhile, rank-selection approaches directly search for layer-wise rank configurations that balance compression and accuracy. PARS (Sobolev et al., 2022) uses Bayesian optimization to identify suitable tensor ranks, while Eo et al. (2021) combine modified beam search with stable-rank-based training to obtain compression-friendly low-rank models. ShrinkML (Abdelfattah et al., 2019) applies reinforcement learning to optimize per-layer compression ratios for SVD-based low-rank compression in end-to-end ASR models without considering the data distribution. Guo & Yu (2025) introduce layer-wise binary-search rank selection under a theoretical loss-error bound, although their search procedure still requires evaluating SVD-based low-rank approximations during rank selection. Prior studies show that the warm-up duration before switching to low-rank training can significantly affect final accuracy (Wang et al., 2023). Cuttlefish (Wang et al., 2023) addresses this by observing that layer-wise stable ranks change rapidly early in training before converging, and uses this behavior to adaptively choose both the transition point and per-layer ranks.

More recently, Horváth et al. (2023) introduced MAESTRO, which extends ordered dropout (OD) (Horvath et al., 2021) through LoD for low-rank factorized layers. Inspired by this ordered-subnetwork principle, BOLT learns a spectral ordering that concentrates high-importance modes in early prefixes and suppresses redundant tail modes. MAESTRO learns importance implicitly through LoD sampling, hierarchical group-lasso shrinkage, and threshold-based pruning. BOLT instead uses an explicit spectral parameterization, with scale-aware mode strength and soft structural self-transfer to learn prefix-compatible factors. BOLT further formulates rank selection as a global parameter budget problem, solved by sparse Pareto DP over prefix damage estimates obtained from first-order Taylor terms and directional GN curvature, followed by replay-based validation before physical compaction.

3 Problem Statement

We consider a neural network $f_\Theta(\cdot)$ parameterized by Θ and trained on the dataset $\mathcal{D}_{\text{train}} = \{(X_i, y_i)\}_{i=1}^N$, with layer-wise weight matrices $\{W_\ell\}_{\ell=1}^{\mathbb{L}}$, where $\ell \in \{1, \dots, \mathbb{L}\}$ indexes the network layers. Let L_{sup} denote the sample-wise supervised loss associated with the downstream task. The corresponding empirical supervised loss is $\mathcal{L}_{\text{sup}}(\Theta; \mathcal{D}_{\text{train}}) = \frac{1}{N}\sum_{i=1}^N L_{\text{sup}}\left(f_\Theta(X_i), y_i\right)$. For classification, $L_{\text{sup}} = L_{\text{CE}}$, where L_{CE} denotes the cross-entropy (CE) loss. Also, let $\mathcal{S} \subseteq \{1, \dots, \mathbb{L}\}$ denote the subset of layers whose weights are represented using low-rank factorizations. For layer ℓ, a fully connected weight matrix is $W_\ell \in \mathbb{R}^{m_\ell \times n_\ell}$, while convolutional kernels are unfolded into matrix form before factorization. The construction is motivated by SVD form $W_\ell = U_\ell \Sigma_\ell V_\ell^\top$, for which a rank-$k_\ell$ approximation retains the first k_ℓ singular components.

Each eligible layer $\ell \in \mathcal{S}$ is represented as an ordered sum of learned rank-one modes, $\widehat{W}_\ell = \sum_{j=1}^{r_\ell} \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$, where r_ℓ is the maximum factorization rank, $u_{\ell,j}$ and $v_{\ell,j}$ are learned factor vectors, and $\tilde{\sigma}_{\ell,j}$ is the corresponding spectral coefficient. A prefix rank k_ℓ retains $\widehat{W}_\ell^{(k_\ell)} = \sum_{j=1}^{k_\ell} \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$, with $k_\ell \in \{k_\ell^{\min}, \dots, r_\ell\}$. Let P_{dense} denote the parameter count of the original dense network and let $B = \rho_B P_{\text{dense}}$, where $\rho_B \in (0, 1]$ is the target budget ratio. If $c_\ell \in \mathbb{N}_{>0}$ denotes the physical parameter cost of retaining one rank-one mode in layer ℓ, then the rank-dependent structural parameter count is $P_{\text{struct}}(\mathbf{k}) = \sum_{\ell \in \mathcal{S}} c_\ell k_\ell$. Let P_{fixed} denote all parameters unaffected by $\mathbf{k}$, so that the retained physical parameter count is $P_{\text{phys}}(\mathbf{k}) = P_{\text{fixed}} + P_{\text{struct}}(\mathbf{k})$. Given the ordered low-rank representation, the central problem is to select layer-wise prefix ranks $\mathbf{k} = \{k_\ell : \ell \in \mathcal{S}\}$ that satisfy a prescribed global parameter budget while minimizing the resulting loss change:

$$\begin{aligned} \mathbf{k}^\star_{\text{true}} = \arg\min_{\mathbf{k}} \quad & \mathcal{L}_{\text{sup}}\left(\Theta_{\mathbf{k}}; \mathcal{D}_{\text{train}}\right) - \mathcal{L}_{\text{sup}}\left(\Theta; \mathcal{D}_{\text{train}}\right) \\ \text{s.t.} \quad & P_{\text{phys}}(\mathbf{k}) \le B, \quad k_\ell \in \{k_\ell^{\min}, \dots, r_\ell\} \qquad \forall \ell \in \mathcal{S} \end{aligned} \tag{1}$$

Here, Θ denotes the full-prefix ordered model at the rank-selection stage, whereas $\Theta_{\mathbf{k}}$ denotes the network obtained using the selected layer-wise prefix ranks $\mathbf{k}$.

4 BOLT

This section presents the two stage BOLT compression method, with Algorithm 1 summarizing the overall procedure.

4.1 Learning Data-driven Ordered Prefix Substructure

Consider an eligible layer $\ell \in \mathcal{S}$ with weight matrix $W_\ell \in \mathbb{R}^{m_\ell \times n_\ell}$, approximated by r_ℓ learned rank-one modes, $\widehat{W}_\ell = \sum_{j=1}^{r_\ell} \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$, where $\tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$ is the j-th learned mode. Let $U_\ell = [u_{\ell,1}, \ldots, u_{\ell,r_\ell}] \in \mathbb{R}^{m_\ell \times r_\ell}$ and $V_\ell = [v_{\ell,1}, \ldots, v_{\ell,r_\ell}]^\top \in \mathbb{R}^{r_\ell \times n_\ell}$ denote the corresponding factor matrices, such that $u_{\ell,j} = U_{\ell,:,j}$ and $v_{\ell,j}^\top = V_{\ell,j,:}$. The model-specific layers included in $\mathcal{S}$ are detailed in Appendix G . In a simplified linear setting with $y = W_\ell x$ and centered inputs $x \sim \mathcal{X}$, a common low-rank objective minimizes $\mathbb{E}_{x\sim\mathcal{X}} \|\widehat{W}_\ell x - W_\ell x\|_2^2$. However, this only encourages the complete operator to approximate W_ℓ accurately and does not ensure that a truncated prefix $\widehat{W}_\ell^{(k)} = \sum_{j=1}^{k} \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$, with $k < r_\ell$, preserves the same behavior. BOLT therefore uses a prefix-stability principle, where partial ordered operators are trained to match the full ordered operator:

$$\min_{\{u_{\ell,j}, v_{\ell,j}, \tilde{\sigma}_{\ell,j}\}_{j=1}^{r_\ell}} \mathbb{E}_{x\sim\mathcal{X}} \left[\left\| \widehat{W}_\ell x - W_\ell x \right\|_2^2 + \alpha\, \mathbb{E}_{\xi\sim\Pi} \left\| \widehat{W}_{\ell,\xi} x - \mathrm{sg}\left(\widehat{W}_\ell x\right) \right\|_2^2 \right] \tag{2}$$

Here, $\xi = (\xi_1, \ldots, \xi_{r_\ell})$ denotes an ordered-dropout mask satisfying $\xi_1 \geq \xi_2 \geq \cdots \geq \xi_{r_\ell}$, with $\xi_j \in \{0, 1\}$, and $\widehat{W}_{\ell,\xi} = \sum_{j=1}^{r_\ell} \xi_j \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$ is the corresponding partial ordered operator for layer ℓ. The distribution Π denotes the ordered-dropout induced distribution over such prefix-restricted operators. The operator $\mathrm{sg}(\cdot)$ denotes stop-gradient, which preserves the numerical value of $\widehat{W}_\ell x$ in the forward pass while preventing gradients from propagating through the full-operator target during backpropagation. Equation 2 motivates the soft-transfer mechanism used for ordered-substructure learning in BOLT.

BOLT extends the same prefix-stability principle to nonlinear networks at the model output level, with the corresponding supervised and soft-transfer losses detailed in Appendix H. Let $z_i^{\mathrm{full}} = f_\Theta(X_i)$ denote the output logits of the full ordered model, $z_i^{\mathrm{part}} = f_{\Theta,\xi}(X_i)$ denote the output logits of a prefix-restricted submodel obtained with ordered dropout active. The full model produces the soft teacher target $q_i = \mathrm{sg}\left(\mathrm{softmax}(z_i^{\mathrm{full}})\right)$, and the prefix-consistent learning objective is

$$\mathcal{L}_{\mathrm{learn}} = L_{\mathrm{sup}}\left(z_i^{\mathrm{full}}, y_i\right) + \lambda_{\mathrm{soft}} L_{\mathrm{soft}}\left(z_i^{\mathrm{part}}, q_i\right) \tag{3}$$

Where λ_{soft} is the soft-transfer weight. For classification, where $L_{\mathrm{sup}} = L_{\mathrm{CE}}$, the soft-transfer loss is

$$L_{\mathrm{soft}}\left(z_i^{\mathrm{part}}, q_i\right) = -\sum_{c=1}^{C_{\mathrm{cls}}} q_{i,c} \log\left[\mathrm{softmax}\left(z_i^{\mathrm{part}}\right)\right]_c \tag{4}$$

where $c \in \{1, \ldots, C_{\mathrm{cls}}\}$ indexes the class. The full ordered model is trained with labels while the partial ordered model matches the full model's predictive behavior. The partial logits z_i^{part} are obtained using positional ordered dropout over rank-one modes where later modes are dropped more often than earlier modes. This encourages information to concentrate in prefix modes and makes suffix modes more removable. Details are provided in Appendix B.

Because the learned rank-one factors are scale non-identifiable, rescaling $\tilde{\sigma}_{\ell,j}$, $U_{\ell,:,j}$, and $V_{\ell,j,:}$ can leave the represented component unchanged. Thus, $\tilde{\sigma}_{\ell,j}$ alone is not a reliable measure of mode importance. BOLT instead uses the scale-aware strength $S_{\ell,j} = \tilde{\sigma}_{\ell,j} \|U_{\ell,:,j}\|_2 \|V_{\ell,j,:}\|_2$, which more closely tracks the target singular spectrum as shown in Fig. 1a. Furthermore, $S_{\ell,j}$ is not guaranteed to remain ordered. Let $\pi_\ell(j)$ denote the position assigned to mode j after sorting the modes in descending order of $S_{\ell,j}$. We define the resulting ordered canonical strength as $S'_{\ell,\pi_\ell(j)} = S_{\ell,j}$. BOLT then canonicalizes and reindexes each mode as

$$U'_{\ell,:,\,\pi_\ell(j)} = \frac{U_{\ell,:,j}}{\|U_{\ell,:,j}\|_2} \qquad V'_{\ell,\pi_\ell(j),:} = \frac{V_{\ell,j,:}}{\|V_{\ell,j,:}\|_2} \qquad \tilde{\sigma}'_{\ell,\pi_\ell(j)} = S'_{\ell,\pi_\ell(j)} = S_{\ell,j} \tag{5}$$

Accordingly, $\tilde{\sigma}'_{\ell,\pi_\ell(j)} U'_{\ell,:,\,\pi_\ell(j)} V'_{\ell,\pi_\ell(j),:} = \tilde{\sigma}_{\ell,j} U_{\ell,:,j} V_{\ell,j,:}$ so each currently active rank-one component is preserved up to permutation and reciprocal factor rescaling, while the ordered spectral

coefficient becomes the scale-invariant mode strength used to canonicalize and maintain the ordered prefix representation. The ordered-substructure learning objective then is:

$$\mathcal{L}_{\mathrm{os}}(\Theta) = \mathbb{E}_{\substack{(X_i,y_i)\sim\mathcal{D}_{\mathrm{train}} \\ \xi\sim\Pi}} \Big[L_{\mathrm{sup}}(f_\Theta(X_i), y_i) + \lambda_{\mathrm{soft}} L_{\mathrm{soft}}(f_{\Theta,\xi}(X_i), q_i) \Big] \tag{6}$$

This objective combines supervised learning and prefix consistency soft transfer in Eq. 3. At each update, the teacher targets q_i are computed from the current full model and held fixed during differentiation, yielding a semi-gradient update. Both outputs use the same ordered SVD parameterized network with ordered dropout applied only to the partial pass. BOLT next performs discrete layer wise rank selection under the prescribed parameter budget.

4.2 Budget-Constrained Prefix Rank Selection

After learning the data-driven ordered substructure, BOLT considers $k_\ell \in \{k_\ell^{\min}, \dots, r_\ell^+\}$, where retaining k_ℓ keeps modes $1, \dots, k_\ell$ and removes the complete suffix $k_\ell + 1, \dots, r_\ell^+$. Direct evaluation of the supervised-loss change for every joint prefix configuration in Eq. 1, together with its exact second-order loss geometry, is computationally impractical for modern networks (Idelbayev & Carreira-Perpinán, 2020). BOLT therefore constructs a tractable local approximation using first order Taylor approximation and directional GN curvature.

Let $\mathcal{B}_{\mathrm{rep}} \subset \mathcal{D}_{\mathrm{train}}$ denote a small replay set. Then, for classification where $\mathcal{L}_{\mathrm{sup}} = \mathcal{L}_{\mathrm{CE}}$, the cross-entropy loss over the replay is

$$\mathcal{L}_{\mathrm{CE}}^{\mathrm{rep}}(\Theta) = \frac{1}{|\mathcal{B}_{\mathrm{rep}}|} \sum_{(X_i,y_i)\in\mathcal{B}_{\mathrm{rep}}} L_{\mathrm{CE}}\left(f_\Theta(X_i), y_i\right) \tag{7}$$

To estimate the effect of suffix removal, BOLT associates each active mode with a temporary gate $g_{\ell,j}$, such that $\tilde{\sigma}_{\ell,j} \to g_{\ell,j}\tilde{\sigma}_{\ell,j}$, with $g_{\ell,j} = 1$ at the current state. For a candidate rank k_ℓ, the gate perturbation for mode j is $\Delta g_{\ell,j}^{(k_\ell)} = -\mathbb{I}(j > k_\ell)$, where indicator function $\mathbb{I}(\cdot)$ equals 1 when its argument is true and 0 otherwise. We define the suffix perturbation vector $\Delta\mathbf{g}_\ell^{(k_\ell)} = [\Delta g_{\ell,1}^{(k_\ell)}, \dots, \Delta g_{\ell,r_\ell^+}^{(k_\ell)}]^\top$ that collects these gate perturbations. The replay-loss gradient with respect to the structural gates is

$$\mathbf{a}_\ell = \nabla_{\mathbf{g}_\ell} \mathcal{L}_{\mathrm{CE}}^{\mathrm{rep}}(\Theta)\big|_{\mathbf{g}_\ell=\mathbf{1}} \qquad \mathbf{g}_\ell = (g_{\ell,1}, \dots, g_{\ell,r_\ell^+})^\top$$

The corresponding first-order loss-change estimate is $\mathbf{a}_\ell^\top \Delta\mathbf{g}_\ell^{(k_\ell)}$. Since suffix removal changes multiple gates from one to zero, BOLT augments this first-order estimate with GN curvature. For sample i, let $p_i = \mathrm{softmax}(z_i)$ and the logit Jacobian with respect to the gates of layer ℓ be $J_{\ell,i} = \frac{\partial z_i}{\partial \mathbf{g}_\ell}$. For softmax cross-entropy, the Hessian with respect to the logits is $H_i = \mathrm{diag}(p_i) - p_i p_i^\top$, $H_i \succeq 0$ The corresponding layer-wise GN matrix is

$$G_\ell = \frac{1}{|\mathcal{B}_{\mathrm{rep}}|} \sum_{(X_i,y_i)\in\mathcal{B}_{\mathrm{rep}}} J_{\ell,i}^\top H_i J_{\ell,i} \tag{8}$$

Since $H_i \succeq 0$, it follows that $G_\ell \succeq 0$. BOLT then defines the directional GN estimate of the predictive loss change associated with retaining prefix k_ℓ as

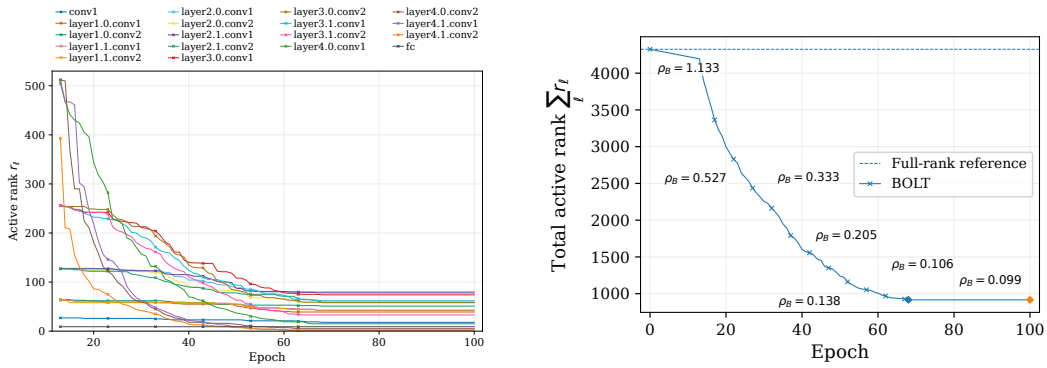
$$D_\ell^{\mathrm{GN}}(k_\ell) = \mathbf{a}_\ell^\top \Delta\mathbf{g}_\ell^{(k_\ell)} + \frac{1}{2} \left(\Delta\mathbf{g}_\ell^{(k_\ell)}\right)^\top G_\ell \Delta\mathbf{g}_\ell^{(k_\ell)} \tag{9}$$

Equation 9 evaluates each candidate suffix as a joint structural perturbation. In particular,

$$\left(\Delta\mathbf{g}_\ell^{(k_\ell)}\right)^\top G_\ell \Delta\mathbf{g}_\ell^{(k_\ell)} = \sum_{u>k_\ell} \sum_{v>k_\ell} [G_\ell]_{u,v} \tag{10}$$

which retains within-layer interactions among all modes removed after k_ℓ. Since every admissible k_ℓ is assessed as a complete prefix, individual mode sensitivities are not required to follow a monotonic ordering. The required directional quantities can be computed through Jacobian-vector or vector-Jacobian products without constructing the full network Hessian. The resulting curve $D_\ell^{\mathrm{GN}}(k_\ell)$ assigns an estimated predictive loss change to each admissible prefix rank. BOLT then selects ranks per structural layer by solving:

$$\mathbf{k}^\star = \arg\min_{\mathbf{k}} \sum_{\ell\in\mathcal{S}} D_\ell^{\mathrm{GN}}(k_\ell) \qquad \text{s.t.} \qquad P_{\mathrm{fixed}} + \sum_{\ell\in\mathcal{S}} c_\ell k_\ell \le B \tag{11}$$



(a) Layer-wise ranks retained by BOLT in ResNet-18. (b) Progressive active rank reduction

Figure 2: BOLT rank selection during training on ResNet-18 with CIFAR10. Layer-wise active ranks vary across layers according to their estimated contribution, while the total active rank decreases progressively until the target budget is reached.

where $k_\ell \in \{k_\ell^{\min}, \ldots, r_\ell^+\}$, with $k_\ell^{\min} \geq 1$ and $P_{\text{fixed}} > 0$. This jointly balances the estimated supervised-loss change associated with each complete prefix against its physical parameter cost $c_\ell k_\ell$ across layers. BOLT solves the multiple-choice knapsack problem in Eq. 11 with sparse Pareto dynamic programming. At each layer, it discards rank choices that exceed the parameter budget, retains a bounded number of the lowest-damage states at each parameter cost, and removes states that are dominated in both cost and estimated damage. Since the GN objective is a local surrogate, BOLT retains multiple low-damage Pareto candidates and verifies them on the replay set using both the observed cross-entropy change $\Delta\mathcal{L}_{\text{CE}}^{\text{rep}}(\mathbf{k})$ and predictive-distribution drift $\mathcal{D}_{\text{KL}}(\mathbf{k})$. For a candidate $\mathbf{k}$, these quantities are defined as

$$\Delta\mathcal{L}_{\text{CE}}^{\text{rep}}(\mathbf{k}) = \mathcal{L}_{\text{CE}}^{\text{rep}}(\Theta_{\mathbf{k}}) - \mathcal{L}_{\text{CE}}^{\text{rep}}(\Theta) \tag{12}$$

$$\mathcal{D}_{\text{KL}}(\mathbf{k}) = \frac{1}{|\mathcal{B}_{\text{rep}}|} \sum_{(X_i, y_i) \in \mathcal{B}_{\text{rep}}} D_{\text{KL}}\left(p_i^{\text{ref}} \,\|\, p_i^{\mathbf{k}}\right) \tag{13}$$

Here p_i^{ref} and $p_i^{\mathbf{k}}$ denote the predictive distributions of the reference and candidate models respectively, while $D_{\text{KL}}(\cdot\|\cdot)$ denotes the KL divergence. Candidates satisfying the parameter budget and the prescribed CE and distribution drift limits are accepted. If the lowest-surrogate Pareto candidate violates these limits, subsequent low-damage Pareto candidates are examined. When an evaluated candidate satisfies both the parameter budget and the replay acceptance criteria, BOLT accepts its prefix configuration and physically removes the discarded suffix modes.

4.3 Theoretical Guarantees and Assumptions

The assumptions, supporting propositions, and proofs underlying the following theorems are provided in Appendix C.

Theorem 1: ***(Exact Surrogate Optimization)*** For the fixed additive prefix-damage surrogate, BOLT's sparse Pareto DP exactly minimizes the surrogate objective over the budget-feasible prefix set.

Theorem 2: ***(Error-Bounded Near-Optimality)*** The true-loss regret of the Pareto-selected prefix is bounded by the surrogate errors of the selected and true-optimal feasible prefixes. Under a uniform surrogate error bound, the regret is at most twice that error.

5 Experiments

Architectures. We evaluate the compression performance of BOLT across CNN and Transformer-based architectures spanning vision and language tasks. The CNN architectures include ResNet-18, ResNet-50 from the residual network family (He et al., 2016) and VGG-19 from the VGG family (Simonyan & Zisserman, 2014). Transformer-based experiments use BERT-base-cased (Devlin et al., 2019) and DeBERTa-v3-base (He et al., 2021) for language classification, M2M-100 1.2B (Fan et al., 2021) for machine translation, and ViT-Huge/14 (Dosovitskiy et al., 2020) and DINOv2-Giant (Oquab et al., 2023) for vision tasks. Refer to Appendix F for details.

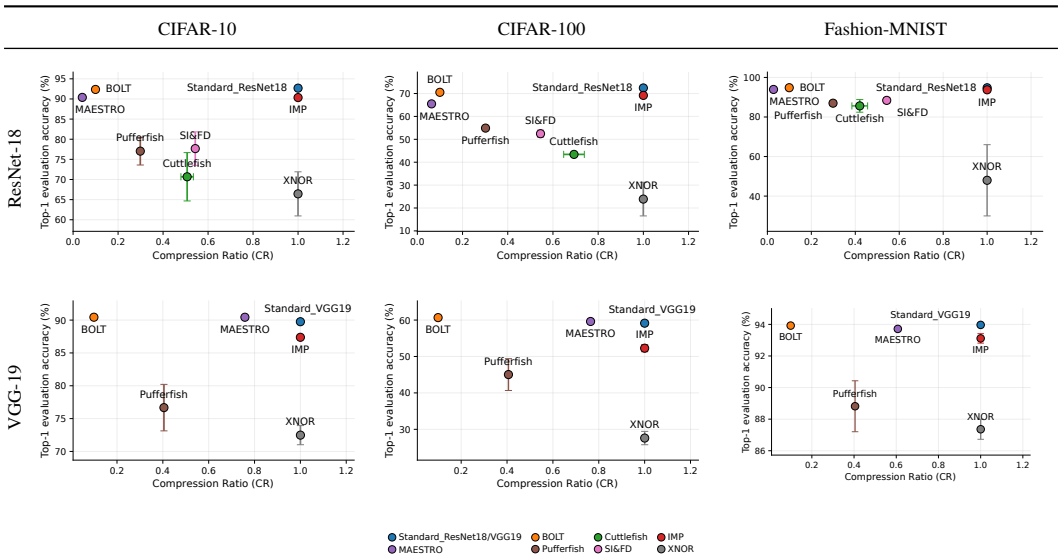


Figure 3: Top-1 test accuracy vs physical compression ratio across architectures and datasets, reported as mean $\pm$ standard deviation over five seeds. Points closer to the upper-left region indicate better accuracy-compression trade-offs. With the target budget ratio set to $\rho_B = 0.10$, BOLT preserves near-dense accuracy across most settings while achieving nearly 90% parameter reduction.

Datasets. We evaluate BOLT across vision and language tasks using standard image classification and sentence-pair classification benchmarks. The vision datasets include MNIST (LeCun, 1998), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), PathMNIST (Yang et al., 2023), and ImageNet-1K (Deng et al., 2009). We conduct language evaluation, using MRPC (Dolan & Brockett, 2005) and MNLI (Williams et al., 2018) from the GLUE benchmark (Wang et al., 2018), together with PAWS (Zhang et al., 2019). Furthermore, we use the Multi30K English–German dataset (Elliott et al., 2016) to evaluate machine translation. Dataset details and experimental partitions are provided in Appendix D.

Baselines. We compare BOLT against six representative compression baselines selected from three major families of model compression algorithms. The low-rank baselines include MAESTRO (Horváth et al., 2023), Cuttlefish (Wang et al., 2023), Pufferfish (Wang et al., 2021), and SI&FD (Khodak et al., 2021). For pruning-based compression, we compare against iterative magnitude pruning (IMP) (Frankle & Carbin, 2018). For quantization-based compression, we compare against XNOR-Net (Rastegari et al., 2016). For fair comparison, MAESTRO, Cuttlefish, Pufferfish, SI&FD, IMP, and XNOR-Net are physically compacted only after training, as many of them do not support BOLT-style intermediate compression. Further details are provided in Appendix E.

Evaluation Metrics. We assess BOLT across three key performance dimensions.
(i) Task performance. On vision tasks, we report Top-1 accuracy. In the case of language evaluation, we report accuracy on MNLI and PAWS and F1 on MRPC, while machine translation on Multi30K is evaluated using BLEU (Papineni et al., 2002) score.
(ii) Compression ratio (CR). We define $\mathrm{CR} = P_{\mathrm{comp}}/P_{\mathrm{dense}}$, where P_{comp} and P_{dense} denote the parameter counts of the compacted and dense models. The target budget is satisfied when $\mathrm{CR} \leq \rho_B$.
(iii) Computational efficiency. We report FLOPs, inference latency, and peak memory usage.

Compute environment. All experiments were conducted on a computational cluster equipped with NVIDIA L40S GPUs.

5.1 Performance Analysis

Performance-Capacity Trade-off. This criterion reflects how well a DNN preserves predictive performance as model capacity is reduced. BOLT demonstrates a strong performance–capacity trade-off by consistently operating near the upper-left region of Fig. 3. Across ResNet-18 and VGG-19, BOLT retains accuracy close to the dense models at substantially lower capacity and in most cases outperforms baseline compression methods at comparable or larger model sizes.

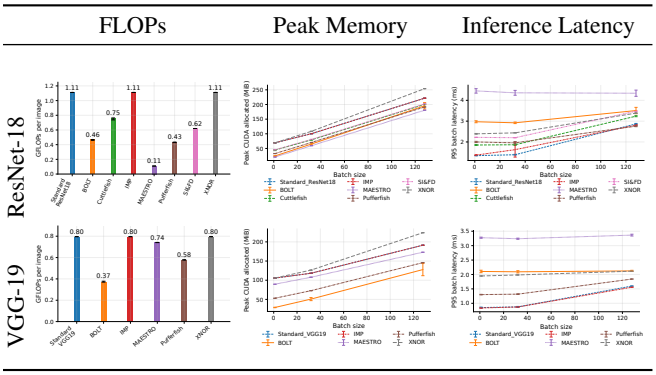


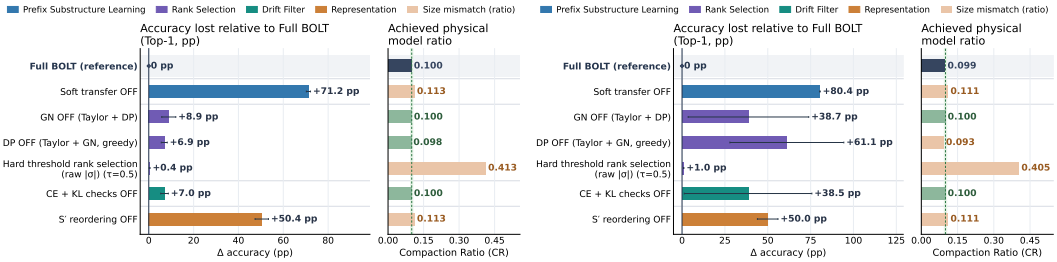
Figure 4: Computational efficiency of ResNet-18 and VGG-19 on CIFAR10. Additional results are reported in Appendix I.

Table 1: Performance across language and vision tasks. #Params: M = million, B = billion.

	Dataset	Model	Performance	Target(ρ_2)	#Params
Language	MNLI	BERT-base-cased	Acc. 82.40%	–	108.3M
		BOLT+BERT-base-cased	**Acc.** 81.77%	0.6	**64.9M**
		DeBERTa-v3	Acc. 88.89%	–	184.4M
		BOLT+DeBERTa-v3	**Acc.** 85.53%	0.6	**110.7M**
	PAWS	BERT-base-cased	Acc. 91.52%	–	108.3M
		BOLT+BERT-base-cased	**Acc.** 89.93%	0.5	**53.1M**
		DeBERTa-v3	Acc. 94.96%	–	184.4M
		BOLT+DeBERTa-v3	**Acc.** 82.47%	0.6	**101.4M**
	MRPC	BERT-base-cased	F1 0.88	–	108.3M
		BOLT+BERT-base-cased	**F1** 0.85	0.3	**32.4M**
		DeBERTa-v3	F1 0.92	–	184.4M
		BOLT+DeBERTa-v3	**F1** 0.89	0.6	**110.6M**
	Multi30K	M2M-100 1.2B	BLEU 43.74	–	1.2B
		BOLT+M2M-100 1.2B	**BLEU** 41.09	0.75	**899.8M**
Vision	PathMNIST	ResNet-50	Top-1 89.75%	–	23.52M
		BOLT+ResNet-50	**Top-1** 89.50%	**0.1**	**2.35M**
	ImageNet-1K	ViT-Huge/14	Top-1 78.61%	–	632.05M
		BOLT+ViT-Huge/14	**Top-1** 74.22%	0.3	**190.3M**
		DINOv2-Giant	Top-1 83.24%	–	1.14B
		BOLT+DINOv2-Giant	**Top-1** 80.26%	0.6	**695.4M**

This behavior extends beyond CNNs to model families with substantially different architectures, scales, and learning objectives, as summarized in Table 1. The most pronounced capacity reduction is observed on PathMNIST, where BOLT compresses ResNet-50 to only 10% of its original parameter count while incurring a marginal Top-1 accuracy drop. More importantly, this behavior persists at substantially larger model scales. On ImageNet-1K, BOLT reduces ViT-Huge/14 from 632.05M to 190.3M parameters while retaining 74.22% Top-1 accuracy, and compresses DINOv2-Giant from 1.14B to 695.4M parameters while preserving 80.26% Top-1 accuracy. This demonstrate that BOLT remains effective across a broad range of model capacities and training scales, consistently meeting parameter budgets while preserving strong predictive performance.

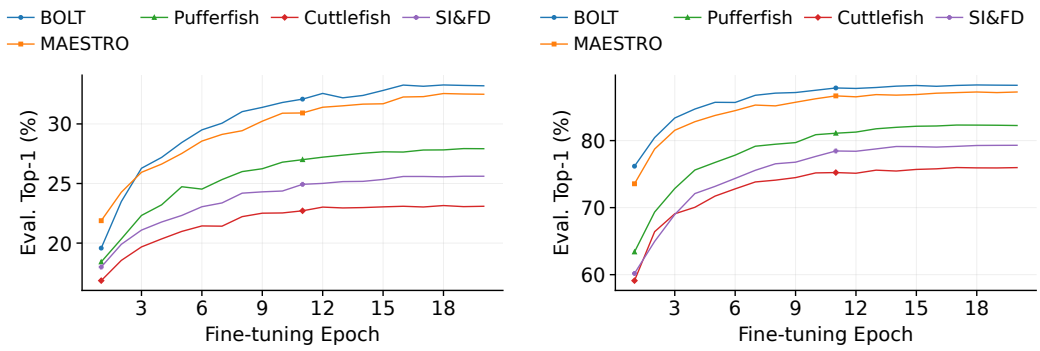
Efficiency and Rank-Selection Dynamics. As shown in Fig. 4, BOLT achieves substantial reductions in computation and peak memory. It reduces computation from 1.11 to 0.46 GFLOPs per image for ResNet-18 and from 0.80 to 0.37 GFLOPs for VGG-19. These savings arise from progressively removing redundant modes according to layer specific sensitivity, as illustrated in Fig. 2. Deeper convolutional layers such as `layer4.0.conv2` and `layer4.1.conv1` undergo substantial compression, whereas `conv1` is comparatively preserved and the `layer2` and `layer3` convolutions exhibit more gradual reductions. This adaptive allocation allows BOLT to distribute the compression budget according to the relative importance of individual layers. As a result, it reduces factor dimensions and intermediate representations, lowering both arithmetic cost and memory usage. These structural savings, however, do not translate directly into lower inference latency. Low rank methods such as BOLT, MAESTRO, Cuttlefish, Pufferfish, and SI&FD typically replace a dense convolution with two factorized convolutions. This introduces additional kernel launches, intermediate activations, and memory traffic. Consequently, BOLT achieves lower FLOPs and peak memory while still exhibiting higher P95 latency than the dense baseline, particularly at smaller batch sizes.



(a) BOLT ablation on CIFAR-10 (ResNet-18) (b) BOLT ablation on CIFAR-10 (VGG-19)

Figure 5: Ablation study of BOLT on CIFAR 10 with ResNet-18 and VGG-19, showing the accuracy degradation $\Delta\text{accuracy}$ and achieved CR when key components are disabled or replaced.

Ablation Study. We assess the core mechanisms of BOLT by ablating its key components. In Fig. 5, we report $\Delta\text{accuracy} = \text{accuracy}_{\text{Full BOLT}} - \text{accuracy}_{\text{Ablation}}$. A positive $\Delta\text{accuracy}$ indicates performance degradation in percentage points (pp) when a mechanism is removed. Removing soft transfer causes the largest accuracy losses of 71.2 and 80.4 pp for ResNet-18 and VGG-19, respectively, while disabling S' reordering results in losses of about 50 pp. To examine rank selection, we replace Pareto DP with greedy allocation or fixed thresholding based on raw $|\sigma|$. BOLT selects



(a) Fine-tuning on CIFAR-100

(b) Fine-tuning on MNIST

Figure 6: Fine-tuning performance of compact ResNet-18 models compressed on CIFAR-10. The curves show test accuracy across fine-tuning epochs, highlighting the stronger transferability of BOLT-compressed representations.

ranks by minimizing the estimated loss increase under the global parameter budget. Removing GN weakens this estimate, while replacing Pareto DP with greedy allocation makes budget allocation less effective. The importance of loss guided selection is further evident from raw $|\sigma|$ thresholding, which retains modes satisfying $|\sigma| > \tau$. This yields model ratios of 0.413 and 0.405 against a target of approximately 0.10 while still underperforming Full BOLT. Furthermore, the 7.0 and 38.5 pp losses without CE and KL filtering highlight the importance of controlling output drift when accepting candidate rank configurations. The ablation results provide clear evidence that BOLT's components jointly preserve performance while meeting the prescribed parameter budget.

Transferability of Compressed Representations. To assess whether compression preserves transferable information, we train ResNet-18 on CIFAR-10 and use the compacted models as frozen feature extractors with a newly trained classification head. As shown in Fig. 6, BOLT achieves the strongest transfer performance across the evaluated tasks, suggesting that its ordered low-rank structure preserves transferable representations more effectively than the compared baselines.

6 Limitations

BOLT builds on established ordered dropout principles (Horváth et al., 2023; Yang et al., 2020; Horvath et al., 2021) by retaining ordered prefixes of learned modes. However, restricting each layer to leading modes may exclude alternative mode subsets or factorizations that achieve better performance under the same budget. Its local GN surrogate also omits cross-layer interactions and may become less accurate under aggressive compression. Furthermore, BOLT's factorized execution introduces additional operations and memory traffic, resulting in higher inference latency.

7 Conclusion

In this study, we introduced BOLT, a budget-aware low-rank compression method that learns data-driven ordered prefix structures under an explicit parameter budget. BOLT combines soft structural self-transfer, scale-invariant mode-strength ordering, first-order Taylor and directional GN loss-change estimation, and sparse Pareto DP to select feasible layer-wise prefix ranks while limiting degradation of the learning objective. Our performance and model-capacity trade-off experiments show that BOLT achieves budget-satisfying physical compression while preserving near-dense performance in most settings. BOLT also demonstrates competitive computational efficiency relative to the baselines. Transferability analysis further suggests that the retained compact substructures preserve useful learned representations for downstream transfer. Future work will investigate cross-layer interactions into rank selection, budget-aligned prefix learning, and optimize factorized execution for lower inference latency.

Reproducibility Statement

We provide the source code used for model training, evaluation, and generation of all reported results to facilitate reproducibility of our experiments. BOLT's substructure learning and rank-selection procedures are described in Sections 4.1 and 4.2 and summarized in Algorithm 1. Additional details on ordered dropout, task-specific objectives, and eligible layers are given in Appendices B, H, and G respectively. Appendix C states the theoretical assumptions and provides proofs of the claims. Dataset partitions, model architectures, and baseline protocols are documented in Appendices D, F, and E, respectively. Section I.1 presents hyperparameter sensitivity analysis and explains the selected settings, with task-specific values reported in Table 7. Appendix I provides additional experimental results, including evaluations across five random seeds for ResNet-18 and VGG-19.

AI Usage Statement

In this study, AI usage was limited to manuscript formatting, grammar checking, implementing parts of the codebase, and debugging code issues. The authors reviewed and validated all AI-assisted contributions and take full responsibility for the manuscript, code, and reported results.

References

Mohamed S Abdelfattah, Ravichander Vipperla, Stefanos Laskaridis, Nicholas D Lane, et al. Shrinkml: End-to-end asr model compression using reinforcement learning. *arXiv preprint arXiv:1907.03540*, 2019.

Jose M Alvarez and Mathieu Salzmann. Compression-aware training of deep networks. *Advances in neural information processing systems*, 30, 2017.

Han Cai, Ji Lin, and Song Han. Efficient methods for deep learning. In *Advanced Methods and Deep Learning in Computer Vision*, pp. 159–190. Elsevier, 2022.

Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10558–10578, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the third international workshop on paraphrasing (IWP2005)*, 2005.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. Multi30k: Multilingual english-german image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pp. 70–74, 2016.

Moonjung Eo, Suhyun Kang, and Wonjong Rhee. A highly effective low-rank compression of deep neural networks with modified beam-search and modified stable rank. *arXiv preprint arXiv:2111.15179*, 2021.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Yifan Guo and Alyssa Yu. Integrating independent layer-wise rank selection with low-rank svd training for model compression: A theory-driven approach. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25, James Kwok (Ed.). International Joint Conferences on Artificial Intelligence Organization*, pp. 5289–5297, 2025.

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

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34:12876–12889, 2021.

Samuel Horváth, Stefanos Laskaridis, Shashank Rajput, and Hongyi Wang. Maestro: Uncovering low-rank structures via trainable decomposition. *arXiv preprint arXiv:2308.14929*, 2023.

Yerlan Idelbayev and Miguel A Carreira-Perpinán. Low-rank compression of neural nets: Learning the rank of each layer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8049–8059, 2020.

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

Mikhail Khodak, Neil Tenenholtz, Lester Mackey, and Nicolo Fusi. Initialization and regularization of factorized neural layers. *arXiv preprint arXiv:2105.01029*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Tobias Meuser, Lauri Lovén, Monowar Bhuyan, Shishir G Patil, Schahram Dustdar, Atakan Aral, Suzan Bayhan, Christian Becker, Eyal De Lara, Aaron Yi Ding, et al. Revisiting edge ai: Opportunities and challenges. *IEEE Internet Computing*, 28(4):49–59, 2024.

Vedant Nanda, Till Speicher, John Dickerson, Krishna Gummadi, Soheil Feizi, and Adrian Weller. Diffused redundancy in pre-trained representations. *Advances in Neural Information Processing Systems*, 36:4055–4079, 2023.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 4. Granada, 2011.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pp. 525–542. Springer, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Konstantin Sobolev, Dmitry Ermilov, Anh-Huy Phan, and Andrzej Cichocki. Pars: Proxy-based automatic rank selection for neural network compression via low-rank weight approximation. *Mathematics*, 10(20):3801, 2022.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 3645–3650, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Hongyi Wang, Saurabh Agarwal, and Dimitris Papailiopoulos. Pufferfish: Communication-efficient models at no extra cost. *Proceedings of Machine Learning and Systems*, 3:365–386, 2021.

Hongyi Wang, Saurabh Agarwal, Yoshiki Tanaka, Eric Xing, Dimitris Papailiopoulos, et al. Cuttlefish: Low-rank model training without all the tuning. *Proceedings of Machine Learning and Systems*, 5: 578–605, 2023.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pp. 1112–1122, 2018.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

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

Mengwei Xu, Dongqi Cai, Wangsong Yin, Shangguang Wang, Xin Jin, and Xuanzhe Liu. Resource-efficient algorithms and systems of foundation models: A survey. *ACM Computing Surveys*, 57(5): 1–39, 2025.

Yuhui Xu, Yuxi Li, Shuai Zhang, Wei Wen, Botao Wang, Wenrui Dai, Yingyong Qi, Yiran Chen, Weiyao Lin, and Hongkai Xiong. Trained rank pruning for efficient deep neural networks. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pp. 14–17. IEEE, 2019.

Atsushi Yaguchi, Taiji Suzuki, Shuhei Nitta, Yukinobu Sakata, and Akiyuki Tanizawa. Decomposable-net: Scalable low-rank compression for neural networks. *arXiv preprint arXiv:1910.13141*, 2019.

Huanrui Yang, Minxue Tang, Wei Wen, Feng Yan, Daniel Hu, Ang Li, Hai Li, and Yiran Chen. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 678–679, 2020.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific data*, 10(1):41, 2023.

Yuan Zhang, Jason Baldridge, and Luheng He. Paws: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1298–1308, 2019.

APPENDIX

A ALGORITHMS

Algorithm 1 BOLT Training and Budget-Constrained Compaction

Input: Training set $\mathcal{D}_{\text{train}}$, model f_Θ, training epochs E, warm-up epoch E_{warm}, target budget ratio ρ_B, ordered-dropout distribution Π, and hyperparameters λ_{soft}, δ_{rep}, and ϵ_{KL}.
Output: Compact model $f_{\Theta_{\mathbf{k}^\star}}$ satisfying the target parameter budget.

Initialize each eligible layer $\ell \in \mathcal{S}$ using an SVD-based rank-one factorization $\widehat{W}_\ell = \sum_{j=1}^{r_\ell} \tilde{\sigma}_{\ell,j} u_{\ell,j} v_{\ell,j}^\top$
Set $B \leftarrow \rho_B P_{\text{dense}}$, $k_\ell \leftarrow r_\ell$ for all $\ell \in \mathcal{S}$, and $\mathbf{k}^\star \leftarrow \{r_\ell : \ell \in \mathcal{S}\}$
for $t = 1, \ldots, E$ **do**
    for each minibatch $(X, y) \subset \mathcal{D}_{\text{train}}$ **do**
        Compute full and ordered-dropout prefix predictions.
        Update Θ using Eq. 6.
    end for
    if $t > E_{\text{warm}}$ **then**
        Canonicalize and reorder active modes by $S_{\ell,j}$.
        Construct a replay set $\mathcal{B}_{\text{rep}} \subset \mathcal{D}_{\text{train}}$.
        Compute $D_\ell^{\text{GN}}(k_\ell)$ using Eq. 9.
        Solve Eq. 11 via sparse Pareto DP.
        for Pareto candidate $\mathbf{k} = \{k_\ell : \ell \in \mathcal{S}\}$, ordered by surrogate damage **do**
            Evaluate $\Delta\mathcal{L}_{\text{CE}}^{\text{rep}}(\mathbf{k})$ and $\mathcal{D}_{\text{KL}}(\mathbf{k})$ using Eqs. 12–13.
            if $P_{\text{phys}}(\mathbf{k}) \le B$ **and** $\Delta\mathcal{L}_{\text{CE}}^{\text{rep}}(\mathbf{k}) \le \delta_{\text{rep}}$ **and** $\mathcal{D}_{\text{KL}}(\mathbf{k}) \le \epsilon_{\text{KL}}$ **then**
                Accept $\mathbf{k}^\star = \{k_\ell^\star : \ell \in \mathcal{S}\} \leftarrow \mathbf{k}$.
                Set the active rank of each layer $\ell \in \mathcal{S}$ to its selected prefix rank $k_\ell^\star$.
                Compact the retained prefixes while preserving the ordered low-rank form
                break
            end if
        end for
    end if
end for
Convert each eligible layer to its final compact form using the retained rank $k_\ell^\star$.
return compact model $f_{\Theta_{\mathbf{k}^\star}}$

B POSITIONAL ORDERED DROPOUT

We develop BOLT's ordered low-rank formulation based on prior work on nested subnetworks (Horvath et al., 2021), trainable low-rank decompositions (Horváth et al., 2023), and directly optimized low-rank factors (Yang et al., 2020). BOLT extends these ideas through positional ordered dropout that assigns higher retention probabilities to earlier low-rank modes. During soft transfer, BOLT forms the partial prediction z_i^{part} by applying a prefix mask to the ordered low-rank network. Earlier modes are therefore retained more frequently than later modes, encouraging predictive information to concentrate in leading prefixes while making suffix modes easier to remove.

For mode j in layer ℓ, the retention probability is

$$p_{\ell,j}^{\text{keep}} = p_{\min} + (p_{\max} - p_{\min}) \left[\frac{1 + \cos\left(\pi \frac{j-1}{r_\ell - 1}\right)}{2} \right]^{\gamma_{\text{drop}}}$$

Here, $p_{\max}$ and $p_{\min}$ specify the retention range, and γ_{drop} controls how quickly retention decreases toward later modes. BOLT retains the leading active mode deterministically by setting $p_{\ell,1}^{\text{keep}} = 1$, while the cosine schedule determines the retention probabilities of the remaining modes.

For each layer, BOLT samples a shared $m \sim \mathrm{Uniform}(0,1)$ and defines the binary mode mask as

$$\xi_{\ell,j} = \mathbb{I}\left(m \le p_{\ell,j}^{\mathrm{keep}}\right)$$

where $\mathbb{I}(\cdot)$ is the indicator function, equal to 1 when its argument is true and 0 otherwise. Since

$$p_{\ell,1}^{\mathrm{keep}} \ge p_{\ell,2}^{\mathrm{keep}} \ge \cdots \ge p_{\ell,r_\ell}^{\mathrm{keep}}$$

the resulting mask satisfies

$$\xi_{\ell,1} \ge \xi_{\ell,2} \ge \cdots \ge \xi_{\ell,r_\ell}$$

and therefore always retains a valid ordered prefix.

C Theoretical Analysis

Let $r_\ell^+ \le r_\ell$ denote the current active rank of structural layer ℓ. We define the feasible prefix set

$$\mathcal{K}_B = \left\{\mathbf{k} : k_\ell \in \{k_\ell^{\min}, \ldots, r_\ell^+\}\ \forall \ell \in \mathcal{S}, \quad P_{\mathrm{phys}}(\mathbf{k}) \le B\right\}$$

Assumption 1 (Joint feasibility within the candidate search)**.** *Let $\mathcal{C}_{RS} \subseteq \mathcal{K}_B$ denote the finite set of budget-feasible prefix configurations evaluated by the rank selection candidate search. We assume that $\exists\, \mathbf{k} \in \mathcal{C}_{RS}$ such that $\Delta\mathcal{L}_{\mathrm{CE}}^{\mathrm{rep}}(\mathbf{k}) \le \delta_{\mathrm{rep}}$ and $\mathcal{D}_{\mathrm{KL}}(\mathbf{k}) \le \epsilon_{\mathrm{KL}}$.*

For compactness, let

$$\Delta\mathcal{L}_{\mathrm{train}}(\mathbf{k}) = \mathcal{L}_{\mathrm{sup}}\left(\Theta_{\mathbf{k}}; \mathcal{D}_{\mathrm{train}}\right) - \mathcal{L}_{\mathrm{sup}}\left(\Theta; \mathcal{D}_{\mathrm{train}}\right)$$

denote the supervised-loss change appearing in Eq. 1. Accordingly,

$$\mathbf{k}_{\mathrm{true}}^\star \in \arg\min_{\mathbf{k} \in \mathcal{K}_B} \Delta\mathcal{L}_{\mathrm{train}}(\mathbf{k})$$

is the best budget-feasible prefix under the empirical training-loss objective. All prefix configurations and associated loss quantities are defined with respect to the fixed model and mode ordering obtained after scale-aware canonicalization at the current rank selection invocation. Throughout this analysis, GN denotes the gate-space curvature $J^\top H J$ induced by softmax cross-entropy.

Assumption 2 (Gate-path regularity)**.** *For every feasible* $\mathbf{k}$*,*

$$t \mapsto \mathcal{L}_{\mathcal{R}}\left(g_0 + t\delta(\mathbf{k})\right)$$

is absolutely continuous on $[0,1]$*, with gate gradient existing almost everywhere.*

Assumption 3 (Finite-sample regularity)**.** *For the finite-sample analysis, conditional on the learned model and fixed training set, the rank-selection sample $\mathcal{R}$ is treated as n independent draws from the empirical training distribution on $\mathcal{D}_{\mathrm{train}}$. The per-example loss change has range width at most $W_{\mathrm{loss}} < \infty$ uniformly over $\mathcal{K}_B$.*

Positional Prefix Stability during Substructure Learning

Under positional ordered dropout in Appendix B, define

$$\bar{p}_{\ell,1} = 1 \qquad \bar{p}_{\ell,j} = p_{\ell,j}^{\mathrm{keep}}\ (j \ge 2) \qquad \bar{p}_{\ell,r_\ell+1} = 0$$

If K_ℓ is the sampled prefix rank, $\Pr(K_\ell = k) = \bar{p}_{\ell,k} - \bar{p}_{\ell,k+1}$. With independent layer-wise draws, the joint positional-prefix probability is

$$\pi(\mathbf{k}) = \prod_{\ell \in \mathcal{S}} \left(\bar{p}_{\ell,k_\ell} - \bar{p}_{\ell,k_\ell+1}\right)$$

Let

$$q(x) = \mathrm{sg}\left[\mathrm{softmax}(z^{\mathrm{full}}(x))\right] \qquad p_\xi(x) = \mathrm{softmax}(z_\xi^{\mathrm{part}}(x))$$

For fixed q,

$$-\sum_c q_c \log p_{\xi,c} = H(q) + D_{\mathrm{KL}}(q \| p_\xi)$$

so minimizing the soft-transfer term with respect to the partial prediction is equivalent to minimizing $D_{\mathrm{KL}}(q \| p_\xi)$. We define

$$\tau_{\mathrm{prefix}} = \mathbb{E}_{x,\xi} D_{\mathrm{KL}}(q \| p_\xi)$$

Proposition 1 (Positional prefix stability)**.** *During substructure learning, for every positional prefix with* $\pi(\mathbf{k}) > 0$*,*

$$\mathbb{E}_x D_{\mathrm{KL}}(q\|p_{\mathbf{k}}) \le \frac{\tau_{\mathrm{prefix}}}{\pi(\mathbf{k})}$$

and

$$\mathbb{E}_x \,\mathrm{TV}(q, p_{\mathbf{k}}) \le \sqrt{\frac{\tau_{\mathrm{prefix}}}{2\pi(\mathbf{k})}}$$

Proof. Since

$$\tau_{\mathrm{prefix}} = \sum_{\mathbf{k}'} \pi(\mathbf{k}')\mathbb{E}_x D_{\mathrm{KL}}(q\|p_{\mathbf{k}'})$$

is a sum of nonnegative terms,

$$\pi(\mathbf{k})\mathbb{E}_x D_{\mathrm{KL}}(q\|p_{\mathbf{k}}) \le \tau_{\mathrm{prefix}}$$

Pinsker's inequality followed by Jensen's inequality gives the total-variation bound. □

This applies to the positional ordering used during ordered-substructure learning. If canonicalization changes that ordering, the result does not directly apply to the resulting selected prefixes.

Directional Gauss–Newton Prefix Damage

Budget-constrained prefix rank selection uses the canonicalized mode ordering. Let $g_{\ell j}$ denote the scalar gate associated with active mode j in layer ℓ. At the current model state, the gate vector is $g_0 = \mathbf{1}$ so removing the suffix beyond k_ℓ corresponds to

$$\delta_{\ell j}(\mathbf{k}) = -\mathbb{I}(j > k_\ell)$$

We define the loss change over the replay set associated with prefix configuration $\mathbf{k}$ as

$$\Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k}) = \mathcal{L}_{\mathcal{R}}(g_0 + \delta(\mathbf{k})) - \mathcal{L}_{\mathcal{R}}(g_0) \qquad a_{\mathcal{R}} = \nabla_g \mathcal{L}_{\mathcal{R}}(g_0)$$

For sample i, let

$$J_i = \left.\frac{\partial z_i}{\partial g}\right|_{g_0} \qquad H_i = \mathrm{diag}(p_i) - p_i p_i^\top$$

and

$$G_{\mathcal{R}} = \frac{1}{|\mathcal{R}|}\sum_i J_i^\top H_i J_i$$

BOLT uses the block-diagonal approximation

$$G_{\mathrm{blk},\mathcal{R}} = \mathrm{blockdiag}_{\ell\in\mathcal{S}}(G_{\ell\ell})$$

giving the additive directional-GN surrogate

$$D^{\mathrm{GN}}_{\mathcal{R}}(\mathbf{k}) = a_{\mathcal{R}}^\top \delta(\mathbf{k}) + \frac{1}{2}\delta(\mathbf{k})^\top G_{\mathrm{blk},\mathcal{R}}\delta(\mathbf{k}) = \sum_{\ell\in\mathcal{S}} D^{\mathrm{GN}}_\ell(k_\ell)$$

For layer ℓ,

$$D^{\mathrm{GN}}_\ell(k_\ell) = a_{\mathcal{R},\ell}^\top \delta_\ell^{(k_\ell)} + \frac{1}{2}(\delta_\ell^{(k_\ell)})^\top G_{\ell\ell}\delta_\ell^{(k_\ell)}$$

Because the complete suffix is removed jointly,

$$(\delta_\ell^{(k_\ell)})^\top G_{\ell\ell}\delta_\ell^{(k_\ell)} = \sum_{u>k_\ell}\sum_{v>k_\ell}[G_{\ell\ell}]_{uv}$$

Thus, within-layer interactions among removed modes are retained, while cross-layer curvature interactions are omitted.

PATHWISE GN APPROXIMATION ERROR

For fixed $\mathbf{k}$, consider

$$g_t = g_0 + t\delta(\mathbf{k}) \qquad t \in [0,1]$$

we define the gate-gradient residual of the directional GN approximation as

$$r(t,\mathbf{k}) = \nabla_g \mathcal{L}_{\mathcal{R}}(g_t) - a_{\mathcal{R}} - tG_{\mathrm{blk},\mathcal{R}}\delta(\mathbf{k})$$

with

$$\rho_2(\mathbf{k}) = \operatorname*{ess\,sup}_{t\in[0,1]} \|r(t,\mathbf{k})\|_2$$

Proposition 2 (Pathwise GN approximation error)**.** *Under Assumption 2,*

$$\left|\Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k}) - D^{\mathrm{GN}}_{\mathcal{R}}(\mathbf{k})\right| \le \rho_2(\mathbf{k})\|\delta(\mathbf{k})\|_2$$

Proof. By absolute continuity along the gate path,

$$\Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k}) = \int_0^1 \nabla_g \mathcal{L}_{\mathcal{R}}(g_t)^\top \delta(\mathbf{k})\, dt$$

The directional GN model gives

$$D^{\mathrm{GN}}_{\mathcal{R}}(\mathbf{k}) = \int_0^1 \left(a_{\mathcal{R}} + tG_{\mathrm{blk},\mathcal{R}}\delta(\mathbf{k})\right)^\top \delta(\mathbf{k})\, dt$$

Using the residual definition,

$$\Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k}) - D^{\mathrm{GN}}_{\mathcal{R}}(\mathbf{k}) = \int_0^1 r(t,\mathbf{k})^\top \delta(\mathbf{k})\, dt$$

Therefore,

$$\begin{aligned}\left|\Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k}) - D^{\mathrm{GN}}_{\mathcal{R}}(\mathbf{k})\right| &\le \int_0^1 \|r(t,\mathbf{k})\|_2 \|\delta(\mathbf{k})\|_2\, dt \\ &\le \rho_2(\mathbf{k})\|\delta(\mathbf{k})\|_2\end{aligned}$$

which establishes the bound. □

Let

$$q_{\mathrm{rem}}(\mathbf{k}) = \sum_{\ell\in\mathcal{S}} (r^+_\ell - k_\ell)$$

Since

$$\|\delta(\mathbf{k})\|_2^2 = q_{\mathrm{rem}}(\mathbf{k})$$

we may take

$$\boxed{E_{\mathrm{model}}(\mathbf{k}) = \rho_2(\mathbf{k})\sqrt{q_{\mathrm{rem}}(\mathbf{k})}}$$

FINITE-SAMPLE AND SURROGATE ERROR

Under Assumption 3, Hoeffding's inequality and a union bound give, with probability at least $1-\eta_{\mathrm{samp}}$

$$|\Delta\mathcal{L}_{\mathrm{train}}(\mathbf{k}) - \Delta\mathcal{L}_{\mathcal{R}}(\mathbf{k})| \le \varepsilon_{\mathrm{samp}} \quad \forall \mathbf{k} \in \mathcal{K}_B$$

where $\eta_{\mathrm{samp}} \in (0,1)$ is the failure probability and

$$\boxed{\varepsilon_{\mathrm{samp}} = W_{\mathrm{loss}}\sqrt{\frac{\log(2|\mathcal{K}_B|/\eta_{\mathrm{samp}})}{2n}}}$$

is the uniform finite-sample error, with W_{loss} denoting the maximum range width of the per-example loss change over $\mathcal{K}_B$. Since

$$|\mathcal{K}_B| \le \prod_{\ell\in\mathcal{S}} R_\ell \qquad R_\ell = r^+_\ell - k^{\min}_\ell + 1$$

we have

$$\varepsilon_{\text{samp}} \le W_{\text{loss}} \sqrt{\frac{|\mathcal{S}| \log R_{\max} + \log(2/\eta_{\text{samp}})}{2n}} \qquad R_{\max} = \max_{\ell} R_{\ell}$$

Let

$$\widetilde{D}(\mathbf{k}) = \sum_{\ell \in \mathcal{S}} \widetilde{D}_{\ell}(k_{\ell})$$

denote the fixed additive surrogate-damage supplied to the Pareto solver after directional-GN estimation and any fixed adjustment used to construct the solver objective. Raw directional GN is the special case $\widetilde{D}_{\ell} = D_{\ell}^{\text{GN}}$. The corresponding surrogate-estimation error is given by

$$\boxed{\varepsilon_{\text{est}}(\mathbf{k}) = \left| D_{\mathcal{R}}^{\text{GN}}(\mathbf{k}) - \widetilde{D}(\mathbf{k}) \right|}$$

Combining the finite-sample, model-approximation, and surrogate-estimation errors gives the total theoretical error

$$E_{\text{BOLT}}(\mathbf{k}) = \varepsilon_{\text{samp}} + E_{\text{model}}(\mathbf{k}) + \varepsilon_{\text{est}}(\mathbf{k})$$

so that

$$\left| \Delta\mathcal{L}_{\text{train}}(\mathbf{k}) - \widetilde{D}(\mathbf{k}) \right| \le E_{\text{BOLT}}(\mathbf{k})$$

Therefore, $E_{\text{BOLT}}(\mathbf{k})$ provides a theoretical error bound, while the quantities $\rho_2(\mathbf{k})$ and $\varepsilon_{\text{est}}(\mathbf{k})$ are not explicitly estimated by BOLT.

Exact Surrogate Optimization and Near-Optimality

Theorem 1 (Exact optimization of the fixed BOLT surrogate)**.** *Let $\widehat{\mathbf{k}}_{\text{DP}}$ denote the prefix configuration returned by the sparse Pareto DP. Then*

$$\widetilde{D}(\widehat{\mathbf{k}}_{\text{DP}}) = \min_{\mathbf{k} \in \mathcal{K}_B} \widetilde{D}(\mathbf{k})$$

Proof. At each layer, the DP expands every affordable prefix alternative. For two partial states with parameter costs p_1, p_2 and accumulated surrogate damages d_1, d_2, if $p_1 \le p_2$ and $d_1 \le d_2$, then every feasible continuation of the second is feasible from the first with no greater accumulated damage. Removing the dominated state therefore cannot remove an optimum. □

For any feasible selected prefix $\widehat{\mathbf{k}} \in \mathcal{K}_B$, its loss regret relative to the true optimal prefix is

$$\text{Reg}(\widehat{\mathbf{k}}) = \Delta\mathcal{L}_{\text{train}}(\widehat{\mathbf{k}}) - \Delta\mathcal{L}_{\text{train}}(\mathbf{k}^{\star}_{\text{true}})$$

Theorem 2 (Error-bounded near-optimality)**.** *Under Assumptions 2–3, with probability at least $1 - \eta_{\text{samp}}$,*

$$\text{Reg}(\widehat{\mathbf{k}}_{\text{DP}}) \le E_{\text{BOLT}}(\widehat{\mathbf{k}}_{\text{DP}}) + E_{\text{BOLT}}(\mathbf{k}^{\star}_{\text{true}})$$

Proof. Using the surrogate-error bound and Theorem 1,

$$\begin{aligned} \Delta\mathcal{L}_{\text{train}}(\widehat{\mathbf{k}}_{\text{DP}}) &\le \widetilde{D}(\widehat{\mathbf{k}}_{\text{DP}}) + E_{\text{BOLT}}(\widehat{\mathbf{k}}_{\text{DP}}) \\ &\le \widetilde{D}(\mathbf{k}^{\star}_{\text{true}}) + E_{\text{BOLT}}(\widehat{\mathbf{k}}_{\text{DP}}) \\ &\le \Delta\mathcal{L}_{\text{train}}(\mathbf{k}^{\star}_{\text{true}}) + E_{\text{BOLT}}(\mathbf{k}^{\star}_{\text{true}}) + E_{\text{BOLT}}(\widehat{\mathbf{k}}_{\text{DP}}) \end{aligned}$$

Subtracting $\Delta\mathcal{L}_{\text{train}}(\mathbf{k}^{\star}_{\text{true}})$ proves the result. □

If there exists a uniform error bound $\varepsilon \ge 0$ such that

$$E_{\text{BOLT}}(\mathbf{k}) = \varepsilon_{\text{samp}} + E_{\text{model}}(\mathbf{k}) + \varepsilon_{\text{est}}(\mathbf{k}) \le \varepsilon \qquad \forall \mathbf{k} \in \mathcal{K}_B$$

then

$$\text{Reg}(\widehat{\mathbf{k}}_{\text{DP}}) \le 2\varepsilon$$

If replay validation returns a feasible final configuration $\widehat{\mathbf{k}}_{\mathrm{BOLT}} \in \mathcal{K}_B$ from the retained candidate prefixes, define its surrogate deviation from the DP optimum as

$$\Gamma_{\mathrm{sel}} = \widetilde{D}(\widehat{\mathbf{k}}_{\mathrm{BOLT}}) - \widetilde{D}(\widehat{\mathbf{k}}_{\mathrm{DP}}) \geq 0$$

Then

$$\mathrm{Reg}(\widehat{\mathbf{k}}_{\mathrm{BOLT}}) \leq \Gamma_{\mathrm{sel}} + E_{\mathrm{BOLT}}(\widehat{\mathbf{k}}_{\mathrm{BOLT}}) + E_{\mathrm{BOLT}}(\mathbf{k}^{\star}_{\mathrm{true}})$$

Under the uniform error condition,

$$\boxed{\mathrm{Reg}(\widehat{\mathbf{k}}_{\mathrm{BOLT}}) \leq \Gamma_{\mathrm{sel}} + 2\varepsilon}$$

When $\widehat{\mathbf{k}}_{\mathrm{BOLT}} = \widehat{\mathbf{k}}_{\mathrm{DP}}$, $\Gamma_{\mathrm{sel}} = 0$. The positional-prefix stability bound and GN approximation may become looser under aggressive compression, therefore validation over the replay set therefore provides an empirical safeguard when selecting the final candidate.

COMPLEXITY

Let

$$B_{\mathrm{struct}} = B - P_{\mathrm{fixed}} \qquad g = \gcd\{c_\ell : \ell \in \mathcal{S}\}$$

and

$$\overline{B} = \left\lfloor \frac{B_{\mathrm{struct}}}{g} \right\rfloor \qquad R_{\max} = \max_{\ell \in \mathcal{S}} (r_\ell^+ - k_\ell^{\min} + 1)$$

If at most K labels are retained at each normalized exact cost, at most $K(\overline{B}+1)$ states survive per layer. Since each state considers at most $R_{\max}$ prefixes, hence time complexity

$$T = O\big(K|\mathcal{S}|R_{\max}(\overline{B}+1)\big)$$

For fixed K,

$$\boxed{T = O\big(|\mathcal{S}|R_{\max}(\overline{B}+1)\big)}$$

If each state stores its partial rank allocation, reconstruction-aware memory is

$$\boxed{M = O\big(|\mathcal{S}|K(\overline{B}+1)\big)}$$

while storing only scalar frontier values requires $O\big(K(\overline{B}+1)\big)$ memory when paths are reconstructed separately. Equivalently, shifting $h_\ell = k_\ell - k_\ell^{\min}$ reserves $P_{\min} = \sum_{\ell \in \mathcal{S}} c_\ell k_\ell^{\min}$ and yields the tighter residual-budget quantity

$$\overline{B}_{\mathrm{res}} = \left\lfloor \frac{B - P_{\mathrm{fixed}} - P_{\min}}{g} \right\rfloor$$

Thus, BOLT's discrete allocation is pseudo-polynomial in the normalized budget and avoids explicit enumeration of joint prefix configurations.

$$\prod_{\ell \in \mathcal{S}} (r_\ell^+ - k_\ell^{\min} + 1)$$

The cost of estimating directional-GN damage is separate and depends on the chosen exact or stochastic Jacobian/vector-product estimator.

D DATASET

We evaluate BOLT across image and language classification benchmarks. The image datasets include MNIST (LeCun, 1998), Fashion-MNIST (Xiao et al., 2017), CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011), PathMNIST (Yang et al., 2023), and ImageNet-1K (Deng et al., 2009), while MRPC (Dolan & Brockett, 2005) and PAWS (Zhang et al., 2019) are used for sentence-pair classification.

MNIST and Fashion-MNIST contain 28×28 grayscale images from 10 classes, each with 60,000 training and 10,000 test samples. CIFAR-10 and CIFAR-100 contain 32×32 RGB images with

Table 2: Dataset partitions used in the experiments. MNIST, Fashion-MNIST, CIFAR-10, CIFAR-100, SVHN, PathMNIST, and MRPC use a fixed-seed 90/10 training-validation split, with the predefined test set retained for held-out evaluation. PAWS and Multi30K use their predefined splits, while MNLI uses its predefined training and validation splits. The GLUE test split is excluded from local evaluation because its ground-truth labels are unavailable for direct scoring. For ImageNet-1K, 500,000 stratified training images are split into 490,000 for training and 10,000 for internal validation, with the official 50,000-image validation set retained for held-out evaluation.

Dataset	**Classes**	**Original train**	**Training**	**Validation**	**Held-out test/eval.**
MNIST	10	60,000	54,000	6,000	10,000
Fashion-MNIST	10	60,000	54,000	6,000	10,000
CIFAR-10	10	50,000	45,000	5,000	10,000
CIFAR-100	100	50,000	45,000	5,000	10,000
SVHN	10	73,257	65,931	7,326	26,032
PathMNIST	9	89,996	80,996	9,000	7,180
ImageNet-1K	1,000	1,281,167	490,000	10,000	50,000
MRPC	2	3,668	3,301	367	408
PAWS	2	49,401	49,401	8,000	8,000
MNLI	3	392,702	392,702	9,815	–
Multi30K	–	29,000	29,000	1,014	1,000

50,000 training and 10,000 test samples across 10 and 100 classes, respectively. Meanwhile SVHN contains 32×32 RGB street-number images from 10 classes. PathMNIST is a nine-class colorectal histopathology dataset. ImageNet-1K contains 1,000 object classes with 1,281,167 training images and 50,000 validation images.

Language evaluation includes MRPC as a binary semantic-equivalence benchmark, MNLI as a three-class natural-language-inference benchmark, and PAWS as a binary paraphrase-identification benchmark containing sentence pairs with high lexical overlap but potentially different meanings. We use the `google-research-datasets/paws` dataset with the predefined `train` split for training and the `validation` split for evaluation, with accuracy as the primary metric.

We evaluate machine translation using the Multi30K English–German parallel dataset (Elliott et al., 2016). Each example contains an English source sentence and its corresponding German translation. Our experiments use the predefined splits consisting of 29,000 training sentence pairs, 1,014 validation sentence pairs, and 1,000 test sentence pairs.

In the case of MNIST, Fashion-MNIST, CIFAR-10, CIFAR-100, SVHN, PathMNIST, and MRPC, we partition the designated training split using a fixed seed, with 90% of the samples used for training and the remaining 10% reserved for internal validation. The dataset's predefined test or evaluation split is kept separate from both subsets and used as a held-out set for test evaluation. PAWS, MNLI, and Multi30K use their predefined training and validation splits, with Multi30K additionally retaining its predefined test split for final translation evaluation. For ImageNet-1K, we use stratified sampling to select 500,000 images from the official training split, of which 490,000 are used for training and 10,000 are reserved for internal validation. The full 50,000-image official validation split is kept separate as a held-out evaluation set. The experimental dataset partitions are summarized in Table 2.

E Baseline Implementation

Our baseline implementations are based on the publicly available code released by the respective authors: Cuttlefish (Wang et al., 2023), Pufferfish (Wang et al., 2021), XNOR-Net (Rastegari et al., 2016), SI&FD (Khodak et al., 2021), IMP (Frankle & Carbin, 2018), and MAESTRO (Horváth et al., 2023).

All methods use the same experimental interface. For each dataset, models are trained on the same training split and evaluated on the same held-out unseen test split. The test split is not used for model selection, pruning, rank selection, or compaction, preventing data leakage. Each method is trained for the same fixed number of epochs with the same preprocessing, batch size, optimizer setting where applicable, and seeded randomness for reproducibility.

For fair comparison, compression is reported using physically materialized compact models. Baselines that produce logical compression through masks, binarization, low-rank factors, or thresholded structures are physically compacted only after training, and the compact model is evaluated using the same validation and test loaders. This does not affect their training trajectory. In contrast, BOLT performs budget-aware projection during training and can physically compact the ordered low-rank model after each epoch.

In our implementation, MAESTRO, Pufferfish, XNOR-Net, and BOLT are evaluated on both ResNet-18 and VGG-19. Cuttlefish, SI&FD, and IMP are evaluated on ResNet-based settings.

F Model Architecture

We evaluate the methods across both CNN and Transformer-based architectures to assess their applicability to different model families and data modalities. The CNN backbones consist of VGG-19 (Simonyan & Zisserman, 2014) and ResNet-18, ResNet-50 (He et al., 2016) while the Transformer-based experiments use BERT-base-cased (Devlin et al., 2019), DeBERTa-v3-base (He et al., 2021), ViT-Huge/14 (Dosovitskiy et al., 2020), DINOv2-Giant (Oquab et al., 2023), and M2M-100 1.2B (Fan et al., 2021). The major layer-wise parameter shapes and architectural hyper-parameters of these models are summarized below.

F.1 CNN Architectures

For the VGG-19 and ResNet-18 experiments, we use the same optimization configuration unless otherwise specified. Both models are trained for 100 epochs using SGD with an initial learning rate of 0.01, momentum of 0.9, and weight decay of 10^{-4}. The learning rate is reduced by a factor of 0.1 at 50% and 75% of the training horizon, corresponding to epochs 50 and 75 in the 100-epoch runs.

The VGG-19 and CIFAR-style ResNet-18 architectures used in the experiments are detailed in Tables 3 and 4, respectively. For the CNN architectures, C denotes the number of output classes for the corresponding dataset. Tensor dimensions follow the PyTorch parameter-storage convention. Convolutional kernels are represented as $C_{\text{out}} \times C_{\text{in}} \times k_h \times k_w$, while linear-layer weights are represented as $d_{\text{out}} \times d_{\text{in}}$.

F.2 Transformer Architectures

Transformer-based experiments instantiate all models directly from their corresponding Hugging Face checkpoints using the Transformers library (Wolf et al., 2020). Language classification uses BERT-base-cased (Devlin et al., 2019), instantiated from `google-bert/bert-base-cased`, and DeBERTa-v3-base (He et al., 2021), instantiated from `microsoft/deberta-v3-base`. BERT-base-cased contains 12 Transformer encoder layers with a hidden dimension of (768) and 12 attention heads. DeBERTa-v3-base also contains 12 encoder layers with a hidden dimension of (768) and 12 attention heads, together with disentangled relative-position attention.

Machine translation uses M2M-100 1.2B (Fan et al., 2021), instantiated from `facebook/m2m100_1.2B`. M2M-100 1.2B is a multilingual encoder–decoder Transformer containing 24 encoder layers and 24 decoder layers, with a hidden dimension of (1024), 16 attention heads in each layer, and a feed-forward dimension of (8192).

For vision experiments, we use ViT-Huge/14 (Dosovitskiy et al., 2020), instantiated from `google/vit-huge-patch14-224-in21k`, and DINOv2-Giant (Oquab et al., 2023), instantiated from `facebook/dinov2-giant-imagenet1k-1-layer`. ViT-Huge/14 contains 32 Transformer encoder layers with a hidden dimension of 1280, 16 attention heads, and 14×14 patch embeddings. Meanwhile, DINOv2-Giant uses a Vision Transformer backbone with 40 encoder layers, a hidden dimension of 1536, 24 attention heads, and 14×14 patch embeddings, together with an image-classification head. For all Transformer experiments, we use the model configurations provided directly with their corresponding Hugging Face checkpoints.

Parameter	Shape	Layer hyper-parameter
layer1.conv1.weight	$64 \times 3 \times 3 \times 3$	stride:1; padding:1
layer2.conv2.weight	$64 \times 64 \times 3 \times 3$	stride:1; padding:1
pooling.max	N/A	kernel size:2; stride:2
layer3.conv3.weight	$128 \times 64 \times 3 \times 3$	stride:1; padding:1
layer4.conv4.weight	$128 \times 128 \times 3 \times 3$	stride:1; padding:1
pooling.max	N/A	kernel size:2; stride:2
layer5.conv5.weight	$256 \times 128 \times 3 \times 3$	stride:1; padding:1
layer6.conv6.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
layer7.conv7.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
layer8.conv8.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
pooling.max	N/A	kernel size:2; stride:2
layer9.conv9.weight	$512 \times 256 \times 3 \times 3$	stride:1; padding:1
layer10.conv10.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer11.conv11.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer12.conv12.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
pooling.max	N/A	kernel size:2; stride:2
layer13.conv13.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer14.conv14.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer15.conv15.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer16.conv16.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
pooling.avg	N/A	kernel size:2
classifier.weight	$C \times 512$	C output classes
classifier.bias	C	N/A

Table 3: Major layer-wise parameter shapes and hyper-parameters of the VGG-19 architecture used in the experiments. Here C denotes the number of output classes.

G ELIGIBLE LAYERS FOR BOLT COMPACTION

Table 5: Layers selected for BOLT factorization and compaction

Architecture	BOLT-eligible layers	Not factorized by BOLT
Fully connected	Linear layers (`nn.Linear`)	Activations, normalization, dropout, and other non-target parameters.
CNN	Convolutional (`nn.Conv2d`) and linear (`nn.Linear`) layers	Normalization, activations, pooling, dropout, and other non-target parameters.
Transformer	Attention and feed-forward network (FFN) linear layers (`nn.Linear`), including query (Q), key (K), value (V), attention-output, and FFN projections.	Embeddings, normalization layers, activations, positional or class tokens, poolers, patch embeddings, and task-specific heads.

BOLT performs low-rank factorization and subsequent rank-selection-based compaction on a selected set of eligible layers $\mathcal{S}$. Each eligible weight tensor is represented in matrix form and decomposed into an ordered sum of learned rank-one modes. Linear layers are factorized directly, while convolutional kernels are unfolded into matrix form before factorization. BOLT then selects a prefix rank for each eligible layer and physically removes the remaining suffix modes.

Table 5 summarizes the layer types considered for BOLT compaction. Fully connected networks expose their linear layers to BOLT, while CNN additionally include convolutional kernels. In Transformer architectures (Vaswani et al., 2017), each block contains multi-head attention (MHA) and position-wise feed-forward network (FFN) components. BOLT factorizes the trainable linear weight matrices in these components, including the query (Q), key (K), value (V), and attention-output projections in MHA and the linear projections in the FFN. These `nn.Linear` weights use the same ordered low-rank factorization as fully connected layers.

Parameter	Shape	Layer hyper-parameter
conv1.weight	$64 \times 3 \times 3 \times 3$	stride:1; padding:1
layer1.block1.conv1.weight	$64 \times 64 \times 3 \times 3$	stride:1; padding:1
layer1.block1.conv2.weight	$64 \times 64 \times 3 \times 3$	stride:1; padding:1
layer1.block2.conv1.weight	$64 \times 64 \times 3 \times 3$	stride:1; padding:1
layer1.block2.conv2.weight	$64 \times 64 \times 3 \times 3$	stride:1; padding:1
layer2.block1.conv1.weight	$128 \times 64 \times 3 \times 3$	stride:2; padding:1
layer2.block1.conv2.weight	$128 \times 128 \times 3 \times 3$	stride:1; padding:1
layer2.block1.downsample.weight	$128 \times 64 \times 1 \times 1$	stride:2; padding:0
layer2.block2.conv1.weight	$128 \times 128 \times 3 \times 3$	stride:1; padding:1
layer2.block2.conv2.weight	$128 \times 128 \times 3 \times 3$	stride:1; padding:1
layer3.block1.conv1.weight	$256 \times 128 \times 3 \times 3$	stride:2; padding:1
layer3.block1.conv2.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
layer3.block1.downsample.weight	$256 \times 128 \times 1 \times 1$	stride:2; padding:0
layer3.block2.conv1.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
layer3.block2.conv2.weight	$256 \times 256 \times 3 \times 3$	stride:1; padding:1
layer4.block1.conv1.weight	$512 \times 256 \times 3 \times 3$	stride:2; padding:1
layer4.block1.conv2.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer4.block1.downsample.weight	$512 \times 256 \times 1 \times 1$	stride:2; padding:0
layer4.block2.conv1.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
layer4.block2.conv2.weight	$512 \times 512 \times 3 \times 3$	stride:1; padding:1
pooling.avg	N/A	global average pooling
classifier.weight	$C \times 512$	C output classes
classifier.bias	C	N/A

Table 4: Major layer-wise parameter shapes and hyper-parameters of the CIFAR-style ResNet-18 architecture. Here C denotes the number of output classes. Normalization and bias parameters are omitted except for the classification bias.

Parameters outside $\mathcal{S}$ are not subject to low-rank factorization or rank selection and therefore remain unchanged during BOLT compaction. These parameters contribute to P_{fixed} in the physical parameter budget.

H Data-driven Supervised and Prefix-consistency Objectives

BOLT retains the original supervised learning objective of the underlying network and augments it during ordered-substructure learning with a prefix-consistency objective. The supervised loss L_{sup} in Eq. 3 is instantiated according to the predictive setting. In each case, the full ordered model is optimized using ground-truth supervision while a prefix-restricted model obtained through ordered dropout is trained to preserve the predictive distribution of the current full model. Consequently, the learned ordered prefix structure is driven by predictive behavior on training data rather than by weight magnitude alone.

Language sequence classification. BOLT considers categorical natural language processing (NLP) settings that include both single-sentence and sentence-pair classification, with supervision defined by standard cross-entropy. Let $z_i^{\text{full}} \in \mathbb{R}^C$ denote the class logits and $y_i \in \{1, \dots, C\}$ the ground-truth class. Then the supervised objective is:

$$L_{\text{sup}}^{\text{NLP}} = -\frac{1}{N} \sum_{i=1}^{N} \log \left[\text{softmax}\left(z_i^{\text{full}}\right)\right]_{y_i} \tag{14}$$

For the ordered-prefix pass, BOLT forms the stop-gradient teacher distribution $q_i = \text{sg}\big(\text{softmax}\left(z_i^{\text{full}}\right)\big)$ and minimizes

$$L_{\text{soft}}^{\text{NLP}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} q_{i,c} \log \left[\text{softmax}\left(z_i^{\text{part}}\right)\right]_c \tag{15}$$

Thus, the full model learns from labelled examples, while randomly sampled ordered prefixes are directly trained to reproduce its data-dependent predictive behavior.

Vision Transformer classification. BOLT uses the same formulation for ViT image classification, with X_i denoting an input image and $f_\Theta(X_i)$ denoting the class logits produced by the full ordered model. The supervised objective is

$$L_{\text{sup}}^{\text{ViT}} = -\frac{1}{N}\sum_{i=1}^{N} \log\left[\text{softmax}\left(f_\Theta(X_i)\right)\right]_{y_i} \tag{16}$$

The full model provides the stop-gradient teacher distribution $q_i = \text{sg}(\text{softmax}\left(f_\Theta(X_i)\right))$ and the prefix-consistency loss is

$$L_{\text{soft}}^{\text{ViT}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} q_{i,c} \log\left[\text{softmax}\left(f_{\Theta,\xi}(X_i)\right)\right]_c \tag{17}$$

Ordered dropout acts on the factorized Transformer projections, while the prefix-learning signal is supplied at the model-output level.

Autoregressive translation. In the case of translation, let X_i denote the source sequence, $Y_i = (y_{i,1}, \ldots, y_{i,T_i})$ the target sequence, and Ω_i the set of non-padding target positions. Under teacher forcing, let $z_{i,t}^{\text{full}} \in \mathbb{R}^{|\mathcal{V}|}$ denote the vocabulary logits for predicting target token $y_{i,t}$, conditioned on the source sequence X_i and the preceding target tokens $y_{i,<t}$. Then the supervised objective is the mean token-level negative log-likelihood over all valid target positions:

$$L_{\text{sup}}^{\text{MT}} = -\frac{1}{N_{\text{tok}}}\sum_i \sum_{t\in\Omega_i} \log\left[\text{softmax}\left(z_{i,t}^{\text{full}}\right)\right]_{y_{i,t}} \qquad N_{\text{tok}} = \sum_i |\Omega_i| \tag{18}$$

The full model supplies the tokenwise stop-gradient teacher distribution $q_{i,t} = \text{sg}\big(\text{softmax}\left(z_{i,t}^{\text{full}}\right)\big)$ and prefix consistency is imposed over the same non-padding target positions:

$$L_{\text{soft}}^{\text{MT}} = \frac{1}{N_{\text{tok}}}\sum_i \sum_{t\in\Omega_i} D_{\text{KL}}\left(q_{i,t} \,\|\, \text{softmax}\left(z_{i,t}^{\text{part}}\right)\right) \tag{19}$$

Since $q_{i,t}$ is held fixed during differentiation, $D_{\text{KL}}\big(q_{i,t}\|p_{i,t}^{\text{part}}\big)$ differs from soft-target cross-entropy only by the entropy of the fixed teacher distribution.

Data-driven loss change estimation for rank selection. The same predictive outputs used to learn the ordered prefix structure also define the local supervised-loss geometry used during rank selection. Categorical NLP and ViT classification associate each replay example with a single categorical prediction. Meanwhile for translation, BOLT selects a valid non-padding target position from each replay sequence and treats the corresponding next-token prediction as one categorical replay unit. Let i index such a replay predictive unit, let z_i denote its output logits, and let y_i denote its class or target-token label. With predictive distribution $p_u = \text{softmax}(z_u)$, the supervised replay loss can be written uniformly as

$$\mathcal{L}_{\text{sup}}^{\text{rep}} = -\frac{1}{|\mathcal{U}_{\text{rep}}|}\sum_{u\in\mathcal{U}_{\text{rep}}} \log[p_u]_{y_u} \tag{20}$$

where a predictive unit corresponds to an input example for classification and to a selected non-padding next-token prediction for translation. BOLT utilizes the replay loss to estimate the effect of removing each candidate suffix through temporary structural gates as described in section 4.2.

I Experimental Results

The training accuracy plots for BOLT and the baseline methods across different datasets and architectures are presented in Fig. 7. For the ResNet-18 and VGG-19 experiments, BOLT was trained for 100 epochs over five random seeds, with up to 15% of the initial epochs reserved for warm-up. Overall, the plots show that BOLT retains sufficient model capacity throughout training under compression. In several settings, its training accuracy approaches or closely matches the near-saturated performance of the corresponding dense models, often reaching values close to 100%. This indicates that the proposed ordered low-rank compression preserves sufficient representational capacity to fit the training data effectively. The corresponding held-out test accuracy and efficiency results are reported in Table 6.

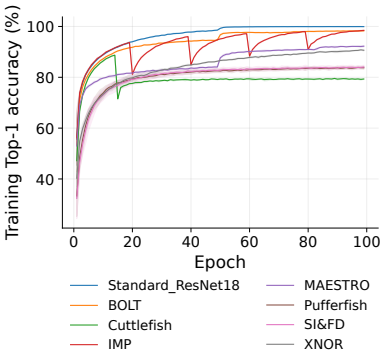
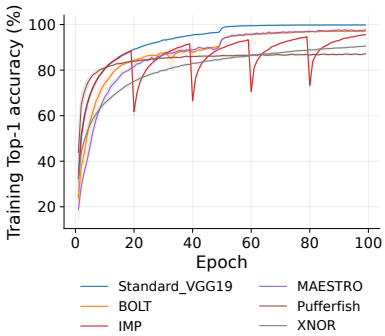
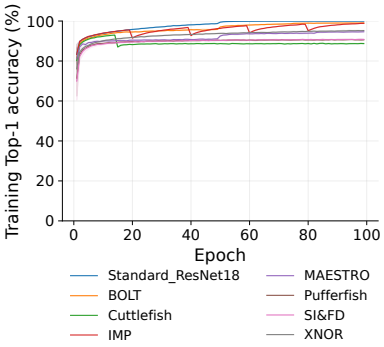
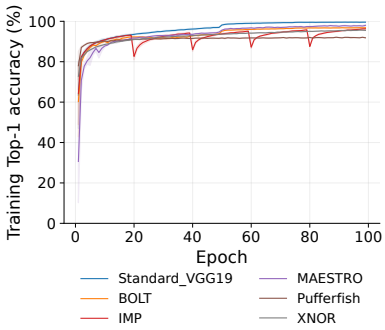
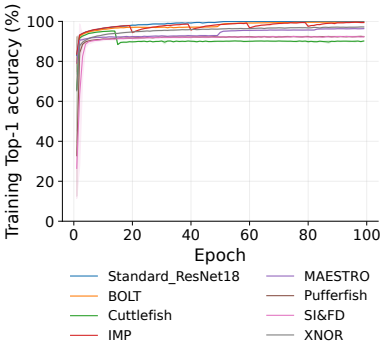
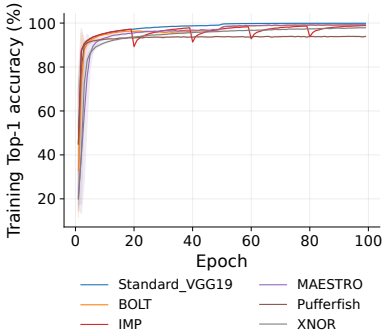
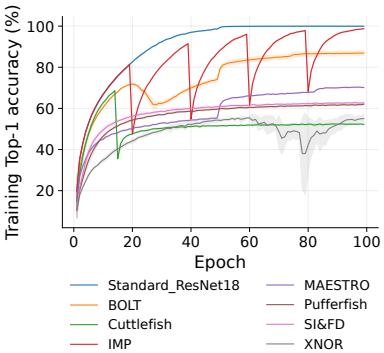
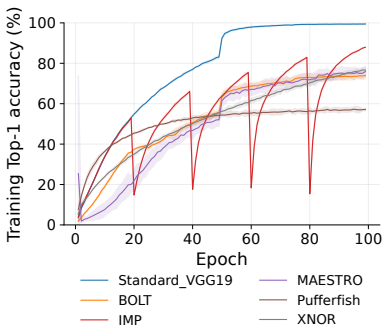
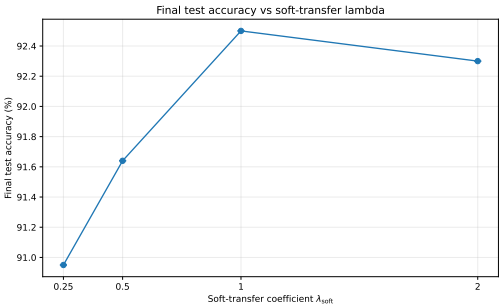
Dataset	ResNet-18	VGG-19
CIFAR-10		
Fashion-MNIST		
SVHN		
CIFAR-100		

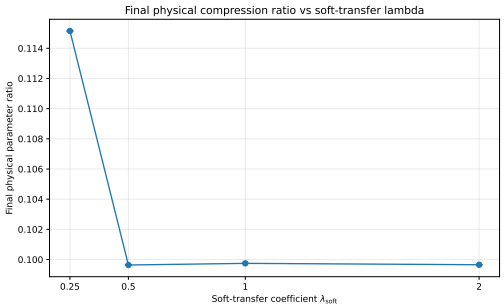
Figure 7: Training accuracy of ResNet-18 and VGG-19 across the evaluated datasets.

Table 6: Accuracy and efficiency comparison across datasets and architectures. Results are reported as mean ± standard deviation. FLOPs are reported in GFLOPs/image, P95 latency in milliseconds, and peak memory usage in MiB. Unavailable or unsupported results are denoted by (–). BOLT (ours) is highlighted in light gray.

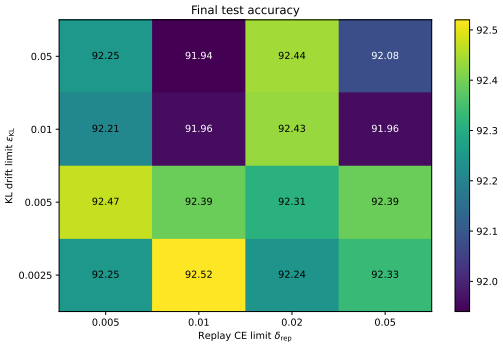
		ResNet-18					VGG-19				
Dataset	**Method**	**Top-1 Acc. (%)**	**Compact Ratio**	**FLOPs (GFLOPs/img)**	**P95 Lat. (ms)**	**Peak Mem. (MiB)**	**Top-1 Acc. (%)**	**Compact Ratio**	**FLOPs (GFLOPs/img)**	**P95 Lat. (ms)**	**Peak Mem. (MiB)**
CIFAR-10	Standard	92.69 ± 0.16	1.000 ± 0.000	1.111 ± 0.000	2.869 ± 0.021	221.57 ± 0.00	89.75 ± 0.20	1.000 ± 0.000	0.796 ± 0.000	1.599 ± 0.005	191.30 ± 0.00
	BOLT	92.37 ± 0.21	0.100 ± 0.000	0.464 ± 0.006	3.509 ± 0.153	195.55 ± 10.83	90.44 ± 0.14	0.099 ± 0.001	0.374 ± 0.004	2.124 ± 0.021	127.77 ± 15.91
	Cuttlefish	70.67 ± 6.00	0.507 ± 0.028	0.751 ± 0.013	3.239 ± 0.025	200.56 ± 1.18	–	–	–	–	–
	IMP	90.35 ± 0.23	1.000 ± 0.000	1.111 ± 0.000	2.806 ± 0.025	221.57 ± 0.00	87.38 ± 0.17	1.000 ± 0.000	0.796 ± 0.000	1.559 ± 0.002	191.30 ± 0.00
	MAESTRO	90.39 ± 0.24	0.041 ± 0.000	0.109 ± 0.001	4.342 ± 0.150	180.55 ± 0.02	90.43 ± 0.09	0.758 ± 0.000	0.741 ± 0.001	3.365 ± 0.031	172.96 ± 0.04
	Pufferfish	77.04 ± 3.44	0.299 ± 0.000	0.432 ± 0.000	2.758 ± 0.024	191.56 ± 0.00	76.67 ± 3.54	0.405 ± 0.000	0.578 ± 0.000	1.841 ± 0.002	145.87 ± 0.00
	SI&FD	77.67 ± 4.16	0.543 ± 0.000	0.620 ± 0.000	3.480 ± 0.023	201.95 ± 0.00	–	–	–	–	–
	XNOR	66.43 ± 5.48	1.000 ± 0.000	1.111 ± 0.000	3.386 ± 0.013	254.06 ± 0.00	72.48 ± 1.46	1.000 ± 0.000	0.796 ± 0.000	2.115 ± 0.003	223.87 ± 0.00
CIFAR-100	Standard	72.43 ± 0.20	1.000 ± 0.000	1.111 ± 0.000	2.797 ± 0.003	221.74 ± 0.00	59.14 ± 0.73	1.000 ± 0.000	0.796 ± 0.000	1.581 ± 0.010	191.47 ± 0.00
	BOLT	70.53 ± 0.29	0.100 ± 0.000	0.412 ± 0.051	3.263 ± 0.135	183.07 ± 0.01	60.67 ± 0.36	0.099 ± 0.000	0.308 ± 0.008	2.123 ± 0.047	132.08 ± 13.13
	Cuttlefish	43.41 ± 0.84	0.693 ± 0.046	0.776 ± 0.022	3.184 ± 0.035	208.61 ± 1.96	–	–	–	–	–
	IMP	69.18 ± 0.44	1.000 ± 0.000	1.111 ± 0.000	2.721 ± 0.019	221.74 ± 0.00	52.27 ± 0.97	1.000 ± 0.000	0.796 ± 0.000	1.549 ± 0.003	191.47 ± 0.00
	MAESTRO	65.48 ± 0.18	0.063 ± 0.000	0.134 ± 0.001	4.187 ± 0.052	181.49 ± 0.01	59.61 ± 0.64	0.764 ± 0.004	0.743 ± 0.005	3.324 ± 0.029	173.57 ± 0.27
	Pufferfish	54.90 ± 0.25	0.302 ± 0.000	0.433 ± 0.000	2.682 ± 0.017	191.74 ± 0.00	45.03 ± 4.37	0.406 ± 0.000	0.578 ± 0.000	1.833 ± 0.003	146.05 ± 0.00
	SI&FD	52.45 ± 1.61	0.545 ± 0.000	0.620 ± 0.000	3.386 ± 0.012	202.13 ± 0.00	–	–	–	–	–
	XNOR	23.89 ± 7.35	1.000 ± 0.000	1.111 ± 0.000	3.304 ± 0.005	254.23 ± 0.00	27.61 ± 1.83	1.000 ± 0.000	0.796 ± 0.000	2.107 ± 0.002	224.04 ± 0.00
FashionMNIST	Standard	94.78 ± 0.05	1.000 ± 0.000	1.111 ± 0.000	2.827 ± 0.006	221.57 ± 0.00	93.92 ± 0.09	1.000 ± 0.000	0.796 ± 0.000	1.629 ± 0.001	191.30 ± 0.00
	BOLT	94.81 ± 0.10	0.099 ± 0.001	0.455 ± 0.009	3.375 ± 0.074	187.96 ± 8.51	94.06 ± 0.12	0.099 ± 0.001	0.265 ± 0.140	1.854 ± 0.602	124.27 ± 17.66
	Cuttlefish	85.61 ± 3.19	0.420 ± 0.036	0.703 ± 0.070	3.035 ± 0.108	196.86 ± 1.53	–	–	–	–	–
	IMP	93.75 ± 0.10	1.000 ± 0.000	1.111 ± 0.000	2.765 ± 0.009	221.57 ± 0.00	93.17 ± 0.24	1.000 ± 0.000	0.796 ± 0.000	1.591 ± 0.005	191.30 ± 0.00
	MAESTRO	93.93 ± 0.22	0.028 ± 0.000	0.066 ± 0.002	4.352 ± 0.025	179.99 ± 0.02	93.85 ± 0.21	0.608 ± 0.001	0.640 ± 0.000	3.382 ± 0.027	161.50 ± 0.06
	Pufferfish	86.98 ± 1.18	0.299 ± 0.000	0.432 ± 0.000	2.714 ± 0.008	191.56 ± 0.00	89.37 ± 1.38	0.405 ± 0.000	0.578 ± 0.000	1.866 ± 0.001	145.87 ± 0.00
	SI&FD	88.40 ± 1.44	0.543 ± 0.000	0.620 ± 0.000	3.390 ± 0.008	201.95 ± 0.00	–	–	–	–	–
	XNOR	47.99 ± 18.01	1.000 ± 0.000	1.111 ± 0.000	3.335 ± 0.004	254.06 ± 0.00	85.47 ± 2.70	1.000 ± 0.000	0.796 ± 0.000	2.131 ± 0.004	223.87 ± 0.00
MNIST	Standard	99.64 ± 0.02	1.000 ± 0.000	1.111 ± 0.000	2.855 ± 0.017	221.57 ± 0.00	99.59 ± 0.02	1.000 ± 0.000	0.796 ± 0.000	1.655 ± 0.002	191.30 ± 0.00
	BOLT	99.63 ± 0.05	0.099 ± 0.000	0.439 ± 0.042	3.412 ± 0.174	188.28 ± 9.09	99.62 ± 0.04	0.100 ± 0.000	0.338 ± 0.019	2.119 ± 0.035	120.91 ± 11.16
	Cuttlefish	96.35 ± 1.87	0.532 ± 0.207	0.690 ± 0.094	3.024 ± 0.036	201.62 ± 8.82	–	–	–	–	–
	IMP	99.62 ± 0.03	1.000 ± 0.000	1.111 ± 0.000	2.790 ± 0.016	221.57 ± 0.00	99.48 ± 0.08	1.000 ± 0.000	0.796 ± 0.000	1.617 ± 0.008	191.30 ± 0.00
	MAESTRO	99.61 ± 0.05	0.022 ± 0.000	0.045 ± 0.003	4.265 ± 0.033	179.74 ± 0.00	40.77 ± 50.96	0.595 ± 0.006	0.620 ± 0.009	3.335 ± 0.042	160.52 ± 0.43
	Pufferfish	98.49 ± 0.58	0.299 ± 0.000	0.432 ± 0.000	2.747 ± 0.014	191.56 ± 0.00	96.86 ± 1.71	0.405 ± 0.000	0.578 ± 0.000	1.887 ± 0.006	145.87 ± 0.00
	SI&FD	98.44 ± 0.44	0.543 ± 0.000	0.620 ± 0.000	3.443 ± 0.026	201.95 ± 0.00	–	–	–	–	–
	XNOR	53.25 ± 21.58	1.000 ± 0.000	1.111 ± 0.000	3.354 ± 0.006	254.06 ± 0.00	90.32 ± 13.28	1.000 ± 0.000	0.796 ± 0.000	2.135 ± 0.004	223.87 ± 0.00
SVHN	Standard	96.18 ± 0.04	1.000 ± 0.000	1.111 ± 0.000	2.831 ± 0.003	221.57 ± 0.00	96.19 ± 0.02	1.000 ± 0.000	0.796 ± 0.000	1.616 ± 0.003	191.30 ± 0.00
	BOLT	96.25 ± 0.17	0.099 ± 0.000	0.386 ± 0.031	3.592 ± 0.178	183.02 ± 0.01	96.39 ± 0.08	0.100 ± 0.000	0.343 ± 0.010	2.086 ± 0.044	128.09 ± 13.53
	Cuttlefish	81.21 ± 2.81	0.472 ± 0.021	0.730 ± 0.027	3.021 ± 0.142	199.07 ± 0.90	–	–	–	–	–
	IMP	95.53 ± 0.16	1.000 ± 0.000	1.111 ± 0.000	2.774 ± 0.002	221.57 ± 0.00	95.50 ± 0.10	1.000 ± 0.000	0.796 ± 0.000	1.572 ± 0.004	191.30 ± 0.00
	MAESTRO	96.41 ± 0.08	0.030 ± 0.001	0.062 ± 0.001	5.840 ± 1.300	180.09 ± 0.02	96.48 ± 0.11	0.463 ± 0.000	0.546 ± 0.001	3.324 ± 0.048	150.44 ± 0.03
	Pufferfish	90.70 ± 2.58	0.299 ± 0.000	0.432 ± 0.000	2.702 ± 0.002	191.56 ± 0.00	92.47 ± 0.49	0.405 ± 0.000	0.578 ± 0.000	1.838 ± 0.001	145.87 ± 0.00
	SI&FD	91.47 ± 0.51	0.543 ± 0.000	0.620 ± 0.000	3.403 ± 0.013	201.95 ± 0.00	–	–	–	–	–
	XNOR	90.31 ± 1.25	1.000 ± 0.000	1.111 ± 0.000	3.322 ± 0.005	254.06 ± 0.00	88.91 ± 0.29	1.000 ± 0.000	0.796 ± 0.000	2.118 ± 0.001	223.87 ± 0.00



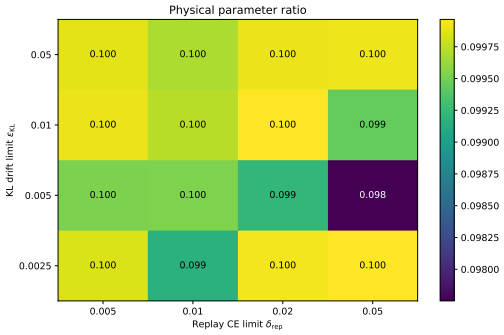
(a) Test accuracy versus λ_{soft}.



(b) Physical parameter ratio versus λ_{soft}.



(c) Test accuracy across the CE and KL limits.



(d) Physical parameter ratio across the CE and KL limits.

Figure 8: Sensitivity of BOLT on CIFAR-10 with ResNet-18 to the soft-transfer weight λ_{soft} and the drift limits. The top row shows test accuracy and the final physical parameter ratio as λ_{soft} varies. The bottom row shows the same metrics as the replay CE limit δ_{rep} and KL drift limit ϵ_{KL} vary jointly.

Table 7: Replay CE and KL drift limits used in BOLT experiments.

Dataset	Architecture	δ_{rep}	ϵ_{KL}
CIFAR-10, CIFAR-100, MNIST, Fashion-MNIST, SVHN	ResNet-18, ResNet-50, VGG-19	0.02	0.01
PathMNIST	ResNet-50, ResNet-18, VGG-19	0.02	0.01
MRPC, MNLI, PAWS	BERT-base-cased, DeBERTa-v3-base	0.08	0.02
ImageNet-1K	ViT-Huge/14, DINOv2-Giant	0.08	0.02
Multi30k	M2M100-1.2B	0.08	0.02

I.1 Configuring λ_{soft}, δ_{rep}, and ϵ_{KL}

The ablation study shows that removing the CE and KL drift checks reduces accuracy even when the final model remains near the same physical parameter budget. We therefore examine the replay CE limit δ_{rep} and KL drift limit ϵ_{KL} jointly. In the sensitivity sweep (Fig. 8), the tested combinations achieve physical parameter ratios close to the target, while their accuracies vary with the choice of limits. The resulting accuracy and model size tradeoffs show that consistently tightening or relaxing both limits does not produce the best result. During the prefix substructure learning sweep, a small λ_{soft} yields lower accuracy, and the model at 0.25 also exceeds the target physical ratio. Increasing the coefficient to 1 gives the highest observed accuracy while meeting the budget. Increasing it further to 2 offers no additional accuracy gain. We therefore use $\lambda_{\text{soft}} = 1$ throughout our experiments. Table 7 reports the CE and KL limits used for each task group.