COMPRESSION OF VISION TRANSFORMER BY REDUCTION OF KERNEL COMPLEXITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-attention and transformer architectures have become foundational components in modern deep learning. In this work, we introduce Transformer with Kernel Complexity Reduction, or KCR-Transformer, a compact transformer block equipped with differentiable channel selection, guided by a novel and sharp theoretical generalization bound. To reduce the substantial computational cost of the MLP layers, the KCR-Transformer performs channel selection on the outputs of its self-attention layer. Furthermore, we provide a rigorous theoretical analysis establishing a tight generalization bound for networks equipped with KCR-Transformer blocks. Leveraging such strong theoretical results, the channel pruning by KCR-Transformer is conducted in a generalization-aware manner, ensuring that the resulting network retains a provably small generalization error. Our KCR-Transformer is compatible with many popular transformer networks, such as ViT and Swin, and it reduces the FLOPs of the vision transformers while maintaining or even improving the prediction accuracy. In the experiments, we replace all the transformer blocks in the vision transformers with KCR-Transformer blocks, leading to KCR-Transformer networks with different backbones. The resulting KCR-Transformers achieve superior performance on various computer vision tasks, even outperforming the original models with fewer FLOPs and parameters. The code of the KCR-Transformer is available at https://anonymous.4open.science/r/KCR-Transformer.

1 INTRODUCTION

Vision transformers achieve strong performance (Dosovitskiy et al., 2021; Liu et al., 2021; Zhu et al., 2021; Liu et al., 2023a; Singh et al., 2022) with substantial computational cost (Dosovitskiy et al., 2021; Touvron et al., 2021). To mitigate this overhead, numerous model compression techniques have been proposed, including knowledge distillation (Zhao et al., 2023; Yang et al., 2023), quantization (Lin et al., 2022; Li et al., 2023), neural architecture search (Gong et al., 2022; Su et al., 2022), and pruning (Chen et al., 2021b; Chavan et al., 2022; Zheng et al., 2022; Kong et al., 2022; Wang et al., 2022; Xu et al., 2024; Zhang et al., 2024). Among these methods, pruning has proven particularly effective due to the substantial parameter redundancy in ViT models. Despite substantial progress, conventional model compression methods are largely guided by empirical performance–efficiency trade-offs rather than principled theoretical foundations.

More recently, inspired by advances in the theoretical understanding of deep neural networks (DNNs) through the lens of kernel methods, such as the Neural Tangent Kernel (NTK) (Jacot et al., 2018), the kernel-based compression method has emerged as a principled alternative. The kernel-based methods aim to preserve the generalization capability of the compressed model by ensuring that the compressed model retains the training dynamics, convergence behavior, and inductive biases of the original network through spectral alignment of the NTK (Wei et al., 2023; Chen et al., 2021c; Mok et al., 2022; Wang et al., 2023; Rachwan et al., 2022). For instance, NTK-SAP (Wang et al., 2023) and Early-Lottery (Rachwan et al., 2022) leverage spectral preservation of the NTK during pruning to maintain the eigenspectrum, thereby preserving the generalization characteristics of the NTK of the original DNNs. Despite recent progress, a substantial gap persists between theory and practice in enhancing the generalization capability of compressed DNNs through NTK-based kernel learning. Existing theoretical frameworks, particularly those grounded in NTK analyses, are predominantly limited to over-parameterized DNNs (Cao & Gu, 2019; Arora et al., 2019; Ghor-

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.

bani et al., 2019), rendering them unsuited for modern architectures such as vision transformers with finite-width and diverse network architectures. Moreover, the linearized nature of the NTK regime inherently fails to model the dynamically evolving kernel characteristic of realistic training dynamics, thereby limiting its applicability to compressed models utilizing large-scale training for compelling performance on real-world tasks (Nichani et al., 2022; Damian et al., 2022; Takakura & Suzuki, 2024). To address this challenge, we first provide a theoretical analysis that establishes tight generalization upper and lower bounds in Theorem 3.1. Both an upper and lower bound of the expected loss, referred to as KCR upper and lower bounds, are established based on the training loss and the kernel complexity (KC) of the kernel gram matrix computed over the training data. The tightness of the KCR upper and lower bounds is studied in Section 4.6. *In contrast to the current NTK-based methods with their feature learning capability limited by the linear region of NTK, the KC measures the complexity of the dynamically evolving kernel formed by the DNN during the training process, accommodating the rich feature learning capability of DNNs*.

Since the training loss is usually optimized to a small value by training the DNN, the KCR upper and lower bounds in Theorem 3.1 can be tight and close to the expected loss if the KC is small. However, computing the KC involves the costly computation of the eigenvalues of a potentially large-scale gram matrix. To mitigate this issue, we introduce an approximate truncated nuclear norm (TNN) through the efficient Nyström method (Kumar et al., 2012), and the KC is reduced by the reduction of the approximate TNN. The approximate TNN is efficiently computed and added to the standard cross-entropy loss as a separable regularization term, termed KCR regularization, so that it can be optimized by the standard SGD-based optimization algorithms. Based on the reduction of the KC, we propose a novel vision transformer termed the Transformer with Kernel Complexity Reduction, or KCR-Transformer, whose training involves a search stage and a retrain stage. The channel selection in the input/output features of the MLP is performed in the search stage to obtain a compressed network architecture with reduced computation costs, and the compressed network is then retrained with the KCR regularization to reduce the KC, guaranteeing the generalization capability of the compressed model and leading to enhanced prediction accuracy. Extensive experiments verify that reducing the approximate TNN effectively reduces KC.

It is worthwhile to mention that the KCR regularization term is theoretically motivated by the fact that the reduction of the KC reduces the generalization error bound of DNNs, and the KCR regularization term is a critical part of the training loss in Equation (2) in Section 3.2 when performing both search and retraining stages of generalization-aware compression for all the experiments in this paper.

Contributions. The contributions of this paper are presented as follows.

First, we present a compact transformer block termed Transformer with Kernel Complexity Reduction, or KCR-Transformer. By selecting the channels in the input of the MLP layers via channel pruning, KCR-Transformer blocks effectively reduce the computational cost of the vision transformer. KCR-Transformer blocks can be used to replace all the transformer blocks in many popular vision transformers, such as ViT (Dosovitskiy et al., 2021) and Swin (Liu et al., 2021), rendering compact KCR-Transformer networks with comparable or even better performance. Experimental results show that KCR-Transformer not only reduces the parameter size and FLOPs but also outperforms the original models on tasks including image classification, object detection, semantic segmentation, and visual question answering.

Second, we provide a theoretical analysis showing tight generalization upper and lower bounds for the KCR-Transformer network. With such strong theoretical results in Theorem 3.1, the channel pruning by KCR-Transformer is performed in a *generalization-aware* manner. That is, the channel pruning of KCR provably keeps a small generalization error bound for the DNN with KCR-Transformer blocks, effectively guaranteeing the generalization capability of the DNN after channel pruning. This goal is achieved through the reduction of the KC of the DNN with KCR-Transformer blocks by the efficient and separable approximate TNN regularization. The reduction of the approximate TNN effectively reduces the KC, leading to the superior prediction accuracy of the compressed vision transformers by KCR-Transformer. Furthermore, unlike existing NTK-based compression methods constrained by NTK's linear regime, our KC captures the complexity of the evolving DNN kernel, enabling its rich feature learning capacity.

It is noted that the Gumbel–Softmax-based channel selection is a standard technique adopted from existing literature on differentiable neural architecture search (Wan et al., 2020; Chen et al., 2021a; Gong et al., 2022; Wei et al., 2024). The key novelty of our approach lies in the novel KC reduction algorithm with a theoretical guarantee for generalization-aware channel pruning. We note that the search and retraining of KCR-Transformer networks with KCR regularization are efficient, as evidenced in Section D.5 of the appendix. This paper is organized as follows. The related works in efficient vision transformers and compression of vision transformers are discussed in Section 2. The formulation of KCR-Transformer with our theoretical results is detailed in Section 3. The effectiveness of KCR-Transformer is demonstrated in Section 4 for image classification, dense prediction tasks, and multi-modal learning tasks. Throughout this paper, we use $a \lesssim b$ to denote $a \leq Cb$ if there exists such a positive constant C, and $a = \Theta(b)$ indicates that $a \lesssim b$ and $b \lesssim a$. $[n]$ denotes all the natural numbers between 1 and n inclusively.

2 Related Works

Efficient Vision Transformers and Compression of Vision Transformers. Efficient vision transformers have been developed through sparse attention mechanisms (Zhu et al., 2021; Yuan et al., 2021; Papa et al., 2024), the integration of convolutional operations (Cai et al., 2023; Mehta & Rastegari, 2022; Yuan et al., 2021; Bravo-Ortiz et al., 2024), Neural Architecture Search (Chen et al., 2021a; Gong et al., 2022; Wei et al., 2024), and Knowledge Distillation (Graham et al., 2021; Radosavovic et al., 2020; Gong et al., 2022; Yang et al., 2024). Pruning has also been widely explored to compress vision transformers (Chen et al., 2021b; Chavan et al., 2022; Zheng et al., 2022; Ahmed et al., 2025; Xu et al., 2024; Zhang et al., 2024).

Kernel Methods for DNNs. Kernel methods provide a principled framework for analyzing the training dynamics, generalization, and architectures of DNNs. A central line of work studies the neural tangent kernel (NTK) (Jacot et al., 2018) and its extensions to finite-width regimes (Seleznova & Kutyniok, 2022), deep narrow networks (Lee et al., 2022), and NTK evolution during training (Fort et al., 2020), while others discuss the limitations of purely kernel-based theories (Woodworth et al., 2020; Barrett & Dherin, 2021). Kernel-based perspectives further reveal hierarchical or implicit kernel structures in deep models (Montavon et al., 2011; Belkin et al., 2018; Xiao et al., 2020; Canatar & Pehlevan, 2022; Deng et al., 2022), and motivate RKHS and operator-theoretic foundations for deep learning (Hashimoto et al., 2023), hierarchical kernels for representation learning (Huang et al., 2023), kernel interpretations of self-attention (Song et al., 2021; Chen et al., 2023), and kernel-based transformer designs (Nguyen et al., 2022; 2023).

Kernel-Based Model Compression Methods. NTK-based methods offer a complementary perspective for model compression. NTK-Comp (Gu et al., 2022) studies pruning in wide multilayer perceptrons under Gaussian inputs and quantization that preserves the NTK spectrum, and MLP-Fusion (Wei et al., 2023) compresses transformers by clustering neurons to preserve both functional outputs and NTK similarity. NTK-based metrics are also used for training-free architecture search (Chen et al., 2021c) and early performance prediction in neural architecture search (Mok et al., 2022), though they can weaken in highly non-linear regimes, and preserving the NTK spectrum during pruning is shown to be important for stable training (Wang et al., 2023; Rachwan et al., 2022). However, these methods depend on static or "lazy" training regimes, which restricts their use for models with dynamically evolving representations.

3 Formulation

We introduce the KCR-Transformer, a compact transformer block designed to reduce the computational overhead of vision transformers through differentiable channel pruning in the MLP layers. To guide this pruning in a theoretically grounded manner, we present a novel generalization bound based on the kernel complexity (KC) of the network, and introduce the training algorithm of the network with KCR-Transformer for minimizing the upper bound.

3.1 Channel Selection for Attention Outputs

The vision transformer blocks usually apply a series of MLP layers to the output of the multi-head self-attention, which incurs substantial computation costs. To improve the efficiency of the vision transformer block, we propose pruning the channels in the attention outputs, thereby reducing the computation cost of the MLP layers, leading to the compact KCR-Transformer. To this end, we

maintain a decision mask $g \in \{0,1\}^D$, where $g_i = 1$ indicates that the i-th channel is selected, and 0 otherwise. Thus, the informative channels can be selected by multiplying g by each row of the attention output. To optimize the binary decision mask with gradient descent, we replace g with Gumbel Softmax weights in the continuous domain, which is computed by $\widehat{g}_i = \sigma\left(\frac{\alpha_i + \epsilon_i^{(1)} - \epsilon_i^{(2)}}{\tau}\right)$, where $\epsilon_i^{(1)}$ and $\epsilon_i^{(2)}$ are Gumbel noise. τ is the temperature. $\alpha \in \mathbb{R}^D$ is the sampling parameter. We define α as the architecture parameters that can be optimized by gradient descent during the differentiable search process. The decision mask g is trained via standard gradient descent by optimizing its corresponding sampling parameters α through the Gumbel Softmax relaxation, which enables differentiable channel selection. By gradually decreasing the temperature τ in the search process, α_i will be optimized such that g_i will approach 1 or 0. As detailed in Section C of the appendix, the decay of the temperature τ and the sampling of the Gumbel noise follow the common settings in differentiable pruning and NAS literature (Wan et al., 2020; Chang et al., 2019; Wu et al., 2019). Note that since the MLP input is exactly the output of the multi-head self-attention module in standard transformer blocks, and both share the same channel dimension due to the residual connection, we multiply the decision mask g with both the input and output features of the MLP layers. Such a design follows established practice in channel pruning literature of vision transformers (Chen et al., 2021b; Chavan et al., 2022; Zheng et al., 2022), where both input and output channels of the MLP layers are pruned jointly to maintain architectural consistency. After the search is finished, we apply the gather operation on the attention outputs from the selected channels. The dimension of the input and output features of the MLP layers is then changed to $\tilde{D} = \sum_{i=1}^{D} g_i$. The propagation of the selected dimensions across layers is detailed in Section B.1 of the appendix.

3.2 Generalization Bound by Approximate Kernel Complexity Loss and Its Optimization

Given the training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$ where $\mathbf{x}_i$ is the i-th input training feature, $\mathbf{y}_i \in \mathbb{R}^C$ is the corresponding one-hot vector as the class label of $\mathbf{x}_i$ and C is the number of classes. We denote the label matrix as $\mathbf{Y} \in \mathbb{R}^{n \times C}$ where the i-th row is $\mathbf{y}_i$ for all $i \in [n]$. Let $\mathbf{F} \in \mathbb{R}^{n \times d}$ be the features extracted on the entire training data set, where $\Phi(\cdot) \in \mathbb{R}^d$ denotes the mapping function of a DNN, such as ViT (Dosovitskiy et al., 2021), and d is the output dimension of the DNN before the final softmax layer for classification. We remark that almost all the DNNs use a linear layer to generate the output of the network for discriminative learning tasks, so that the mapping function of a DNN can be formulated as $\Phi(\cdot) = \Phi(\mathcal{W}, \cdot) = F(\mathcal{W}_2, \cdot)\mathcal{W}_1$, where $\mathcal{W}_1 \in \mathbb{R}^{d \times C}$ contains the weights in the final linear layer of the DNN where d is the hidden dimension, $F(\mathcal{W}_2, \cdot) \in \mathbb{R}^d$ represents the feature extraction backbone of the network before the final linear layer, and $\mathcal{W}_2$ are the weights of the backbone F, with $\mathcal{W} = \{\mathcal{W}_1, \mathcal{W}_2\}$. The DNN is denoted as $\mathrm{NN}_{\mathcal{W}}(\cdot)$. It is noted that such a formulation does not impose any limitation on the feature backbone F so as to admit a broad class of DNNs with various architectures for real-world vision discriminative tasks. We define a positive definite kernel for the DNN as $K(\mathbf{x}, \mathbf{x}') = F(\mathbf{x})^\top F(\mathbf{x}')$ (in the standard kernel-learning sense, i.e., its gram matrix is positive semidefinite; see Section A.2), where $\mathcal{X}$ is the input domain of the DNN and $F(\mathcal{W}, \cdot)$ is abbreviated into $F(\cdot)$. Compared to existing NTK-based methods (Wang et al., 2023; Wei et al., 2023; Gu et al., 2022) where a static NTK is used as the kernel, our kernel K is dynamic with the learned feature backbone F during the training process of the DNN. Let $\mathbf{F}_i = F(\mathcal{W}_2, \mathbf{x}_i)$ be the learned representation for the i-th training data, and $\mathbf{F} \in \mathbb{R}^{n \times d}$ is the matrix of all the learned representations over the training data. Then the gram matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ of the kernel K over the training data is calculated by $\mathbf{K} = \mathbf{F}\mathbf{F}^\top \in \mathbb{R}^{n \times n}$, and the eigenvalues of $\mathbf{K}_n \coloneqq \mathbf{K}/n$ are $\widehat{\lambda}_1 \geq \widehat{\lambda}_2 \ldots \geq \widehat{\lambda}_{r_0} \geq \widehat{\lambda}_{r_0+1} = \ldots = \widehat{\lambda}_n = 0$ with $r_0 = \min\{n, d\}$, since $\mathrm{rank}(\mathbf{K}) \leq r_0$.

Suppose the input feature $\mathbf{x}$ and its class label vector $\mathbf{y}$ follow an unknown joint distribution P, then the expected risk of the DNN is defined as $L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\in P}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right]$, which also represents the generalization error of the DNN. The following theorem, Theorem 3.1, based on the local complexity of the function class of the DNN feature extraction backbones and rooted in the well-established local Rademacher complexity literature (Bartlett et al., 2005; Koltchinskii, 2006; Mendelson, 2002), gives sharp upper and lower bounds for the generalization error of the DNN. Theorem 3.1 uses the kernel complexity (KC) of the dynamic kernel K over the training data as $\mathrm{KC}(\mathbf{K}) \coloneqq \min_{h \in [0, r_0]} \left(\frac{h}{n} + \sqrt{\frac{1}{n}\sum_{i=h+1}^{r_0} \widehat{\lambda}_i}\right)$.

Theorem 3.1. Let K be the dynamic kernel after a particular optimization epoch by GD or SGD. Then for every $\delta > 0$, with probability at least $1 - \exp(-\delta)$, we have

$$\underbrace{\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] - \mathrm{KC}(\mathbf{K}) - \frac{\delta}{n}}_{\text{KCR Lower Bound}} \lesssim L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) \lesssim \underbrace{\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] + \mathrm{KC}(\mathbf{K}) + \frac{\delta}{n}}_{\text{KCR Upper Bound}}, \tag{1}$$

where $\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] = 1/n \cdot \sum_{i=1}^n \|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}_i) - \mathbf{y}_i\|_2^2$ is the empirical loss on the training data.

The proof of this theorem is deferred to Section A of the appendix. Section A.1 extends Theorem 3.1 to the cross-entropy loss with only constant-factor changes, and Section A.2 shows that the KCR bounds hold both before and after channel selection. The empirical loss $\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right]$ is usually optimized to a small value by training the network $\mathrm{NN}_{\mathcal{W}}$, so it can be observed from (1) that the KCR upper and lower bounds for $L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}})$ can be tight and close to $L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}})$ if $\mathrm{KC}(\mathbf{K})$ is small, guaranteeing the generalization capability of the DNN. To reduce the KC, we introduce the truncated nuclear norm (TNN) of $\mathbf{K}$ denoted as $\|\mathbf{K}\|_r \coloneqq \sum_{i=r+1}^{r_0} \widehat{\lambda}_i$ where $r \in [0\colon r_0]$, with $r_0 = \gamma_0 \min\{n, d\}$. γ_0 is to be chosen by cross-validation. It can be verified that a smaller TNN $\|\mathbf{K}\|_r$ leads to a smaller kernel complexity $\mathrm{KC}(\mathbf{K})$. Since the computation of the TNN involves the computation of the eigenvalues of the potentially large-scale gram matrix $\mathbf{K}$ over large-scale training data, we then describe below how to efficiently and effectively approximate the TNN. We first compute the approximate top-r_0 eigenvectors of $\mathbf{K}_n$, $\tilde{\mathbf{U}}^{(r_0)}$, by the Nyström method (Kumar et al., 2012). Here $\mathbf{A}^{(r)}$ denotes a submatrix of $\mathbf{A}$ formed by its top r columns.

Efficient Computation of the Top-r_0 Eigenvectors of $\mathbf{K}$. To approximate the top-r_0 eigenvectors, $\mathbf{U}^{(r_0)} \in \mathbb{R}^{n \times r_0}$, of the gram matrix $\mathbf{K}$ using the Nyström method (Kumar et al., 2012), we first sample m landmark points from the training set, indexed by $\mathcal{I} \subset [n]$ with $|\mathcal{I}| = m \ll n$. Let $\mathbf{F}_{\mathcal{I}} \in \mathbb{R}^{m \times d}$ be the features corresponding to the landmark set. We define $\mathbf{C} = \mathbf{F}\mathbf{F}_{\mathcal{I}}^\top \in \mathbb{R}^{n \times m}$ as the cross-covariance matrix and $\mathcal{W} = \mathbf{F}_{\mathcal{I}}\mathbf{F}_{\mathcal{I}}^\top \in \mathbb{R}^{m \times m}$ as the gram matrix on the landmarks. Next, we compute the top-r_0 eigen-decomposition of $\mathcal{W}$ as $\mathcal{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$, where $\mathbf{Q} \in \mathbb{R}^{m \times r_0}$ contains the top-r_0 eigenvectors and $\mathbf{\Lambda} \in \mathbb{R}^{r_0 \times r_0}$ is the diagonal matrix of corresponding eigenvalues. The Nyström approximation of $\mathbf{K}$ is then given by $\tilde{\mathbf{K}} = \mathbf{C}\mathcal{W}^\dagger\mathbf{C}^\top$, and the approximate top-r_0 eigenvectors are computed as $\tilde{\mathbf{U}}^{(r_0)} = \mathbf{C}\mathbf{Q}\mathbf{\Lambda}^{-1/2} \in \mathbb{R}^{n \times r_0}$, which serves as an efficient approximation to $\mathbf{U}^{(r_0)}$ with significantly reduced computational cost.

We let $\mathbf{U}_r = \tilde{\mathbf{U}}^{(r)}$, then the sum of the top-r eigenvalues of $\mathbf{K}_n$ is approximated by $\mathrm{tr}(\mathbf{U}_r{}^\top \mathbf{K}_n \mathbf{U}_r)$. Since $\mathrm{tr}(\mathbf{K}_n) = \sum_{i=1}^n K(\mathbf{x}_i, \mathbf{x}_i)/n = \sum_{i=1}^n \widehat{\lambda}_i$, and $\mathrm{tr}(\mathbf{U}_r{}^\top \mathbf{K}_n \mathbf{U}_r) = \sum_{i=1}^n (\sum_{s=1}^r \sum_{k=1}^n [\mathbf{U}_r^\top]_{si} [\mathbf{K}_n]_{ik} [\mathbf{U}_r]_{ks})$, we can approximate the $\|\mathbf{K}\|_r$ by $\overline{\|\mathbf{K}\|}_r = \mathrm{tr}(\mathbf{K}_n) - \mathrm{tr}(\mathbf{U}_r{}^\top \mathbf{K}_n \mathbf{U}_r)$ which is separable. In particular, $\overline{\|\mathbf{K}\|}_r = \sum_{i=1}^n K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^n (\sum_{s=1}^r \sum_{k=1}^n [\mathbf{U}_r^\top]_{si} [\mathbf{K}_n]_{ik} [\mathbf{U}_r]_{ks})$. We remark that $\overline{\|\mathbf{K}\|}_r$ is ready to be optimized by standard SGD algorithms because it is separable and expressed as the summation of losses on individual training data points. The training loss of a neural network with KCR-Transformer blocks has $\mathrm{KCR}(\mathcal{W}) = \overline{\|\mathbf{K}\|}_r$, the approximate TNN, as a regularization term. The following functions are needed for minibatch-based training with SGD, with the subscript j indicating the corresponding loss on the j-th batch $\mathcal{B}_j$:

$$\mathcal{L}_{\text{train},j}^{(t)}(\mathcal{W}) = \mathrm{CE}_j^{(t)}(\mathcal{W}) + \eta \mathrm{KCR}_j(\mathcal{W}), \tag{2}$$

where

$$\mathrm{KCR}_j(\mathcal{W}) = \frac{1}{|\mathcal{B}_j|} \sum_{i=1}^{|\mathcal{B}_j|} \left(K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{s=1}^r \sum_{k=1}^n \left[\mathbf{U}_r^\top\right]_{si} \left[\mathbf{K}_n\right]_{ik} \left[\mathbf{U}_r\right]_{ks} \right),$$

$$\mathrm{CE}_j^{(t)}(\mathcal{W}) = \frac{1}{|\mathcal{B}_j|} \sum_{i=1}^{|\mathcal{B}_j|} H(X_i(\mathcal{W}), Y_i).$$

Here $\mathrm{CE}_j^{(t)}$ is the cross-entropy loss on batch $\mathcal{B}_j$ at epoch t. $X_i(\mathcal{W})$ denotes the logits of the network for $\mathbf{x}_i$, and $H(\cdot, \cdot)$ is the cross-entropy function. η is the balance factor.

3.3 Search and Retraining Processes

To obtain a compact vision transformer network with KCR-Transformer, or the KCR-Transformer network, we optimize both the accuracy of the network and the inference cost (FLOPs) of the network. We follow the standard techniques in neural architecture search (Tan et al., 2019), where the attention output channels are pruned in the search process, and the pruned network is retrained in the retraining process. The search process involves channel selection by Gumbel-Softmax and entropy minimization for architecture search. We optimize the FLOPs of the operations whose computation cost is decided by the channel selection for attention outputs described in Section 3.1. We estimate the FLOPs of the MLP after the channel selection on the attention outputs following the KCR-Transformer. The FLOPs related to a single Transformer block is $\texttt{cost}_j = l_j \cdot \left(2\tilde{D}^2 + \tilde{D}\right)$, where j indexes the KCR-Transformer block. $2\tilde{D}^2 + \tilde{D}$ is the FLOPs of a layer of the MLP after the channel selection on the attention outputs, and l_j denotes the number of layers in the MLP of the j-th KCR-Transformer block. The inference cost objective of the network architecture is computed by $\texttt{cost} = \sum_{j=1}^{M} \texttt{cost}_j$, where M is the number of transformer blocks. The overall loss function for search on each batch $\mathcal{B}_j$ at epoch t is formulated by $\mathcal{L}^{(t)}_{\text{search},j}(\mathcal{W}, \alpha) = \mathcal{L}^{(t)}_{\text{train},j}(\mathcal{W}, \alpha) + \lambda \cdot \log \texttt{cost}(\alpha)$, where α is the architecture parameter. λ controls the magnitude of the cost term. In the search phase, the search loss is optimized to perform the two types of channel selection for all the KCR-Transformer blocks. After the search process, we use the selected channels for the attention outputs in a searched network and then perform retraining on the searched network using the training loss (2). Algorithm 1 in Section B of the appendix describes the search and retraining processes.

4 Experimental Results

We present the experimental implementation details in Section 4.1. We evaluate KCR-Transformers for ImageNet-1K classification in Section 4.2. In Section 4.3, we study KCR-Transformer as the feature-extraction backbone for semantic segmentation and object detection. Section 4.4 examines its effectiveness in reducing KC. Section 4.5 evaluates KCR-Transformer as a vision encoder for vision-language models on visual question answering. Section 4.6 studies the expected loss and the approximated KCR upper/lower bounds. Additional experimental results are provided in Section D of the appendix. Sections D.1 and D.2 evaluate transferability and self-supervised pre-training, respectively. Section D.3 describes the implementation details for semantic segmentation, while Section D.4 reports detailed object detection results. Section D.5 evaluates the efficiency of the search and retraining processes. Section D.6 studies robustness on ImageNet-A, ImageNet-C, and ImageNet-R. Section D.7 analyzes sensitivity to γ_0, η, and m. Section D.8 compares KCR-Transformers with competing compression methods under different compression levels, and Section D.9 discusses the effect of the channel-selection ratio and the effectiveness on compact backbones. Section D.10 compares inference time and throughput, and Section D.11 analyzes system-level efficiency, memory footprint, and the KV cache. Section D.12 evaluates complex multimodal reasoning on ScienceQA, and Section D.13 applies KCR to jointly compress a complete vision-language model, LLaVA-1.5-7B, including its decoder FFNs and attention/KV dimensions. Section D.14 presents Grad-CAM visualizations. To assess the influence of KC, Section D.15 compares KCR-Transformers with identical FLOPs and parameter sizes but different KC values, and Section D.16 presents the expected loss and the approximated KCR upper/lower bounds for Swin-T and ViT-S. Section D.17 analyzes the effectiveness and efficiency of KCR regularization with Nyström approximation, and Section D.18 compares KCR with random and magnitude-based channel selection at matched sparsity. Section D.19 visualizes learned channel-selection masks, and Section D.20 performs t-SNE analysis of learned image representations. Additional KC comparisons with baselines are provided in Section D.21. Finally, Section D.22 conducts statistical significance analysis of KCR-Transformer improvements on ImageNet-1K.

4.1 Experiment Settings

Datasets. ImageNet-1K (Russakovsky et al., 2015) is used as the primary image classification benchmark. We evaluate object detection and semantic segmentation on MS-COCO (Lin et al., 2014) and semantic segmentation on ADE20K (Zhou et al., 2019). For vision-language evaluation, we use VQAv2 (Goyal et al., 2017) and SNLI-VE (Xie et al., 2019) to assess KCR-Transformer

as the vision encoder. To examine robustness under distribution shift, we further evaluate on ImageNet-A (Hendrycks et al., 2021), ImageNet-C (Hendrycks & Dietterich, 2019), and ImageNet-R (Hendrycks & Dietterich, 2019). We study transfer learning on Oxford Flowers-102 (Nilsback & Zisserman, 2008), Oxford-IIIT Pet (Parkhi et al., 2012), and Stanford Cars (Krause et al., 2013).

Implementation Details. During architecture search, we randomly sample 100 ImageNet classes (Russakovsky et al., 2015) and train the network for 200 epochs, and the searched network is then retrained on ImageNet-1K for 300 epochs. The values of γ_0 and η are selected by 5-fold cross-validation, we set $m = 50000$ throughout all experiments, and the hyperparameter λ, which controls the searched architecture size, is set to 0.2 in the main experiments. The complete settings for search and retraining are provided in Section C of the appendix, and the sensitivity to γ_0, η, m, and λ is studied in Sections D.7 and D.8.

4.2 Image Classification

We adopt EfficientViT-B1 (Cai et al., 2023), ViT-S (Dosovitskiy et al., 2021), ViT-B (Dosovitskiy et al., 2021), Swin-T (Liu et al., 2021), and Swin-B (Liu et al., 2021) as backbone models. Each transformer block in these architectures is substituted with a KCR-Transformer block. We compare the performance of KCR-Transformers against vision transformers compressed by the state-of-the-art kernel-based model compression methods, NTK-SAP (Wang et al., 2023) and MLP-Fusion (Wei et al., 2023), and the state-of-the-art pruning methods, LPViT (Xu et al., 2024), STAR (Zhang et al., 2024), and DeepCompress (Ahmed et al., 2025). Table 1 shows that KCR-Transformers consistently exhibit reduced computational cost and improved performance compared to the baseline vision transformers. For instance, KCR-Swin-B achieves a 1.2% improvement in top-1 accuracy over Swin-B while reducing the computational cost by 2.8 GFLOPs, highlighting the effectiveness of KCR in enhancing both efficiency and predictive performance. Notably, KCR-EfficientViT-B1 achieves the lowest computational cost among all evaluated models, requiring only 0.44 GFLOPs while even outperforming its uncompressed counterpart by 1.0% and all baselines in the first part of Table 1 in top-1 accuracy, demonstrating the effectiveness of KCR-Transformers for resource-constrained applications. Moreover, Table 10 in the appendix shows that the performance and efficiency improvements by the KCR-Transformer are achieved with only marginal increases ($< 5.4\%$) in the training time. Table 16 in Section D.10 shows that KCR-Transformers exhibit faster inference speed and higher throughput than models compressed by competing baselines. We further compare KCR-Transformers with models compressed by competing compression methods at different compression levels in Section D.8 of the appendix. Although the performance of the KCR-Transformer

Table 1: Top-1 (%) classification accuracy comparison on ImageNet-1k.

Model	Params	FLOPs	Top-1
T2T (Yuan et al., 2021)	6.9 M	1.8 G	76.5
Mobile-Former (Chen et al., 2022)	9.4 M	0.2 G	76.7
EViT (Liu et al., 2023b)	12.4 M	0.5 G	77.1
TinyViT (Wu et al., 2022)	5.4 M	1.3 G	79.1
EfficientFormer (Li et al., 2022a)	12.3 M	1.3 G	79.2
VTC-LFC (Wang et al., 2022)	5.0 M	1.3 G	78.0
SPViT (Kong et al., 2022)	4.9 M	1.2 G	77.8
EfficientViT-B1 (Cai et al., 2023)	9.1 M	0.52 G	79.4
MLP-Fusion-EfficientViT-B1 (Wei et al., 2023)	7.9 M	0.48 G	79.1
NTK-SAP-EfficientViT-B1 (Wang et al., 2023)	8.0 M	0.49 G	79.4
LPViT-EfficientViT-B1 (Xu et al., 2024)	7.9 M	0.46 G	79.1
STAR-EfficientViT-B1 (Zhang et al., 2024)	8.0 M	0.48 G	79.3
DeepCompress EfficientViT-B1 (Ahmed et al., 2025)	7.9 M	0.46 G	79.2
KCR-EfficientViT-B1 (Ours)	7.8 M	0.44 G	**80.4**
ViT-S (Dosovitskiy et al., 2021)	22.1 M	4.3 G	81.2
MLP-Fusion-ViT-S (Wei et al., 2023)	19.8 M	4.0 G	81.0
NTK-SAP-ViT-S (Wang et al., 2023)	20.3 M	3.9 G	80.9
LPViT-ViT-S (Xu et al., 2024)	19.8 M	3.8 G	81.0
STAR-ViT-S (Zhang et al., 2024)	20.2 M	3.9 G	81.2
DeepCompress-ViT-S (Ahmed et al., 2025)	20.0 M	3.9 G	81.1
KCR-ViT-S (Ours)	19.8 M	3.8 G	**82.2**
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	83.7
MLP-Fusion-ViT-B (Wei et al., 2023)	70.2 M	15.3 G	83.5
NTK-SAP-ViT-B (Wang et al., 2023)	71.8 M	15.6 G	83.5
LPViT-ViT-B (Xu et al., 2024)	69.8 M	15.1 G	83.4
STAR-ViT-B (Zhang et al., 2024)	69.5 M	14.7 G	83.5
DeepCompress-ViT-B (Ahmed et al., 2025)	70.5 M	15.1 G	83.6
KCR-ViT-B (Ours)	69.5 M	14.5 G	**84.6**
Swin-T (Liu et al., 2021)	29.0 M	4.5 G	81.3
MLP-Fusion-Swin-T (Wei et al., 2023)	24.8 M	4.1 G	81.0
NTK-SAP-Swin-T (Wang et al., 2023)	25.5 M	4.2 G	81.2
LPViT-Swin-T (Xu et al., 2024)	24.8 M	4.0 G	81.1
STAR-Swin-T (Zhang et al., 2024)	24.6 M	4.0 G	81.4
DeepCompress-Swin-T (Ahmed et al., 2025)	24.8 M	4.1 G	81.1
KCR-Swin-T (Ours)	24.6 M	3.9 G	**82.4**
Swin-B (Liu et al., 2021)	88.0 M	15.4 G	83.5
MLP-Fusion-Swin-B (Wei et al., 2023)	70.8 M	13.3 G	83.2
NTK-SAP-Swin-B (Wang et al., 2023)	72.6 M	13.2 G	83.2
LPViT-Swin-B (Xu et al., 2024)	70.8 M	13.5 G	83.3
STAR-Swin-B (Zhang et al., 2024)	70.2 M	13.4 G	83.6
DeepCompress-Swin-B (Ahmed et al., 2025)	71.5 M	13.0 G	83.1
KCR-Swin-B (Ours)	70.2 M	12.6 G	**84.7**

degrades and falls below that of the original uncompressed model under aggressive compression, KCR-Transformer still significantly outperforms models compressed by competing compression methods at similar compression levels, as evidenced in Table 15 in Section D.8.

4.3 Object Detection and Semantic Segmentation

We incorporate the ImageNet pre-trained KCR-Swin-B into the Cascade Mask R-CNN framework (Cai & Vasconcelos, 2021) for object detection. All models are evaluated on the MS-COCO dataset (Lin et al., 2014). The implementation details for the experiments are presented in Section D.4 of the appendix. It is observed in Table 2 that compressing the Swin backbones with KCR consistently improves both box-level and mask-level detection performance within the Cascade Mask R-CNN framework. For example, KCR-Swin-B achieves a box mAP of 52.5% and a mask mAP of 45.6%, with improvements of 0.6% and 0.6% over the standard Swin-B baseline. Results with AP at IoU thresholds of 50 and 75 are deferred to Table 9 in Section D.4 of the appendix.

Table 2: Object Detection on COCO.

Framework	Feature Backbone	mAP^{box} (%)	mAP^{m} (%)
Mask R-CNN	Swin-B	51.9	45.0
Mask R-CNN	NTK-SAP-Swin-B	51.5	44.6
Mask R-CNN	KCR-Swin-B (Ours)	**52.5**	**45.6**

Table 3: Segmentation on ADE20K.

Framework	Feature Backbone	Val mIoU (%)
SETR	ViT-L	50.3
UperNet	Swin-B	51.6
UperNet	NTK-SAP-Swin-B	51.3
UperNet	KCR-Swin-B (Ours)	**52.4**

In addition, we evaluate the performance of KCR for segmentation on the ADE20K (Zhou et al., 2019) using UperNet (Xiao et al., 2018) with our KCR-Swin-B as the feature extraction backbone. We include Swin-B (Liu et al., 2021) and SETR (Zheng et al., 2021) as baselines for comparisons. We follow the training and evaluation protocol in (Liu et al., 2021). More implementation details for the experiments are presented in Section D.3. As shown in Table 3, UperNet equipped with our KCR-Swin-B backbone achieves the highest validation mIoU of 52.4%, surpassing UperNet with Swin-B by 0.8% and SETR with ViT-L by 2.1%.

4.4 Study on the Effects of KCR-Transformer in Reducing the KC

We study the effectiveness of KCR-Transformer in reducing the KC on ViT-S and ViT-B in Table 4, with additional results on EfficientViT-B1, Swin-T, and Swin-B provided in Table 24 in Section D.21. As shown in Table 4, the KCR-enhanced models consistently exhibit a substantial reduction in KC while simultaneously achieving lower parameter counts, reduced FLOPs, and higher top-1 classification accuracy on ImageNet-1K. For instance, KCR-ViT-S reduces the KC from 0.45 to 0.18 and improves the top-1 accuracy from 81.2% to 82.2% with fewer FLOPs and parameters. These improvements demonstrate that our approximate TNN regularization effectively reduces the KC of the vision backbones, leading to improved generalization capability. In contrast, the existing kernel-based compression methods, NTK-SAP (Wang et al., 2023) and MLP-Fusion (Wei et al., 2023), are limited by the conventional NTK, so the top-1 accuracy of the models pruned by these methods is significantly worse than KCR-Transformers.

Table 4: Comparison of the KC between KCR-Transformers and models compressed by the baselines using ViT-S and ViT-B.

Model	# Params	FLOPs	Top-1 (%)	KC
ViT-S (Dosovitskiy et al., 2021)	22.1 M	4.3 G	81.2	0.45
MLP-Fusion-ViT-S (Wei et al., 2023)	19.8 M	4.0 G	81.0	0.43
NTK-SAP-ViT-S (Wang et al., 2023)	20.3 M	3.9 G	80.9	0.38
LPViT-ViT-S (Xu et al., 2024)	19.8 M	3.8 G	81.0	0.42
STAR-ViT-S (Zhang et al., 2024)	20.2 M	3.9 G	81.2	0.43
DeepCompress-ViT-S (Ahmed et al., 2025)	20.0 M	3.9 G	81.1	0.44
KCR-ViT-S (Ours)	19.8 M	3.8 G	**82.2**	**0.18**
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	83.7	0.42
MLP-Fusion-ViT-B (Wei et al., 2023)	70.2 M	15.3 G	83.5	0.37
NTK-SAP-ViT-B (Wang et al., 2023)	71.8 M	15.6 G	83.5	0.37
LPViT-ViT-B (Xu et al., 2024)	69.8 M	15.1 G	83.4	0.38
STAR-ViT-B (Zhang et al., 2024)	69.5 M	14.7 G	83.5	0.38
DeepCompress-ViT-B (Ahmed et al., 2025)	70.5 M	15.1 G	83.6	0.39
KCR-ViT-B (Ours)	69.5 M	14.5 G	**84.6**	**0.16**

4.5 Visual Question Answering with KCR-Transformer as the Vision Encoder

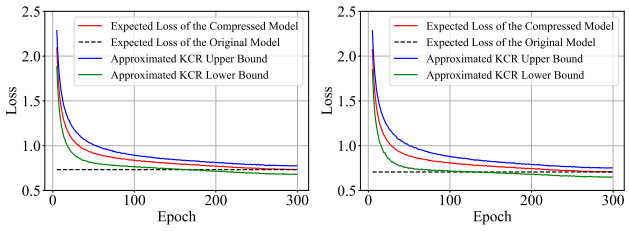
We further evaluate the KCR-Transformer in the visual question answering (VQA) task by employing it as the vision encoder within the vision-language model (VLM) framework, FLAVA (Singh et al., 2022). The evaluation is performed on widely used VQA datasets VQAv2 (Goyal et al., 2017) and SNLI-VE (Xie et al., 2019). We replace the vision encoder, ViT-B, in the VLM framework, FLAVA, with our KCR-Transformer during the vision encoder pre-training process. For VQAv2, we follow the settings in FLAVA (Singh et al., 2022) and fine-tune the model using a cross-entropy loss over the top frequent answer candidates. For SNLI-VE, we also follow the settings in FLAVA (Singh et al., 2022) and fine-tune the model using a standard three-way classification loss to distinguish entailment, contradiction, and neutrality. The test-dev VQA score for VQAv2 and the test accuracy for SNLI-VE are reported following (Singh et al., 2022). It is observed in Table 5 that FLAVA with KCR-ViT-B consistently outperforms the original FLAVA with ViT-B and the baseline model with ViT-B compressed by NTK-SAP across both benchmarks while maintaining a lower computational cost. For example, FLAVA with KCR-ViT-B achieves a VQAv2 score of 73.26%, outperforming FLAVA with ViT-B by 0.77% and FLAVA with NTK-SAP-ViT-B by 1.18%.

Table 5: Performance for VQA on VQAv2 and SNLI-VE. ViT-B, NTK-SAP-ViT-B, and KCR-ViT-B are employed as the vision encoder in the VLM framework, FLAVA (Singh et al., 2022).

Model	# Params	FLOPs	VQAv2 (%)	SNLI-VE (%)
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	72.49	78.89
NTK-SAP-ViT-B (Wang et al., 2023)	71.8 M	15.6 G	72.08	78.16
KCR-ViT-B (Ours)	69.5 M	14.5 G	**73.26**	**79.35**

4.6 Study on the KCR Upper and Lower Bounds

Figure 1 illustrates the expected loss computed over the training and validation sets of the ImageNet-1K and the approximated KCR upper/lower bounds computed at different training epochs for Swin-B and ViT-B trained on the ImageNet-1K dataset, and the results for Swin-T and ViT-S are provided in Figure 3 in Section D.16 of the appendix. We note that the approximated KCR upper/lower bounds are the KCR upper/lower bounds with the KC replaced by the approximate KC, A-KC, which is defined as $\text{A-KC}(\mathbf{K}) := \min_{h\in[0,r_0]} \left(\frac{h}{n} + \sqrt{\frac{1}{n}\overline{\|\mathbf{K}\|}_h}\right)$. Since the approximate TNN $\overline{\|\mathbf{K}\|}_h$ is expected to be close to the TNN $\|\mathbf{K}\|_h$ for each $h \in [0\colon r_0]$, the approximate KC $\text{A-KC}(\mathbf{K})$ is also expected to be close to the KC, $\text{KC}(\mathbf{K})$. It is observed that the approximated KCR upper/lower bounds are tightly correlated to the expected loss, revealing the tightness of the upper/lower bounds for the generalization error of the DNNs with KCR-Transformers.



(a) Swin-B (b) ViT-B

Figure 1: Illustration of the expected loss and the approximated KCR upper/lower bounds over different training epochs for Swin-B and ViT-B compressed by KCR.

5 Conclusion

In this paper, we propose KCR-Transformer, a novel and generalization-aware transformer block equipped with differentiable channel selection for the MLP layers in vision transformers. Guided by a novel and sharp theoretical generalization bound derived from the kernel complexity (KC) of the network, KCR-Transformer enables channel pruning in a theoretically grounded and principled manner. Our method is compatible with a wide range of vision transformer architectures and can be seamlessly integrated to replace standard transformer blocks. Extensive experiments across image classification, object detection, semantic segmentation, and visual question answering demonstrate that KCR-Transformer consistently achieves superior performance with fewer FLOPs and parameters, validating its effectiveness for building efficient vision transformers.

AI USE STATEMENT

In this work, we used generative AI tools for interpreting results. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, and generating synthetic data sets, implementing methods, cleaning and reformatting dataset, assisting with translation, supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for improving readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The code of the KCR-Transformer is available at the anonymous link provided in the abstract. The proof of Theorem 3.1 is provided in Section A, together with its extension to the cross-entropy loss in Section A.1 and its validity before and after channel selection in Section A.2. The search and retraining processes are described in Algorithm 1 in Section B, and the hyperparameters and implementation details are provided in Section 4.1 and Section C. The implementation details for semantic segmentation, object detection, and the joint compression of LLaVA-1.5-7B are provided in Sections D.3, D.4, and D.13, respectively.

REFERENCES

Sabbir Ahmed, Abdullah Al Arafat, Deniz Najafi, Akhlak Mahmood, Mamshad Nayeem Rizve, Mohaiminul Al Nahian, Ranyang Zhou, Shaahin Angizi, and Adnan Siraj Rakin. Deepcompress-vit: Rethinking model compression to enhance efficiency of vision transformers at the edge. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 30147–30156. Computer Vision Foundation / IEEE, 2025.

Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, volume 97 of *Machine Learning Research*, pp. 322–332. PMLR, 2019.

David G. T. Barrett and Benoit Dherin. Implicit gradient regularization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *Ann. Statist.*, 33(4):1497–1537, 08 2005.

Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 540–548. PMLR, 2018.

Mario Alejandro Bravo-Ortiz, Esteban Mercado Ruiz, Juan Pablo Villa Pulgarin, Carlos Angel Hormaza-Cardona, Sebastián Quiñones Arredondo, Harold Brayan Arteaga-Arteaga, Simon Orozco-Arias, Oscar Cardona-Morales, and Reinel Tabares-Soto. Cvtstego-net: A convolutional vision transformer architecture for spatial image steganalysis. *J. Inf. Secur. Appl.*, 81:103695, 2024.

Han Cai, Junyan Li, Muyan Hu, Chuang Gan, and Song Han. Efficientvit: Lightweight multi-scale attention for high-resolution dense prediction. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pp. 17256–17267. IEEE, 2023.

Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: high quality object detection and instance segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(5):1483–1498, 2021.

Abdulkadir Canatar and Cengiz Pehlevan. A kernel analysis of feature learning in deep neural networks. In *58th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2022, Monticello, IL, USA, September 27-30, 2022*, pp. 1–8. IEEE, 2022.

Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 10835–10845, 2019.

Jianlong Chang, Xinbang Zhang, Yiwen Guo, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. DATA: differentiable architecture approximation. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 874–884, 2019.

Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric P. Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 4921–4931. IEEE, 2022.

Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 12250–12260. IEEE, 2021a.

Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 19974–19988, 2021b.

Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four GPU hours: A theoretically inspired perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021c.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9620–9629. IEEE, 2021d.

Yingyi Chen, Qinghua Tao, Francesco Tonin, and Johan A. K. Suykens. Primal-attention: Self-attention through asymmetric kernel SVD in primal representation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. Mobile-former: Bridging mobilenet and transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 5260–5269. IEEE, 2022.

Alexandru Damian, Jason D. Lee, and Mahdi Soltanolkotabi. Neural networks can learn representations with gradient descent. In Po-Ling Loh and Maxim Raginsky (eds.), *Conference on Learning Theory, 2-5 July 2022, London, UK*, volume 178 of *Proceedings of Machine Learning Research*, pp. 5413–5452. PMLR, 2022.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations, ICLR 2024*. OpenReview.net, 2024.

Zhijie Deng, Jiaxin Shi, and Jun Zhu. Neuralef: Deconstructing kernels by deep neural networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4976–4992. PMLR, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M. Roy, and Surya Ganguli. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Linearized two-layers neural networks in high dimension. *CoRR*, abs/1904.12191, 2019.

Spyros Gidaris, Andrei Bursuc, Oriane Siméoni, Antonín Vobecký, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. MOCA: self-supervised representation learning by predicting masked online codebook assignments. *Trans. Mach. Learn. Res.*, 2024, 2024.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory, COLT 2018*, volume 75 of *Proceedings of Machine Learning Research*. PMLR, 2018.

Chengyue Gong, Dilin Wang, Meng Li, Xinlei Chen, Zhicheng Yan, Yuandong Tian, Qiang Liu, and Vikas Chandra. Nasvit: Neural architecture search for efficient vision transformers with gradient conflict aware supernet training. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 6325–6334. IEEE Computer Society, 2017.

Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 12239–12249. IEEE, 2021.

Lingyu Gu, Yongqi Du, Yuan Zhang, Di Xie, Shiliang Pu, Robert C. Qiu, and Zhenyu Liao. "lossless" compression of deep neural networks: A high-dimensional neural tangent kernel approach. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Yuka Hashimoto, Masahiro Ikeda, and Hachem Kadri. Deep learning with kernels through RKHM and the perron-frobenius operator. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Elad Hazan, Tomer Koren, and Kfir Y. Levy. Logistic regression: Tight bounds for stochastic and online optimization. In *Proceedings of The 27th Conference on Learning Theory, COLT 2014*, volume 35 of *JMLR Workshop and Conference Proceedings*. JMLR.org, 2014.

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 15262–15271. Computer Vision Foundation / IEEE, 2021.

Wentao Huang, Houbao Lu, and Haizhang Zhang. Hierarchical kernels in deep kernel learning. *J. Mach. Learn. Res.*, 24:391:1–391:30, 2023.

Yongqi Huang, Peng Ye, Chenyu Huang, Jianjian Cao, Lin Zhang, Baopu Li, Gang Yu, and Tao Chen. DeRS: Towards extremely efficient upcycled mixture-of-experts models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025*, 2025.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*. Computer Vision Foundation / IEEE, 2019.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8580–8589, 2018.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part V*, volume 12350 of *Lecture Notes in Computer Science*, pp. 491–507. Springer, 2020.

Vladimir Koltchinskii. Local rademacher complexities and oracle inequalities in risk minimization. *Ann. Statist.*, 34(6):2593–2656, 12 2006.

Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, Minghai Qin, and Yanzhi Wang. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XI*, volume 13671 of *Lecture Notes in Computer Science*, pp. 620–640. Springer, 2022.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2013, Sydney, Australia, December 1-8, 2013*, pp. 554–561. IEEE Computer Society, 2013.

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the nyström method. *J. Mach. Learn. Res.*, 13:981–1006, 2012.

Jongmin Lee, Joo Young Choi, Ernest K. Ryu, and Albert No. Neural tangent kernel analysis of deep narrow neural networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 12282–12351. PMLR, 2022.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022a.

Yawei Li, Kamil Adamczewski, Wen Li, Shuhang Gu, Radu Timofte, and Luc Van Gool. Revisiting random channel pruning for neural network compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*. IEEE, 2022b.

Zhikai Li, Junrui Xiao, Lianwei Yang, and Qingyi Gu. Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 17227–17236, 2023.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pp. 740–755. Springer, 2014.

Yang Lin, Tianyu Zhang, Peiqin Sun, Zheng Li, and Shuchang Zhou. Fq-vit: Post-training quantization for fully quantized vision transformer. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1173–1179. ijcai.org, 2022.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024*. IEEE, 2024a.

Xinyu Liu, Houwen Peng, Ningxin Zheng, Yuqing Yang, Han Hu, and Yixuan Yuan. Efficientvit: Memory efficient vision transformer with cascaded group attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pp. 14420–14430. IEEE, 2023b.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is your multi-modal model an all-around player? In *Computer Vision - ECCV 2024 - 18th European Conference*. Springer, 2024b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9992–10002. IEEE, 2021.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, 2022.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. H-consistency bounds: Characterization and extensions. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023.

Sachin Mehta and Mohammad Rastegari. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Shahar Mendelson. Geometric parameters of kernel machines. In Jyrki Kivinen and Robert H. Sloan (eds.), *Conference on Computational Learning Theory*, volume 2375 of *Lecture Notes in Computer Science*, pp. 29–43. Springer, 2002.

Jisoo Mok, Byunggook Na, Ji-Hoon Kim, Dongyoon Han, and Sungroh Yoon. Demystifying the neural tangent kernel from a practical perspective: Can it be trusted for neural architecture search without training? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 11851–11860. IEEE, 2022.

Grégoire Montavon, Mikio L. Braun, and Klaus-Robert Müller. Kernel analysis of deep networks. *J. Mach. Learn. Res.*, 12:2563–2581, 2011.

Tan Nguyen, Minh Pham, Tam Nguyen, Khai Nguyen, Stanley J. Osher, and Nhat Ho. Fourierformer: Transformer meets generalized fourier integral theorem. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Tan Minh Nguyen, Tam Minh Nguyen, Nhat Ho, Andrea L. Bertozzi, Richard G. Baraniuk, and Stanley J. Osher. A primal-dual framework for transformers and neural networks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Eshaan Nichani, Yu Bai, and Jason D. Lee. Identifying good directions to escape the NTK regime and efficiently learn low-degree plus sparse polynomials. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Sixth Indian Conference on Computer Vision, Graphics & Image Processing, ICVGIP 2008, Bhubaneswar, India, 16-19 December 2008*, pp. 722–729. IEEE Computer Society, 2008.

Lorenzo Papa, Paolo Russo, Irene Amerini, and Luping Zhou. A survey on efficient vision transformers: Algorithms, techniques, and performance benchmarking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):7682–7700, 2024.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pp. 3498–3505. IEEE Computer Society, 2012.

John Rachwan, Daniel Zügner, Bertrand Charpentier, Simon Geisler, Morgane Ayle, and Stephan Günnemann. Winning the lottery ahead of time: Efficient early network pruning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 18293–18309. PMLR, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021.

Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 10425–10433. Computer Vision Foundation / IEEE, 2020.

Garvesh Raskutti, Martin J. Wainwright, and Bin Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *J. Mach. Learn. Res.*, 15(1):335–366, 2014.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015.

Mariia Seleznova and Gitta Kutyniok. Analyzing finite neural networks: Can we trust neural tangent kernel theory? In *Mathematical and Scientific Machine Learning*, pp. 868–895. PMLR, 2022.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 618–626. IEEE Computer Society, 2017.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*. Computer Vision Foundation / IEEE, 2019.

Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. FLAVA: A foundational language and vision alignment model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 15617–15629. IEEE, 2022.

Kyungwoo Song, Yohan Jung, Dongjun Kim, and Il-Chul Moon. Implicit kernel attention. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 9713–9721. AAAI Press, 2021.

Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiao-Gang Wang, and Chang Xu. Vitas: Vision transformer architecture search. In *European Conference on Computer Vision (ECCV)*, volume 13681 of *Lecture Notes in Computer Science*, pp. 139–157. Springer, 2022.

Shokichi Takakura and Taiji Suzuki. Mean-field analysis on two-layer neural networks from a kernel perspective. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 2820–2828. Computer Vision Foundation / IEEE, 2019.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357. PMLR, 2021.

Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, Peter Vajda, and Joseph E. Gonzalez. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 12962–12971. Computer Vision Foundation / IEEE, 2020.

Yite Wang, Dawei Li, and Ruoyu Sun. NTK-SAP: improving neural network pruning by aligning training dynamics. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Zhenyu Wang, Hao Luo, Pichao Wang, Feng Ding, Fan Wang, and Hao Li. VTC-LFC: vision transformer compression with low-frequency components. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Tianxin Wei, Zeming Guo, Yifan Chen, and Jingrui He. Ntk-approximating MLP fusion for efficient language model fine-tuning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36821–36838. PMLR, 2023.

Zimian Wei, Peijie Dong, Zheng Hui, Anggeng Li, Lujun Li, Menglong Lu, Hengyue Pan, and Dongsheng Li. Auto-prox: Training-free vision transformer architecture search via automatic proxy discovery. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational*

Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada, pp. 15814–15822. AAAI Press, 2024.

Blake E. Woodworth, Suriya Gunasekar, Jason D. Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In Jacob D. Abernethy and Shivani Agarwal (eds.), *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pp. 3635–3673. PMLR, 2020.

Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 10734–10742. Computer Vision Foundation / IEEE, 2019.

Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. Tinyvit: Fast pretraining distillation for small vision transformers. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXI*, volume 13681 of *Lecture Notes in Computer Science*, pp. 68–85. Springer, 2022.

Lechao Xiao, Jeffrey Pennington, and Samuel Stern Schoenholz. Disentangling trainability and generalization in deep neural networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10462–10472. PMLR, 2020.

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, volume 11209 of *Lecture Notes in Computer Science*, pp. 432–448. Springer, 2018.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. Visual entailment: A novel task for fine-grained image understanding. *CoRR*, abs/1901.06706, 2019.

Kaixin Xu, Zhe Wang, Chunyun Chen, Xue Geng, Jie Lin, Xulei Yang, Min Wu, Xiaoli Li, and Weisi Lin. Lpvit: Low-power semi-structured pruning for vision transformers. In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXI*, volume 15129 of *Lecture Notes in Computer Science*, pp. 269–287. Springer, 2024.

Zhendong Yang, Ailing Zeng, Chun Yuan, and Yu Li. From knowledge distillation to self-knowledge distillation: A unified approach with normalized loss and customized soft labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17185–17194, 2023.

Zhendong Yang, Zhe Li, Ailing Zeng, Zexian Li, Chun Yuan, and Yu Li. Vitkd: Feature-based knowledge distillation for vision transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024 - Workshops, Seattle, WA, USA, June 17-18, 2024*, pp. 1379–1388. IEEE, 2024.

Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis E. H. Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 538–547. IEEE, 2021.

Yuyao Zhang, Lan Wei, and Nikolaos M. Freris. Synergistic patch pruning for vision transformer: Unifying intra- & inter-layer patch importance. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

Borui Zhao, Renjie Song, and Jiajun Liang. Cumulative spatial knowledge distillation for vision transformers. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 6146–6155, 2023.

Chuanyang Zheng, Zheyang Li, Kai Zhang, Zhi Yang, Wenming Tan, Jun Xiao, Ye Ren, and Shiliang Pu. Savit: Structure-aware vision transformer pruning via collaborative optimization. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 6881–6890. Computer Vision Foundation / IEEE, 2021.

Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *Int. J. Comput. Vis.*, 127(3):302–321, 2019.

Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and José M. Álvarez. Understanding the robustness in vision transformers. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, Proceedings of Machine Learning Research, pp. 27378–27394. PMLR, 2022.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

A Proof of Theorem 3.1

Proof of Theorem 3.1. We consider the dynamic kernel $K(\mathbf{x}, \mathbf{x}') = F(\mathbf{x})^\top F(\mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ defined in Section 3.2. Then it follows from (Bartlett et al., 2005, Theorem 3.3) that for every $K_0 > 1$,

$$L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) \le \frac{K_0}{K_0 - 1}\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] + 6K_0 r^* + \Theta\left(\frac{\delta}{n}\right),$$
$$L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) \ge \frac{K_0}{K_0 + 1}\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] - 6K_0 r^* - \Theta\left(\frac{\delta}{n}\right), \tag{3}$$

and each inequality in (3) holds with probability at least $1 - \exp(-\delta)$. r^* is the fixed point of the kernel complexity of the dynamic kernel following (Raskutti et al., 2014), where r^* is also referred to as the critical population rate. K is a positive definite kernel K and the network $\mathrm{NN}_{\mathcal{W}} \in \mathcal{H}_K$ where $\mathcal{H}_K$ denotes the Reproducing Kernel Hilbert Space (RKHS) associated with K, and we recall that $\mathrm{KC}(\mathbf{K}) = \min_{h \in [0, r_0]} \left(\frac{h}{n} + \sqrt{\frac{1}{n}\sum_{i=h+1}^{r_0} \widehat{\lambda}_i}\right)$ is the kernel complexity. It then follows from (Bartlett et al., 2005, Corollary 6.7) and (3) that

$$L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) \le \frac{K_0}{K_0 - 1}\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] + 6K_0 \mathrm{KC}(\mathbf{K}) + \Theta\left(\frac{\delta}{n}\right),$$
$$L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}}) \ge \frac{K_0}{K_0 + 1}\mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right] - 6K_0 \mathrm{KC}(\mathbf{K}) - \Theta\left(\frac{\delta}{n}\right). \tag{4}$$

Combining (3) and (4) proves (1), and we set $K_0 = 3$ in this theorem.

□

A.1 Extension of Theorem 3.1 to the Cross-Entropy Loss

The KCR upper and lower bounds in Theorem 3.1 are stated for the expected risk under the squared loss, whereas the training loss (2) combines the KCR regularization term with the cross-entropy (CE) loss. In this section, we show that the bounds in (1) also hold with the squared loss replaced by the CE loss, with only constant-factor changes, so that the training loss (2) directly minimizes the empirical-loss term and the KC term of the resulting CE generalization bound.

Logits and post-softmax probabilities. We first clarify the notation. In Theorem 3.1, the squared loss $\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2$ is applied to the post-softmax probability vector $\mathbf{p} = \mathrm{softmax}(X(\mathcal{W}))$, where $X(\mathcal{W}) \in \mathbb{R}^C$ denotes the logits of the network. In contrast, the CE loss $\mathrm{CE}_j^{(t)}(\mathcal{W})$ in (2) is computed from the logits $X_i(\mathcal{W})$ through the cross-entropy function $H(\cdot, \cdot)$. Let $\mathbf{z}_i = X_i(\mathcal{W}) \in \mathbb{R}^C$ be the logits of the i-th sample and $\mathbf{p}_i = \mathrm{softmax}(\mathbf{z}_i)$. For a one-hot label $\mathbf{y} = \mathbf{e}_c$, where $c \in [C]$ is the ground-truth class, we omit the sample index and denote the CE loss and the squared (MSE) loss by

$$L_{\mathrm{CE}} = -\log p_c, \qquad L_{\mathrm{MSE}} = \|\mathbf{p} - \mathbf{e}_c\|_2^2,$$

respectively, where p_c is the c-th entry of $\mathbf{p}$.

Comparison between the MSE and CE losses. Setting $q = 1 - p_c = \sum_{j \ne c} p_j$, we have $L_{\mathrm{MSE}} = q^2 + \sum_{j \ne c} p_j^2$. Since $\frac{q^2}{C-1} \le \sum_{j \ne c} p_j^2 \le q^2$ and $p_c = e^{-L_{\mathrm{CE}}}$, we have

$$\frac{C}{C-1}\left(1 - e^{-L_{\mathrm{CE}}}\right)^2 \le L_{\mathrm{MSE}} \le 2\left(1 - e^{-L_{\mathrm{CE}}}\right)^2 \le 2L_{\mathrm{CE}}. \tag{5}$$

On the bounded loss range $0 < \ell \le L_{\mathrm{CE}} \le B < \infty$, we define

$$a_{\ell,B} := \frac{C}{C-1}\min\left\{\frac{\left(1 - e^{-\ell}\right)^2}{\ell}, \frac{\left(1 - e^{-B}\right)^2}{B}\right\} > 0. \tag{6}$$

The two losses then satisfy

$$a_{\ell,B} L_{\mathrm{CE}} \le L_{\mathrm{MSE}} \le 2L_{\mathrm{CE}}. \tag{7}$$

Let $\widehat{L}_{\mathrm{MSE}}$ and $\widehat{L}_{\mathrm{CE}}$ denote the empirical MSE loss and the empirical CE loss on the training data, respectively, so that $\widehat{L}_{\mathrm{MSE}} = \mathbb{E}_{P_n}\left[\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2\right]$ is the empirical loss in Theorem 3.1. Let $L_{\mathcal{D}}^{\mathrm{MSE}}$ and $L_{\mathcal{D}}^{\mathrm{CE}}$ denote the expected MSE loss and the expected CE loss, respectively, so that $L_{\mathcal{D}}^{\mathrm{MSE}} = L_{\mathcal{D}}(\mathrm{NN}_{\mathcal{W}})$. It then follows from (7) that

$$a_{\ell,B}\widehat{L}_{\mathrm{CE}} \le \widehat{L}_{\mathrm{MSE}} \le 2\widehat{L}_{\mathrm{CE}}, \qquad a_{\ell,B}L_{\mathcal{D}}^{\mathrm{CE}} \le L_{\mathcal{D}}^{\mathrm{MSE}} \le 2L_{\mathcal{D}}^{\mathrm{CE}}. \tag{8}$$

KCR upper and lower bounds with the CE loss. Substituting the relations in (8) into (1) of Theorem 3.1, which holds for every $\delta > 0$ with probability at least $1 - \exp(-\delta)$, yields, up to universal constants,

$$\frac{a_{\ell,B}}{2}\widehat{L}_{\mathrm{CE}} - \frac{1}{2}\mathrm{KC}(\mathbf{K}) - O\left(\frac{\delta}{n}\right) \lesssim L_{\mathcal{D}}^{\mathrm{CE}} \lesssim \frac{2}{a_{\ell,B}}\widehat{L}_{\mathrm{CE}} + \frac{1}{a_{\ell,B}}\mathrm{KC}(\mathbf{K}) + O\left(\frac{\delta}{n}\right). \tag{9}$$

Therefore, Theorem 3.1 also holds with the squared loss $\|\mathrm{NN}_{\mathcal{W}}(\mathbf{x}) - \mathbf{y}\|_2^2$ replaced by the CE loss, with only constant-factor changes. Thus, the training loss (2) directly minimizes the empirical-loss term and the KC term appearing in the resulting CE generalization bound (9), where the KC term is reduced through the KCR regularization term $\mathrm{KCR}_j(\mathcal{W})$. The constant factors in (9) are absorbed into the tunable balance factor η in (2).

Relation to the squared-loss bound. It follows from (5) and (8) that the CE loss controls the empirical squared-loss term in (1), since $\widehat{L}_{\mathrm{MSE}} \le 2\widehat{L}_{\mathrm{CE}}$, while the KCR regularization term controls the KC term. Accordingly, Theorem 3.1 remains a bound for the squared (Brier) risk of the post-softmax prediction $\mathbf{p}$, and the training loss (2) is its valid practical surrogate.

The bounded loss range $0 < \ell \le L_{\mathrm{CE}} \le B < \infty$. The condition $0 < \ell \le L_{\mathrm{CE}} \le B < \infty$ follows from the standard bounded-score assumption. Let $\mathbf{z}(\mathbf{x}) \in \mathbb{R}^C$ denote the logit vector of the network for the input $\mathbf{x}$, so that $\mathbf{z}_i = \mathbf{z}(\mathbf{x}_i)$. If $|[\mathbf{z}(\mathbf{x})]_j| \le R$ for every class $j \in [C]$, where $[\mathbf{z}(\mathbf{x})]_j$ is the j-th entry of $\mathbf{z}(\mathbf{x})$, then

$$\ell = \log\left(1 + (C-1)e^{-2R}\right) \le L_{\mathrm{CE}} \le \log\left(1 + (C-1)e^{2R}\right) = B. \tag{10}$$

The bounded-score assumption is the output-level consequence of using a norm-bounded hypothesis class. Specifically, we consider the uniformly bounded score class

$$\mathcal{H}_R = \left\{\mathbf{h} \colon \sup_{\mathbf{x},j} |[\mathbf{h}(\mathbf{x})]_j| \le R\right\},$$

where $\mathbf{h}(\mathbf{x}) \in \mathbb{R}^C$ denotes the scores of the C classes for the input $\mathbf{x}$. Such a class follows from bounded inputs and bounded parameter norms. For example, for a linear scoring function, $\|\mathbf{x}\|_2 \le c_x$, $\|\mathbf{w}_j\|_2 \le c_w$, and $|b_j| \le c_b$ imply $\left|\mathbf{w}_j^\top \mathbf{x} + b_j\right| \le c_w c_x + c_b$. Layerwise norm constraints analogously bound the logits of neural networks. Bounded logits then ensure that the softmax probabilities are bounded away from 0 and 1, yielding $0 < \ell \le L_{\mathrm{CE}} \le B < \infty$. Bounded hypothesis classes are standard in the generalization analyses of the logistic loss and neural networks, and the bounded hypothesis classes studied in prior works (Hazan et al., 2014; Golowich et al., 2018; Mao et al., 2023) support the above bounded-score assumption in this sense. In particular, logistic regression is explicitly restricted to bounded inputs and a bounded parameter domain for generalization (Hazan et al., 2014, Section 2.1); the Rademacher complexity bounds for neural networks are derived under layerwise norm constraints (Golowich et al., 2018), which imply bounded logits for bounded inputs; and the multiclass logistic loss is directly analyzed over bounded scoring functions (Mao et al., 2023, Section 4.2). As a result, $0 < \ell \le L_{\mathrm{CE}} \le B < \infty$ is a standard bounded-loss and bounded-score condition, and the lower bound $\ell > 0$ is specifically needed for a positive linear comparison between the MSE and CE losses in (7).

A.2 Validity of the KCR Bounds Before and After Channel Selection

In this section, we make explicit the connection between the KCR regularization term, the KCR upper and lower bounds in Theorem 3.1, and the channel selection described in Section 3.1. We show formally that the bounds hold for the networks both before and after channel selection, and empirically that the KCR regularization reduces the KCR upper bound after channel selection.

Connection between the KCR term and channel selection. The kernel complexity $\mathrm{KC}(\mathbf{K})$ is a function of the selected channels, since the dynamic kernel K and its gram matrix $\mathbf{K}$ are computed from the feature backbone F of the network with the selected channels. The channels to be pruned are selected by reducing the KCR regularization term, which in turn reduces the kernel complexity $\mathrm{KC}(\mathbf{K})$.

KCR formally controls the bounds during and after channel selection. The KCR upper and lower bounds in (1) hold before and after channel selection, since the bounds hold for each dynamic kernel, while the networks before and after channel selection correspond to two particular dynamic kernels. In particular, channel selection preserves the positive semidefiniteness of the gram matrix. For either the backbone before channel selection or the resulting backbone after channel selection, let F denote its feature map. The kernel is always $K(\mathbf{x}, \mathbf{x}') = F(\mathbf{x})^\top F(\mathbf{x}')$ as defined in Section 3.2. For arbitrary samples $\{\mathbf{x}_i\}_{i=1}^n$ and coefficients $\boldsymbol{a} \in \mathbb{R}^n$, with $\mathbf{K}$ being the gram matrix of K over $\{\mathbf{x}_i\}_{i=1}^n$, we have

$$\boldsymbol{a}^\top \mathbf{K} \boldsymbol{a} = \sum_{i,j=1}^{n} a_i a_j \langle F(\mathbf{x}_i), F(\mathbf{x}_j) \rangle = \left\| \sum_{i=1}^{n} a_i F(\mathbf{x}_i) \right\|_2^2 \geq 0. \tag{11}$$

Hence $\mathbf{K} \succeq 0$ for any feature map F, including the feature maps produced by hard or soft channel selection, i.e., by the binary decision mask or by its Gumbel-Softmax relaxation in Section 3.1. Channel selection can change the rank, the eigenvalues, and consequently the kernel complexity $\mathrm{KC}(\mathbf{K})$, but it cannot introduce negative eigenvalues. The same argument applies after retraining by taking F to be the retrained feature map. Therefore, Theorem 3.1 applies separately to the networks before and after channel selection. Combined with the comparison between the MSE and CE losses in Section A.1, this justification also transfers to the CE loss.

Terminology. The term "positive definite kernel" in Section 3.2 is used in the standard sense of the kernel learning literature, which indicates that the associated gram matrix is positive semidefinite. Strict positive definiteness of the gram matrix would additionally require linearly independent sample features, which is neither required by Theorem 3.1 nor assumed by the definition of $\mathrm{KC}(\mathbf{K})$.

KCR empirically controls the bounds during and after channel selection. The kernel complexity $\mathrm{KC}(\mathbf{K})$ depends on the dynamic kernel of the selected network. In the search stage, the channel mask is obtained by jointly optimizing the training loss (2) and the computational cost through the search loss in Section 3.3. The searched and pruned network is then retrained with the KCR regularization, which reduces the approximate TNN and consequently reduces $\mathrm{KC}(\mathbf{K})$. Table 6 reports the KC and the KCR upper and lower bounds for the original uncompressed baseline before pruning and for the network after pruning with KCR. Following the proof of Theorem 3.1, the KC is scaled by $6K_0$. The balance factor η is selected by cross-validation as described in Section 4.1, which gives $\eta = 0.2$ for Swin-B and $\eta = 1.0$ for ViT-B. All the entries in Table 6 are identical to the corresponding entries of Table 21 in Section D.15, namely its column for the uncompressed baseline and its columns for $\eta = 0.2$ (Swin-B) and $\eta = 1.0$ (ViT-B).

Table 6: KC, training loss, KCR lower and upper bounds, and top-1 accuracy on ImageNet-1K before pruning (the original uncompressed baseline) and after pruning with KCR. The entries are taken from Table 21, where η is selected by cross-validation. Training Loss denotes the empirical MSE loss. The value of KC is scaled with a constant multiplier of $6K_0$ as in the proof of Theorem 3.1. After pruning, Swin-B is reduced from 88.0M parameters and 15.4G FLOPs to 70.2M parameters and 12.6G FLOPs, and ViT-B is reduced from 86.5M parameters and 17.6G FLOPs to 69.5M parameters and 14.5G FLOPs.

Base Model	Stage	η	KC	Training Loss	KCR Lower Bound	KCR Upper Bound	Top-1 (%)
Swin-B	Before pruning	–	0.46	0.62	0.01	1.39	83.5
	After pruning + KCR	0.2	0.16	**0.35**	0.10	**0.69**	**84.7**
ViT-B	Before pruning	–	0.42	0.58	0.02	1.29	83.7
	After pruning + KCR	1.0	0.16	**0.28**	0.05	**0.58**	**84.6**

As shown in Table 6, after channel selection with KCR, the KC decreases substantially, from 0.46 to 0.16 for Swin-B and from 0.42 to 0.16 for ViT-B. The KCR upper bound is reduced from 1.39 to 0.69 for Swin-B and from 1.29 to 0.58 for ViT-B, and the top-1 accuracy is improved from 83.5%

to 84.7% for Swin-B and from 83.7% to 84.6% for ViT-B. These results, together with the formal argument above, confirm that the KCR bounds still hold after channel selection and that the KCR regularization effectively reduces the KCR upper bound after pruning.

B ALGORITHMS

Algorithm 1 describes the search and the retraining process of the KCR-Transformer.

Algorithm 1 Training Algorithm with the Approximate Truncated Nuclear Norm by SGD

Input: Training dataset $\mathcal{D}$, number of search epochs t_{search}, number of training epochs t_{train}, warm-up epochs t_{warm}, batch size J, learning rates $\eta_{\mathcal{W}}$ and η_α
Output: Trained weights $\mathcal{W}$ of the network

Split $\mathcal{D}$ into $\mathcal{D}_{\mathcal{W}}$ and $\mathcal{D}_\alpha$, where 70% of samples ($\mathcal{D}_{\mathcal{W}}$) are used to update model weights $\mathcal{W}$, and 30% of samples ($\mathcal{D}_\alpha$) are used to optimize architecture parameters α.
Initialize the network weights $\mathcal{W} = \mathcal{W}(0)$ by random initialization.
for $t \leftarrow 1$ to t_{search} **do**
  for $j \leftarrow 1$ to J **do**
    Sample mini-batches $\mathcal{B}_j^{\mathcal{W}} \subset \mathcal{D}_{\mathcal{W}}$, $\mathcal{B}_j^{\alpha} \subset \mathcal{D}_\alpha$.
    Perform gradient descent on $\mathcal{B}_j^{\alpha}$ to update α by $\alpha \leftarrow \alpha - \eta_\alpha \nabla_\alpha \mathcal{L}^{(t)}_{\text{search},j}(\mathcal{W}, \alpha)$
    Perform gradient descent on $\mathcal{B}_j^{\mathcal{W}}$ to update $\mathcal{W}$ by $\mathcal{W} \leftarrow \mathcal{W} - \eta_{\mathcal{W}} \nabla_{\mathcal{W}} \mathcal{L}^{(t)}_{\text{search},j}(\mathcal{W}, \alpha)$
  end for
end for
for $t \leftarrow 1$ to t_{train} **do**
  if $t \bmod 30 = 0$ **then**
    Update $\tilde{\mathbf{U}}^{(r_0)}$, the approximation of $\mathbf{U}^{(r_0)}$.
  end if
  for $j \leftarrow 1$ to J **do**
    Sample mini-batch $\mathcal{B}_j \subset \mathcal{D}_{\mathcal{W}}$.
    if $t > t_{\text{warm}}$ **then**
      Perform gradient descent on $\mathcal{B}_j^{\mathcal{W}}$ to update $\mathcal{W}$ by $\mathcal{W} \leftarrow \mathcal{W} - \eta_{\mathcal{W}} \nabla_{\mathcal{W}} \mathcal{L}^{(t)}_{\text{train},j}(\mathcal{W})$
    else
      Perform gradient descent on $\mathcal{B}_j^{\mathcal{W}}$ to update $\mathcal{W}$ by $\mathcal{W} \leftarrow \mathcal{W} - \eta_{\mathcal{W}} \nabla_{\mathcal{W}} \text{CE}_j^{(t)}(\mathcal{W})$
    end if
  end for
end for
return $\mathcal{W}$

B.1 DIMENSION PROPAGATION OF THE SELECTED CHANNELS ACROSS LAYERS

In this section, we clarify how the feature dimensions reduced by channel selection propagate across the layers of a network with KCR-Transformer blocks. As described in Section 3.1, the decision mask $g \in \{0,1\}^D$ of a KCR-Transformer block is multiplied with both the input and output features of its MLP layers, and after the search the gather operation on the selected channels changes the dimension of these features from D to $\tilde{D} = \sum_{i=1}^{D} g_i$. The reduced output dimension is not recovered to its original dimension in the next layer. Instead, the input dimension of each linear layer is always kept consistent with the output dimension of its preceding layer.

Specifically, let the j-th linear layer of the original network map D_{j-1} input channels to D_j output channels, where, in this section, j indexes the linear layers of the network rather than the KCR-Transformer blocks. After channel selection, this layer maps $\tilde{D}_{j-1}$ input channels to $\tilde{D}_j$ output channels, that is,

$$\mathbb{R}^{D_{j-1}} \to \mathbb{R}^{D_j} \quad \Longrightarrow \quad \mathbb{R}^{\tilde{D}_{j-1}} \to \mathbb{R}^{\tilde{D}_j}, \qquad \tilde{D}_j = \sum_{i=1}^{D_j} g_{j,i} \le D_j,$$

where $\tilde{D}_{j-1}$ is the selected output dimension of the preceding layer, $\tilde{D}_j$ is selected from the original D_j output channels of the current layer, and $g_{j,i} \in \{0,1\}$ indicates whether the i-th output channel

of the j-th linear layer is selected. The next linear layer then takes this $\tilde{D}_j$-dimensional output as its input.

No recursive reduction across depth. Each $\tilde{D}_j$ is selected with respect to the original output dimension D_j of the corresponding layer; it is not obtained by repeatedly multiplying the preceding compressed dimension by a pruning ratio. Therefore, there is no recursive reduction of the form $\tilde{D}_j = \rho\,\tilde{D}_{j-1}$ with a pruning ratio ρ, and the compression is not compounded across depth. As a result, the depth of the network does not cause the feature dimension to progressively shrink to 0 in deep blocks. An all-zero mask would instead correspond to a degenerate searched architecture with no valid task output, rather than being a consequence of stacking deep blocks.

Evidence from the learned masks. The learned channel-selection masks of KCR-ViT-S and KCR-ViT-B visualized in Figure 4 of Section D.19 show that every layer retains channels after the search, while the shallow layers retain more channels and the middle layers are compressed more aggressively.

C Implementation Details

During architecture search, we randomly sample 100 ImageNet classes (Russakovsky et al., 2015) for training. The network is trained for 200 epochs with AdamW and a cosine learning rate schedule, decaying the learning rate from 0.001 to 0.0001. In each epoch, 70% of the training data updates the model weights, while the remaining 30% optimizes the KCR-Transformer architecture parameters. Empirically, $t_{\text{warm}} = 90$ achieves the best performance across all KCR variants. During retraining, the searched network is trained on ImageNet-1K with AdamW using $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the total training epochs to $t_{\text{train}} = 300$ and use $t_{\text{warm}} = 90$ for all experiments. We also search the feature rank $r = \lceil \gamma_0 \min(n, d) \rceil$, where γ_0 is the rank ratio. The values of γ_0 and η are selected by 5-fold cross-validation, with γ_0 chosen from 0.1 to 0.5 at intervals of 0.05, and η chosen from $\{0.01, 0.05, 0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0, 2.5\}$. Throughout all experiments, we set $m = 50000$. As shown in Section D.7, KCR-Transformer is stable under different choices of γ_0, η, and m. The hyperparameter λ controls the searched architecture size and is set to 0.2 in the main experiments. Section D.8 reports KCR-Transformers compressed to different sizes by varying λ. To improve training efficiency, we compute $\tilde{\mathbf{U}}^{(r_0)}$, the approximation of $\mathbf{U}^{(r_0)}$, every 30 epochs.

All experiments are conducted on 40GB NVIDIA A100 GPUs with an effective batch size of 512. Following standard practice (Cai et al., 2023), we apply widely adopted data augmentation techniques during training, including random scaling, random horizontal flipping, and random cropping. The weight decay is set to 0.01. The learning rate is linearly increased from 0.0002 to 0.002 over the first five epochs, then gradually annealed back to 0.0002 using a cosine decay schedule over the remaining epochs. Inference is performed using the exponential moving average (EMA) of model weights.

During the search process, the temperature τ is initialized at 4.5 and decayed by a factor of 0.95 every epoch over the search phase, and the Gumbel noise is sampled per mini-batch from a standard Gumbel$(0, 1)$ distribution following the common settings in differentiable pruning and NAS literature (Wan et al., 2020; Chang et al., 2019; Wu et al., 2019).

D Additional Experiment Results

D.1 Transfer Learning Capability of the KCR-Transformer

We evaluate the transfer learning capability of KCR-ViT-B in comparison to the baseline ViT-B using three widely adopted benchmarks, including Oxford Flowers-102 (Nilsback & Zisserman, 2008), Oxford-IIIT Pet (Parkhi et al., 2012), and Stanford Cars (Krause et al., 2013). Following the established transfer learning protocol in (Kolesnikov et al., 2020), both models are first pre-trained on ImageNet and subsequently fine-tuned on the respective training sets of the target datasets for 50 epochs using the Adam optimizer, with a fixed learning rate of 1×10^{-5}. The experimental results are presented in Table 7. It is observed that KCR-ViT-B consistently outperforms ViT-B and NTK-SAP-ViT-B across all three datasets, while requiring fewer FLOPs and model parameters. For example, KCR-ViT-B achieves a top-1 accuracy of 93.8%, outperforming ViT-B and NTK-SAP-ViT-B by 1.1% and 1.3%, respectively, on the Cars dataset. The improvements demonstrate the

superior transferability of the features learned by the KCR compressed model across diverse visual domains.

Table 7: Top-1 classification accuracy comparison for transfer learning on the Oxford Flowers-102 (Nilsback & Zisserman, 2008), Oxford-IIIT Pet (Parkhi et al., 2012), and Stanford Cars (Krause et al., 2013) datasets.

Model	# Params	FLOPs	ImageNet (%)	Flowers (%)	Pet (%)	Cars (%)
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	83.7	97.8	96.0	92.7
NTK-SAP-ViT-B (Wang et al., 2023)	71.8 M	15.6 G	83.5	97.4	95.6	92.5
KCR-ViT-B	69.5 M	14.5 G	**84.6**	**98.2**	**96.6**	**93.8**

D.2 Self-Supervised Learning with KCR-Transformers

We further evaluate the effectiveness of KCR-Transformers in the self-supervised learning (SSL) setting using both MoCoV3 (Chen et al., 2021d) and MOCA (Gidaris et al., 2024). In our experiments, the ViT-B model pre-trained with each SSL method serves as the baseline. We pre-train ViT-B, NTK-SAP-ViT-B, and our KCR-ViT-B strictly following the training settings described in the respective papers (Chen et al., 2021d; Gidaris et al., 2024) on the ImageNet1K without using training labels. The pre-trained models are then subsequently fine-tuned with class labels for downstream classification on ImageNet1K following the fine-tuning protocols in (Chen et al., 2021d; Gidaris et al., 2024). As shown in Table 8, KCR-ViT-B consistently achieves superior performance over ViT-B and NTK-SAP-ViT-B across both SSL pipelines, while requiring fewer FLOPs and parameters. For example, under the MOCA (Gidaris et al., 2024) pretraining pipeline, KCR-ViT-B achieves a top-1 accuracy of 84.4%, outperforming ViT-B and NTK-SAP-ViT-B by 1.0% and 1.2%, respectively, with reduced computational cost and parameter size. The improvements highlight the effectiveness of the KCR compression method in enhancing the efficiency and accuracy of the models pre-trained by SSL methods.

Table 8: Top-1 classification accuracy comparison for models pre-trained with the self-supervised learning (SSL) methods MoCoV3 (Chen et al., 2021d) and MOCA (Gidaris et al., 2024) on ImageNet1K.

Network	SSL Method	# Params	FLOPs	Top-1 (%)
ViT-B (Dosovitskiy et al., 2021)	MoCoV3 (Chen et al., 2021d)	86.5 M	17.6 G	83.2
NTK-SAP-ViT-B (Wang et al., 2023)		71.8 M	15.6 G	82.9
KCR-ViT-B (Ours)		69.5 M	14.5 G	**84.1**
ViT-B (Dosovitskiy et al., 2021)	MOCA (Gidaris et al., 2024)	86.5 M	17.6 G	83.4
NTK-SAP-ViT-B (Wang et al., 2023)		71.8 M	15.6 G	83.2
KCR-ViT-B (Ours)		69.5 M	14.5 G	**84.4**

D.3 Implementation Details for Semantic Segmentation

ADE20K has 25000 images in total, with 20000 for training, 2000 for validation, and another 3000 for testing. We adopt UperNet (Xiao et al., 2018) as the segmentation framework with our KCR-Swin-B as the feature extraction backbone. We follow the training and evaluation protocol in (Liu et al., 2021), where both our model and the baselines are trained on the training split and evaluated on the validation split of the dataset. All models are optimized using AdamW for a total of 160000 iterations with a batch size of 16, an initial learning rate of 6×10^{-5}, and a weight decay of 0.01. The learning rate follows a linear decay schedule after a warm-up phase of 1500 iterations. To enhance generalization, we employ data augmentation techniques including random horizontal flipping, random rescaling with a ratio range of $[0.5, 2.0]$, and random photometric distortions. Stochastic depth regularization is applied with a drop rate of 0.2. For inference, we use multi-scale testing with scale factors varying from 0.5 to 1.75.

D.4 Implementation Details and Additional Results for Object Detection

We incorporate the ImageNet pre-trained KCR-Swin-T and KCR-Swin-B into the Cascade Mask R-CNN framework (Cai & Vasconcelos, 2021) for object detection. All models are evaluated on the

MS-COCO dataset (Lin et al., 2014), which consists of 117000 training images and 5000 validation images. We follow the training configuration of (Liu et al., 2021), where each input image is resized such that the shorter side falls within $[480, 800]$ pixels while the longer side does not exceed 1333 pixels. Training is performed using the AdamW optimizer with an initial learning rate of 0.0001, a weight decay of 0.05, and a batch size of 16, for a total of 36 epochs following the $3\times$ schedule. In line with (Cai & Vasconcelos, 2021), we report standard COCO metrics, including the box-level mean Average Precision (mAP^{box}) and mask-level mean Average Precision (mAP^{m}), as well as AP at IoU thresholds of 50 and 75.

Table 9: Detailed Object Detection Results on COCO.

Detection Framework	Feature Backbone	mAP^{box} (%)	AP^{b}_{50} (%)	AP^{b}_{75} (%)	mAP^{m} (%)	AP^{m}_{50} (%)	AP^{m}_{75} (%)
Mask R-CNN	Swin-T	50.5	69.3	54.9	43.7	66.6	47.1
Mask R-CNN	Swin-B	51.9	70.9	56.5	45.0	68.4	48.7
Mask R-CNN	KCR-Swin-T (Ours)	50.9	69.7	55.3	44.0	67.1	47.6
Mask R-CNN	KCR-Swin-B (Ours)	52.5	71.4	56.8	45.6	68.9	49.1

It is observed in Table 9 that compressing the Swin backbones with KCR consistently improves both box-level and mask-level detection performance within the Cascade Mask R-CNN framework. For example, KCR-Swin-T achieves a box mAP of 50.9% and a mask mAP of 44.0%, with improvements of 0.4% and 0.3% over the standard Swin-T baseline. Similarly, KCR-Swin-B achieves the highest box mAP of 52.5% and mask mAP of 45.6%, surpassing the Swin-B baseline by 0.6% and 0.6%, respectively. These results demonstrate that KCR compression effectively enhances the feature expressiveness of vision backbones for object detection, with even less computational cost. The consistent improvements across multiple IoU thresholds further validate the robustness and generalization capability of vision backbones compressed by KCR.

D.5 Efficiency Analysis of the Search and Re-Training

In this section, we first compare the training time of the KCR-Transformers with the corresponding baseline vision transformers without the KCR regularization. The comparison is performed on one NVIDIA A100 40GB GPU with full-precision, and the average training time per epoch is reported. It is observed from Table 10 that the KCR regularization only marginally increases the training time ($< 5.4\%$) compared to the corresponding baseline vision transformers, while greatly increasing the top-1 classification accuracy and reducing the model size of FLOPs. For example, KCR-Swin-B achieves a 1.2% top-1 accuracy improvement while reducing the number of parameters from 88.0M to 70.2M and FLOPs from 15.4G to 12.6G, with a negligible training overhead of 1.1 minutes per epoch, which is only 2.87% of the original training time of Swin-B. In addition, we also report the overall search time and re-training time in comparison with the training time of the baseline model. It is observed that the search phase only brings marginal training overhead, costing less than 7.2% of the baseline model's training time.

Table 10: Training time comparisons with baseline methods on the training set of ImageNet-1K.

Model	# Params	FLOPs	Top-1 (%)	Training Efficiency (Minutes/Epoch)	Search Time (Hours)	Training/Re-Training Time (Hours)
ViT-S (Dosovitskiy et al., 2021)	22.1 M	4.3 G	81.2	16.8	-	84.0
KCR-ViT-S (Ours)	19.8 M	3.8 G	**82.2**	17.7	6.0	88.5
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	83.7	33.5	-	167.5
KCR-ViT-B (Ours)	69.5 M	14.5 G	**84.6**	34.7	11.7	173.4
Swin-T (Liu et al., 2021)	29.0 M	4.5 G	81.3	20.8	-	104.0
KCR-Swin-T (Ours)	24.6 M	3.9 G	**82.4**	21.7	7.3	108.5
Swin-B (Liu et al., 2021)	88.0 M	15.4 G	83.5	38.3	-	191.5
KCR-Swin-B (Ours)	70.2 M	12.6 G	**84.7**	39.4	13.3	197.1

D.6 Evaluation on ImageNet-A, ImageNet-C, and ImageNet-R

To further evaluate the robustness of the compressed vision transformers under distribution shift, we compare Swin-B and its compressed variants on ImageNet-A (Hendrycks et al., 2021), ImageNet-C (Hendrycks & Dietterich, 2019), and ImageNet-R (Hendrycks & Dietterich, 2019). ImageNet-A

is a benchmark of natural adversarial examples designed to assess robustness to challenging real-world images. ImageNet-C evaluates corruption robustness by applying a diverse set of common corruptions with multiple severity levels, and the performance is measured by the mean corruption error (mCE), where a lower value indicates better robustness. ImageNet-R contains renditions of ImageNet classes, such as paintings, sketches, and cartoons, and is used to evaluate robustness to large appearance shifts. Following the evaluation protocol in (Zhou et al., 2022), we report top-1 accuracy on ImageNet-A and ImageNet-R, and mCE on ImageNet-C.

The results are presented in Table 11. It is observed that KCR-Swin-B consistently achieves the best robustness performance among all compressed models and even outperforms the uncompressed model. For instance, KCR-Swin-B improves the top-1 accuracy on ImageNet-A by 2.1% while requiring less FLOPs and parameters, indicating that the proposed KCR regularization improves not only standard in-distribution accuracy but also robustness under natural adversarial examples, common corruptions, and rendition shifts. This is because the KCR regularization reduces the kernel complexity of the learned representations, which encourages the model to capture more stable and generalizable feature structures rather than overfitting to spurious patterns in the training distribution. As a result, the compressed model exhibits improved resilience to diverse distribution shifts, including natural adversarial examples, common corruptions, and large appearance variations.

Table 11: Robustness evaluation on ImageNet-A, ImageNet-C, and ImageNet-R. Higher top-1 accuracy is better on ImageNet-A and ImageNet-R, while lower mCE is better on ImageNet-C.

Model	Params	FLOPs	ImageNet-A (top-1, %) ↑	ImageNet-C (mCE, %) ↓	ImageNet-R (top-1, %) ↑
Swin-B (Liu et al., 2021)	88.0 M	15.4 G	35.8	64.2	54.4
MLP-Fusion-Swin-B (Wei et al., 2023)	70.8 M	13.3 G	34.3	66.8	52.6
NTK-SAP-Swin-B (Wang et al., 2023)	72.6 M	13.2 G	34.6	66.1	52.9
LPViT-Swin-B (Xu et al., 2024)	70.8 M	13.5 G	35.1	65.4	53.3
STAR-Swin-B (Zhang et al., 2024)	70.2 M	13.4 G	36.0	64.2	54.5
DeepCompress-Swin-B (Ahmed et al., 2025)	71.5 M	13.0 G	34.0	67.3	52.4
KCR-Swin-B (Ours)	70.2 M	12.6 G	**37.9**	**62.2**	**56.1**

D.7 Sensitivity Analysis of Hyperparameters γ_0, η, m

We further study the sensitivity of KCR-Transformer to the key hyperparameters γ_0, η, and m by conducting experiments on the Swin-B backbone with different values of γ_0, η, and m. When performing the study on one of the hyperparameters, the remaining hyperparameters are set to their corresponding optimal values. The results summarized in Tables 12, 13, and 14. Across all settings, the top-1 accuracy remains stable within an error range of 0.3%, indicating that KCR-Transformer is robust to the choice of these hyperparameters.

Table 12: Sensitivity analysis of KCR-Swin-B to the choices of γ_0 on the ImageNet-1K dataset.

γ_0	0.1	0.2	0.3	0.4	0.5
Top-1 (%)	84.6	84.7	84.5	84.7	84.6

Table 13: Sensitivity analysis of KCR-Swin-B to the choices of η on the ImageNet-1K dataset.

η	0.1	0.2	0.3	0.4	0.5
Top-1 (%)	84.6	84.7	84.7	84.6	84.5

Table 14: Sensitivity analysis of KCR-Swin-B to the choices of m on the ImageNet-1K dataset.

m	25000	50000	75000	100000	200000
Top-1 (%)	84.4	84.7	84.6	84.7	84.7

Table 15: Performance comparison between KCR-Swin-B and models compressed by different model compression methods under different compression ratios. Increasing λ enforces stronger compression, leading to smaller parameter sizes and FLOPs.

Model	λ	# Params (M)	FLOPs (G)	Top-1 (%)
Swin-B	-	88.0	15.4	83.5
NTK-SAP-Swin-B (Wang et al., 2023)	-	72.6	13.2	83.2
MLP-Fusion-Swin-B (Wei et al., 2023)	-	70.8	13.3	83.2
LPViT-Swin-B (Xu et al., 2024)	-	70.8	13.5	83.3
STAR-Swin-B (Zhang et al., 2024)	-	70.2	13.4	83.6
DeepCompress-Swin-B (Ahmed et al., 2025)	-	71.5	13.0	83.1
KCR-Swin-B	0.2	70.2	12.6	**84.7**
NTK-SAP-Swin-B (Wang et al., 2023)	-	57.9	10.2	82.8
MLP-Fusion-Swin-B (Wei et al., 2023)	-	56.5	10.0	83.0
LPViT-Swin-B (Xu et al., 2024)	-	56.0	9.9	82.6
STAR-Swin-B (Zhang et al., 2024)	-	55.5	9.8	83.1
DeepCompress-Swin-B (Ahmed et al., 2025)	-	55.8	9.8	82.9
KCR-Swin-B	0.4	55.4	9.7	**84.2**
NTK-SAP-Swin-B (Wang et al., 2023)	-	49.3	8.4	82.2
MLP-Fusion-Swin-B (Wei et al., 2023)	-	47.9	7.9	82.1
LPViT-Swin-B (Xu et al., 2024)	-	48.9	8.4	82.2
STAR-Swin-B (Zhang et al., 2024)	-	47.5	8.0	82.3
DeepCompress-Swin-B (Ahmed et al., 2025)	-	48.5	8.2	82.4
KCR-Swin-B	0.6	46.7	7.8	**83.4**
NTK-SAP-Swin-B (Wang et al., 2023)	-	40.2	6.3	81.5
MLP-Fusion-Swin-B (Wei et al., 2023)	-	41.3	6.8	81.7
LPViT-Swin-B (Xu et al., 2024)	-	40.5	6.3	81.6
STAR-Swin-B (Zhang et al., 2024)	-	40.2	6.2	81.8
DeepCompress-Swin-B (Ahmed et al., 2025)	-	40.1	6.2	81.8
KCR-Swin-B	0.8	39.5	6.0	**82.9**

D.8 KCR-Transformers with Different Compression Ratios

In this section, we study the performance of KCR-Transformer under different compression ratios by varying the hyperparameter λ in the search loss and compare its performance with models compressed by competing model compression methods at different compression ratios. Larger values of λ impose stronger penalization on the computational cost, thereby yielding more aggressively compressed architectures. We evaluate four representative settings, $\lambda \in \{0.2, 0.4, 0.6, 0.8\}$, using the Swin-B backbone on the ImageNet-1K dataset. In addition, we compare KCR-Swin-B with Swin-B compressed by NTK-SAP (Wang et al., 2023) of similar sizes. As shown in Table 15, KCR-Transformer consistently outperforms models compressed by all competing methods across all compression levels. Although the performance of KCR-Swin-B degrades and falls below that of the original uncompressed model under aggressive compression, KCR-Swin-B still significantly outperforms models compressed by competing compression methods at similar compression levels. For example, when $\lambda = 0.8$, KCR-Swin-B outperforms DeepCompress-Swin-B by 1.1% in top-1 accuracy while using $0.6M$ fewer parameters and $0.2G$ fewer FLOPs.

D.9 Effect of the Channel-Selection Ratio and Effectiveness on Compact Backbones

In this section, we analyze how the channel-selection (compression) ratio affects the performance of KCR-Transformers, complementing the results in Section D.8, and we examine whether KCR remains effective when the backbone is already compact with small channel dimensions.

Effect of the channel-selection ratio. The compression level of a KCR-Transformer network, i.e., the size of the searched architecture, is controlled by the weight λ of the cost term in the search loss described in Section 3.3, while the channels retained in each layer are determined by the learned decision masks (see Section D.19). Our results show that a moderate compression ratio even improves the accuracy, owing to the theoretically grounded KCR-based channel selection, whereas aggressive compression degrades the accuracy, which is consistent with the broadly observed trade-

off in the model compression literature. As shown in Table 15, at a moderate ratio ($\lambda = 0.2$), KCR-Swin-B with 70.2M parameters outperforms the original uncompressed Swin-B (Liu et al., 2021) with 88.0M parameters (84.7% vs. 83.5%). As λ increases, the top-1 accuracy of KCR-Swin-B decreases and reaches 82.9% with 39.5M parameters at $\lambda = 0.8$. Even under such aggressive compression, KCR-Swin-B still exceeds DeepCompress-Swin-B (Ahmed et al., 2025) by 1.1% in top-1 accuracy at a similar model size, as discussed in Section D.8.

Effectiveness on compact backbones. To examine whether KCR remains effective for a vision transformer with small channel dimensions, we consider the already compact EfficientViT-B1 (Cai et al., 2023) with 9.1M parameters and 0.52G FLOPs. As shown in Table 1, KCR further reduces the model to 7.8M parameters and 0.44G FLOPs while improving its top-1 accuracy from 79.4% to 80.4%. It is observed that KCR-EfficientViT-B1 achieves the highest top-1 accuracy among the compared compression methods at this scale, including MLP-Fusion (Wei et al., 2023), NTK-SAP (Wang et al., 2023), LPViT (Xu et al., 2024), STAR (Zhang et al., 2024), and DeepCompress (Ahmed et al., 2025). In addition, Table 16 shows that KCR-EfficientViT-B1 has the lowest inference time (0.271 ms/img) and the highest throughput (3690 img/s) among all the EfficientViT-B1 variants. These results demonstrate that KCR remains effective for compact backbones with small channel dimensions.

Table 16: Comparisons with baseline methods on ImageNet-1K validation set (inference time measured on 2× NVIDIA A100 40GB GPU, FP16 precision, batch size = 128).

Model	# Params	FLOPs	Top-1 (%)	Inference Time (ms/img)	Throughput (img/s)
EfficientViT-B1 (Cai et al., 2023)	9.1 M	0.52 G	79.4	0.312	3205
MLP-Fusion-EfficientViT-B1 (Wei et al., 2023)	7.9 M	0.48 G	79.1	0.298	3356
NTK-SAP-EfficientViT-B1 (Ahmed et al., 2025)	8.0 M	0.49 G	79.4	0.304	3289
LPViT-EfficientViT-B1 (Xu et al., 2024)	7.9 M	0.46 G	79.1	0.289	3460
STAR-EfficientViT-B1 (Zhang et al., 2024)	8.0 M	0.48 G	79.3	0.301	3322
DeepCompress-EfficientViT-B1 (Ahmed et al., 2025)	7.9 M	0.46 G	79.2	0.285	3509
KCR-EfficientViT-B1	7.8 M	0.44 G	**80.4**	**0.271**	**3690**
ViT-S (Dosovitskiy et al., 2021)	22.1 M	4.3 G	81.2	0.642	1558
MLP-Fusion-ViT-S (Wei et al., 2023)	19.8 M	4.0 G	81.0	0.596	1678
NTK-SAP-ViT-S (Wang et al., 2023)	20.3 M	3.9 G	80.9	0.583	1715
LPViT-ViT-S (Xu et al., 2024)	19.8 M	3.8 G	81.0	0.568	1761
STAR-ViT-S (Zhang et al., 2024)	20.2 M	3.9 G	81.2	0.589	1698
DeepCompress-ViT-S (Ahmed et al., 2025)	20.0 M	3.9 G	81.1	0.574	1742
KCR-ViT-S (Ours)	19.8 M	3.8 G	**82.2**	**0.561**	**1783**
ViT-B (Dosovitskiy et al., 2021)	86.5 M	17.6 G	83.7	1.594	627
MLP-Fusion-ViT-B (Wei et al., 2023)	70.2 M	15.3 G	83.5	1.442	693
NTK-SAP-ViT-B (Wang et al., 2023)	71.8 M	15.6 G	83.5	1.467	681
LPViT-ViT-B (Xu et al., 2024)	69.8 M	15.1 G	83.4	1.432	698
STAR-ViT-B (Zhang et al., 2024)	69.5 M	14.7 G	83.5	1.401	714
DeepCompress-ViT-B (Ahmed et al., 2025)	70.5 M	15.1 G	83.6	1.426	702
KCR-ViT-B (Ours)	69.5 M	14.5 G	**84.6**	**1.378**	**726**
Swin-T (Liu et al., 2021)	29.0 M	4.5 G	81.3	0.673	1486
MLP-Fusion-Swin-T (Wei et al., 2023)	24.8 M	4.1 G	81.0	0.631	1585
NTK-SAP-Swin-T (Wang et al., 2023)	25.5 M	4.2 G	81.2	0.644	1553
LPViT-Swin-T (Xu et al., 2024)	24.8 M	4.0 G	81.1	0.615	1626
STAR-Swin-T (Zhang et al., 2024)	24.6 M	4.0 G	81.4	0.607	1647
DeepCompress-Swin-T (Ahmed et al., 2025)	24.8 M	4.1 G	81.1	0.622	1607
KCR-Swin-T (Ours)	24.6 M	3.9 G	**82.4**	**0.598**	**1672**
Swin-B (Liu et al., 2021)	88.0 M	15.4 G	83.5	1.522	657
MLP-Fusion-Swin-B (Wei et al., 2023)	70.8 M	13.3 G	83.2	1.384	722
NTK-SAP-Swin-B (Wang et al., 2023)	72.6 M	13.2 G	83.2	1.369	730
LPViT-Swin-B (Xu et al., 2024)	70.8 M	13.5 G	83.3	1.395	717
STAR-Swin-B (Zhang et al., 2024)	70.2 M	13.4 G	83.6	1.376	727
DeepCompress-Swin-B (Ahmed et al., 2025)	71.5 M	13.0 G	83.1	1.357	737
KCR-Swin-B (Ours)	70.2 M	12.6 G	**84.7**	**1.312**	**763**

D.10 Inference Time and Throughput Comparison

In this section, we further compare the inference latency of the KCR-Transformers against their corresponding baseline vision transformers without the KCR regularization. All measurements are conducted on two NVIDIA A100 40GB GPUs using FP16 precision with a batch size of 128. As shown in Table 16, KCR-Transformer variants consistently achieve lower inference latency com-

pared to their counterparts, while simultaneously improving top-1 classification accuracy and reducing the inference time, leading to higher throughput. For instance, KCR-Swin-B achieves a 1.2% gain in top-1 accuracy over Swin-B, while reducing inference time from 1.522 ms/image to 1.312 ms/image and improving the throughput by 106 images/second. In addition, KCR-Swin-B outperforms DeepCompress-Swin-B by 1.6% in top-1 accuracy, while reducing the inference time by 0.045 ms/image and improving the throughput by 26 images/second. These results underscore the practical deployment advantages of KCR regularization, which enhances accuracy and efficiency without sacrificing runtime performance.

D.11 System-Level Efficiency, Memory Footprint, and the KV Cache

In this section, we analyze the system-level efficiency of KCR-Transformers, including the relation between the MLP channel selection by KCR and the key-value (KV) cache in vision-language models (VLMs), the measured encoder-level speedup, and the memory footprint during inference.

On the KV cache. A persistent KV cache is used when autoregressive decoding reuses the keys and values of previous tokens. The vision transformers studied in this paper process image tokens in a non-causal and bidirectional manner and do not maintain such an autoregressive KV cache. In a VLM, the persistent KV cache is used by the LLM decoder rather than the vision encoder. Therefore, KCR applied to the vision encoder and decoder-side KV-cache compression operate on different modules and can be used together.

System-level effect in a VLM. For a vision encoder followed by a decoder-only LLM, KCR directly reduces the computation of the vision encoder and the dimension of the features received by the vision-to-language projector. Therefore, the projector can also be constructed with a smaller input dimension. These savings are complementary to decoder-side KV-cache compression.

Relation between the KV cache and MLP channel pruning. We consider a VLM comprising a vision encoder, a vision-to-language projector, and a decoder-only LLM. The vision transformer encodes an image into visual tokens. The projector converts their channels into the LLM input space, after which the LLM performs autoregressive decoding and stores the keys and values of previous visual-prefix and language tokens. The cache memory is proportional to $2LNd_{\mathrm{KV}}b$, where L is the number of decoder layers, N is the cached sequence length, d_{KV} is the key/value dimension per layer, and b is the number of bytes per element. KCR prunes the MLP channels in the vision encoder. When the encoder output contains fewer channels, the projector receives a smaller input dimension, which reduces its parameters, FLOPs, and activation memory. This makes the conversion to the LLM input space and the input preparation for decoding more efficient, thereby improving end-to-end VLM inference. However, after the projection into the fixed LLM hidden space, KCR applied only to the vision encoder does not change L, N, or d_{KV}. Therefore, it does not directly reduce the persistent decoder KV-cache allocation or the per-token decoder cost. Direct KV-cache savings require complementary methods such as KV quantization, eviction, or visual-token reduction. Thus, KCR on the vision encoder provides encoder- and projector-side savings that are complementary to decoder-side KV-cache compression and can be combined with these methods. The compression of the complete VLM by KCR, including the decoder FFN channels, the attention Q/K/V/O projections, and the KV dimensions of the decoder, is studied in Section D.13.

Measured encoder-level speedup and estimated inference memory. Table 17 summarizes the efficiency of ViT-B, Swin-B, and their KCR-compressed counterparts. The inference time and throughput are the same measurements as those in Table 16 of Section D.10, which are obtained on two NVIDIA A100 40GB GPUs with FP16 precision and a batch size of 128. It is observed that KCR reduces the inference latency and improves the throughput while achieving higher accuracy. For example, KCR improves the throughput from 627 to 726 images/second for ViT-B and from 657 to 763 images/second for Swin-B. For memory, KCR reduces the parameter memory at FP16 from 173.0 MB to 139.0 MB for ViT-B and from 176.0 MB to 140.4 MB for Swin-B, a reduction of about 20%. The last column of Table 17 reports the estimated per-GPU peak memory during inference at a batch size of 128. These peak-memory values are estimates rather than directly profiled measurements. KCR reduces the estimated peak memory from 7.6 GB to 6.6 GB for ViT-B and from 7.0 GB to 6.1 GB for Swin-B, a reduction of about 13%. This reduction is smaller than the FLOP reduction because the attention modules, which are not compressed by KCR, still account for a large share of the activation memory.

Table 17: Encoder-level inference efficiency and memory footprint of ViT-B (Dosovitskiy et al., 2021), Swin-B (Liu et al., 2021), and KCR-Transformers on ImageNet-1K (2× NVIDIA A100 40GB GPUs, FP16 precision, batch size = 128). The inference time and throughput are the same as those in Table 16. The parameter memory is computed at FP16 precision. The peak memory is the *estimated* per-GPU peak memory during inference, not a directly profiled measurement.

Model	# Params	FLOPs	Top-1 (%)	Inference Time (ms/img)	Throughput (img/s)	Param. Memory (MB)	Est. Peak Memory (GB)
ViT-B	86.5 M	17.6 G	83.7	1.594	627	173.0	7.6
KCR-ViT-B (Ours)	69.5 M	14.5 G	**84.6**	**1.378**	**726**	139.0	6.6
Swin-B	88.0 M	15.4 G	83.5	1.522	657	176.0	7.0
KCR-Swin-B (Ours)	70.2 M	12.6 G	**84.7**	**1.312**	**763**	140.4	6.1

D.12 Complex Multimodal Reasoning on ScienceQA

Beyond image classification, KCR-Transformers are evaluated on object detection on MS-COCO, semantic segmentation on ADE20K, and visual question answering on VQAv2 and SNLI-VE in Section 4, together with the robustness evaluation on ImageNet-A, ImageNet-C, and ImageNet-R, transfer learning, and self-supervised pretraining experiments in the appendix. In this section, we further evaluate KCR-Transformer on tasks requiring complex multimodal reasoning. To this end, we adopt ScienceQA (Lu et al., 2022), a multimodal multiple-choice science question-answering benchmark. We follow its official test split and report the accuracy on the image-context (IMG) subset. Specifically, we use KCR-ViT-B as the vision encoder in the VLM framework FLAVA (Singh et al., 2022) and follow the same fine-tuning protocol as the VQA experiments in Section 4.5. As in Section 4.5, we compare FLAVA with KCR-ViT-B against FLAVA with the original ViT-B and with NTK-SAP-ViT-B as the vision encoder.

Table 18: Accuracy (%) on the image-context (IMG) subset of the ScienceQA (Lu et al., 2022) test split. ViT-B (Dosovitskiy et al., 2021), NTK-SAP-ViT-B (Wang et al., 2023), and KCR-ViT-B are employed as the vision encoder in the VLM framework, FLAVA (Singh et al., 2022). The numbers of parameters and FLOPs are those of the vision encoders, the same as in Table 5.

Model (Vision Encoder in FLAVA)	# Params	FLOPs	ScienceQA (IMG) Acc. (%)
ViT-B	86.5 M	17.6 G	70.0
NTK-SAP-ViT-B	71.8 M	15.6 G	69.5
KCR-ViT-B (Ours)	69.5 M	14.5 G	**71.2**

It is observed in Table 18 that FLAVA with KCR-ViT-B achieves 71.2% accuracy, outperforming FLAVA with the original uncompressed ViT-B and FLAVA with NTK-SAP-ViT-B by 1.2% and 1.7%, respectively. These results are consistent with those on VQAv2 and SNLI-VE in Table 5, and show that KCR remains effective for complex multimodal reasoning while using fewer FLOPs and parameters than the original uncompressed vision encoder. The ScienceQA-IMG results of the complete VLM, LLaVA-1.5-7B, jointly compressed by KCR are further reported in Section D.13 under the official LLaVA-1.5 evaluation protocol, which differs from the FLAVA fine-tuning protocol used in this section.

D.13 Joint KCR Compression of a Complete Vision-Language Model

The KCR-Transformers in the main paper select the channels of the attention outputs and thereby prune the input and output channels of the MLP layers of vision transformers, while the query, key, and value projections of the attention modules are not compressed. Moreover, as discussed in Section D.11, the persistent KV cache of a vision-language model (VLM) is used by its language decoder rather than its vision encoder, so KCR applied only to the vision encoder does not directly reduce the decoder KV-cache allocation. In this section, we extend KCR to jointly compress a complete VLM, LLaVA-1.5-7B (Liu et al., 2024a), including its CLIP (Radford et al., 2021) vision encoder, vision-language projector, decoder feed-forward network (FFN) channels, and attention Q/K/V/O and KV dimensions. The resulting jointly compressed model is referred to as KCR-VLM. We describe this extension below and provide end-to-end system measurements. KCR-VLM reduces the model from 7.06B to 4.65B parameters, the FP16 weight memory by 34.3%, the allocated KV-cache memory by 35.9%, the peak GPU memory by 31.4%, and the end-to-end latency by 21.8%,

while improving the decoding throughput by 28.7%. At comparable model sizes, KCR-VLM also exceeds random and magnitude-based structured selection by 3.2 and 2.1 average benchmark points, respectively, providing evidence that its favorable accuracy-efficiency trade-off arises from KC-guided compression rather than redundancy alone. We also explicitly account for the uncompressed residual stream, embeddings, normalization layers, sampling, and runtime overhead, which explains why the system-level speedup is smaller than the channel reduction.

KCR compression of the LLaMA decoder. We apply KCR to both the decoder FFN channels and the attention/KV dimensions of the LLaMA decoder of LLaVA-1.5-7B. For each SwiGLU FFN, we jointly select the output channels of the gate and up projections and the corresponding input channels of the down projection. For attention, one structured mask selects complete head blocks in the Q/K/V projections and the corresponding input columns of the output projection. The shared mask preserves the query-key products and the value aggregation, while the complete-head selection preserves the rotary-position coordinate pairs. The residual-stream dimension, embeddings, normalization layers, and vocabulary projection remain unchanged. Therefore, KCR reduces the FFN computation and the KV-cache dimensions without changing the residual interface. Following the KCR-Transformer in Section 3.1, the Gumbel-Softmax masks and the network parameters are jointly optimized.

Training objective and hyperparameter selection. We replace the image-classification cross-entropy loss in (2) with the standard autoregressive next-token prediction (NTP) loss $\mathcal{L}_{\text{NTP}}$, averaged over the assistant-response tokens. The visual-prefix, system-prompt, user-prompt, and padding positions are masked. The search loss is

$$\mathcal{L}_{\text{search}} = \mathcal{L}_{\text{NTP}} + \eta\,\text{KCR} + \lambda \log C_{\text{FLOPs}}(g),$$

where η balances next-token prediction and KCR regularization, and λ controls the FLOPs penalty. We tune both η and λ following Section 4.1 and Section C and retain the selected η during retraining; the effect of varying λ on the size of the searched KCR-Transformers is studied in Section D.8. The value of λ controls the searched model size but does not impose a fixed channel-retention ratio; the retention of 63.9% reported below is produced by the learned masks. Similar to $\texttt{cost}(\alpha)$ in the search loss of Section 3.3, the cost $C_{\text{FLOPs}}(g)$ is determined by the decision masks g and covers the selected channels in the vision encoder, projector, decoder FFNs, and Q/K/V/O projections. It therefore encourages K/V reduction without a separate KV-memory regularizer.

Applicability of KCR to the cross-entropy-based NTP loss. As shown in Section A.1, under the standard bounded-score condition, Theorem 3.1 still holds with the squared loss replaced by the cross-entropy (CE) loss, with only constant-factor changes. Thus, the training loss (2) directly minimizes the empirical CE loss and the KCR regularization term appearing in the resulting generalization bound with the CE loss. This justifies the applicability of KCR to the training of an LLM with the CE-based NTP loss. Since the bound is established for independent classification examples, the scope of the theoretical claim for next-token prediction is stated at the end of this section.

Computation of the KCR kernel. We compute one dynamic kernel from the output of the final RMSNorm layer of the decoder, immediately before the vocabulary projection. For the i-th input sequence, we average the final hidden states at the supervised positions:

$$\mathbf{f}_i = |\mathcal{M}_i|^{-1} \sum_{t \in \mathcal{M}_i} \mathbf{h}_{i,t}^{(L)},$$

where i indexes the input sequence, $\mathbf{f}_i$ is its sequence-level representation, $\mathcal{M}_i$ is the set of supervised token positions in the i-th sequence, $|\mathcal{M}_i|$ is the number of such positions, t indexes a token position in $\mathcal{M}_i$, $\mathbf{h}_{i,t}^{(L)}$ is the hidden state at position t of the i-th sequence after the final decoder layer, and L is the number of decoder layers. These positions correspond to caption prediction during feature alignment and assistant-response prediction during instruction tuning. The visual-prefix and prompt states are not separate kernel samples, but they influence $\mathbf{f}_i$ through causal conditioning. We stack the sequence-level representations into $\mathbf{F}$, form the gram matrix $\mathbf{K} = \mathbf{F}\mathbf{F}^\top$, and compute KCR using the Nyström-based approximate truncated nuclear norm $\overline{\|\mathbf{K}\|_r}$ in Section 3.2. This global kernel changes with the model and the masks, and its gradients guide the masks of the vision encoder, projector, decoder FFNs, and attention/KV dimensions.

Training data and protocol. Unlike the ImageNet experiments described in Section C, KCR-VLM follows the training recipe of LLaVA-1.5. Specifically, we initialize KCR-VLM from LLaVA-1.5-7B and follow its official two-stage data and training recipe (Liu et al., 2024a). The architecture

Table 19: System-level inference efficiency of the dense LLaVA-1.5-7B (Liu et al., 2024a) and the jointly compressed KCR-VLM (one NVIDIA A100-80GB GPU, FP16 precision, batch size $= 1$, total cached length of 2048 tokens, greedy decoding of 128 new tokens). KCR-VLM jointly compresses the vision encoder, projector, decoder FFNs, and KV dimensions. Weights: FP16 weight memory; KV Cache: allocated KV-cache memory; Peak Mem.: peak GPU memory; TTFT: time to first token; Decode: per-token decoding latency; Throughput: decoding throughput; Latency: end-to-end latency.

Model	Params (B)	Weights (GiB)	KV Cache (GiB)	Peak Mem. (GiB)	TTFT (ms)	Decode (ms/tok)	Throughput (tok/s)	Latency (s)
LLaVA-1.5-7B	7.06	13.16	1.03	15.53	177.6	10.8	92.7	1.56
KCR-VLM (Ours)	4.65	**8.65**	**0.66**	**10.66**	**142.1**	**8.4**	**119.3**	**1.22**

Table 20: Multimodal performance of the dense LLaVA-1.5-7B (Liu et al., 2024a) and the structured-compression methods, including random structured selection (Li et al., 2022b), magnitude-based structured selection (Li et al., 2017), and KCR-VLM. KCR-VLM is the same jointly compressed model evaluated in Table 19. The average is computed over VQAv2, GQA, ScienceQA-IMG, TextVQA, and MMBench; all compressed models use the same data and retraining schedule. Bold numbers indicate the best results among the compressed models.

Model	Size (B)	VQAv2 (%)	GQA (%)	ScienceQA-IMG (%)	TextVQA (%)	MMBench (%)	Average (%)
LLaVA-1.5-7B	7.06	78.5	62.0	66.8	58.2	64.3	66.0
Random Structured Selection	4.83	76.3	59.0	61.8	53.9	59.2	62.0
Magnitude-Based Structured Selection	4.75	76.9	59.8	63.1	55.0	60.7	63.1
KCR-VLM (Ours)	4.65	**78.0**	**61.5**	**65.7**	**57.3**	**63.5**	**65.2**

search uses a stratified 100K subset of LLaVA-mix-665K (Liu et al., 2024a), where 70% of the examples are used to update the network weights and the remaining 30% are used to update the architecture parameters. After the search, we gather the selected channels into compact dense matrices and retrain the searched network with $\mathcal{L}_{\text{NTP}} + \eta\,\text{KCR}$. LLaVA-mix-665K is also used by DeRS (Huang et al., 2025) for efficient VLM training.

Model size and KV-cache reduction. The architecture search retains 63.9% of the selected FFN and attention dimensions, reducing the complete VLM from 7.06B to 4.65B parameters. KCR-VLM therefore retains 65.8% of the original parameters; the embeddings, normalization layers, LM head, and other unpruned components remain unchanged. Since the K/V dimensions are physically reduced, the KV-cache allocation is also directly lowered.

System-level inference efficiency. We perform inference on one NVIDIA A100-80GB GPU using FP16 precision with a batch size of 1. Each example contains one 336×336 image producing 576 visual tokens, the total cached length is 2048 tokens, and 128 new tokens are generated by greedy decoding. We use FlashAttention-2 (Dao, 2024) and compact hardware-aligned dense kernels, without quantization or CPU offloading. The latency is measured after warm-up and reported as the median over 100 examples. As shown in Table 19, KCR-VLM reduces the FP16 weight memory by 34.3%, the KV-cache allocation by 35.9%, and the peak memory by 31.4%, while increasing the decoding throughput by 28.7% and reducing the end-to-end latency by 21.8%. The speedup is smaller than the channel reduction because the residual stream, embeddings, normalization, sampling, and runtime overhead remain unchanged. In addition, the compact dense matrices avoid irregular gather/scatter operations.

Multimodal performance and compression baselines. Following the official LLaVA-1.5 evaluation protocol (Liu et al., 2024a), we report the performance on VQAv2 (Goyal et al., 2017), GQA (Hudson & Manning, 2019), ScienceQA-IMG (Lu et al., 2022), TextVQA (Singh et al., 2019), and MMBench (Liu et al., 2024b). Similar to the matched-sparsity comparison on image classification in Section D.18, we compare KCR-VLM with two structured selection baselines. Random structured selection (Li et al., 2022b) uniformly samples hardware-aligned FFN and attention/KV blocks, whereas magnitude-based structured selection (Li et al., 2017) retains the blocks with the largest associated ℓ_1 weight norms. Both baselines use the next-token CE loss without KCR, the same data and retraining schedule as KCR-VLM, and the nearest hardware-aligned model size not smaller than that of KCR-VLM.



Figure 2: Grad-CAM visualization across four ImageNet classes, including wallaby, pug, moped, and thatch. The first row illustrates the original images. The second, third, and fourth row illustrates the Grad-CAM heatmaps from ViT-B (Dosovitskiy et al., 2021), NTK-SAP-ViT-B (Wang et al., 2023), and KCR-ViT-B, respectively.

As shown in Table 20, KCR-VLM achieves an average performance of 65.2, exceeding random and magnitude-based structured selection by 3.2 and 2.1 points despite using 0.18B and 0.10B fewer parameters, respectively. It remains comparable to the dense LLaVA-1.5-7B (65.2 vs. 66.0) while reducing the peak memory by 31.4% and increasing the decoding throughput by 28.7% (Table 19). These results demonstrate a favorable accuracy-efficiency trade-off without a uniform channel-selection ratio.

D.14 Visualization Results

To qualitatively assess the discriminative capacity and spatial focus of different transformer variants, we apply the Grad-CAM technique (Selvaraju et al., 2017) to visualize the class-specific activation regions in the input images. We compute the Grad-CAM heatmaps for images from four representative ImageNet classes, including wallaby, pug, moped, and thatch, comparing the baseline ViT-B, NTK-SAP-ViT-B, and our KCR-ViT-B. Figure 2 illustrates that the proposed KCR-ViT-B consistently generates activation maps that are more spatially focused on the salient regions of the target object, while suppressing irrelevant background signals. In contrast, the attention maps of the ViT-B and NTK-SAP-ViT-B tend to exhibit higher activation in non-discriminative areas, such as sky, grass, or surrounding clutter, which may introduce unnecessary noise into the prediction process. The above observation suggests that the proposed KC regularization not only facilitates compression and computational efficiency but also enhances the model's representation learning capability for the classification task by guiding it to attend more selectively to task-relevant features.

D.15 Study on the Impact of KC

To assess the influence of KC on the performance of the KCR-Transformer, we perform an ablation study in which models have identical FLOPs and parameter sizes but different KC values. We set $\lambda = 0.2$ in this ablation study as that in the main paper. By varying the balancing weight η of the KCR regularization term in the training loss, the resulting models achieve different KC levels while maintaining matched model sizes. It is observed in Table 21 that reducing KC up to a certain range improves classification accuracy, whereas overly aggressive KC reduction eventually leads to a slight degradation in performance. In particular, for Swin-B at a compression ratio of 79.8% with $\lambda = 0.2$, increasing η from 0.01 to 0.2 reduces the KC from 0.30 to 0.16 and improves the top-1 accuracy from 83.9% to 84.7%. For ViT-B at a compression ratio of 80.3% with $\lambda = 0.2$, increasing η from 0.01 to 1.0 reduces the KC from 0.33 to 0.16 and improves the top-1 accuracy from 83.8% to 84.6%. Table 21 further shows that, for both Swin-B and ViT-B, the value of η that achieves the highest top-1 accuracy also yields the minimum KCR upper bound, which are 0.69 for Swin-B and 0.58 for ViT-B, thereby supporting the effectiveness of the KCR regularization term. Also, such best η is the same value selected by cross-validation, as described in Section 4.1 of the main paper.

Table 21: Impact of KC on the Top-1 Accuracy. The first column denotes the baseline uncompressed model. Training Loss denotes the empirical MSE loss. The models reported in this table are obtained by applying KCR to the original Swin-B and ViT-B architectures in the same manner as in Table 1, reducing Swin-B from 88.0M parameters and 15.4G FLOPs to 70.2M parameters and 12.6G FLOPs, respectively, and reducing ViT-B from 86.5M parameters and 17.6G FLOPs to 69.5M parameters and 14.5G FLOPs, respectively. The value of KC is scaled with a constant multiplier of $6K_0$ as in the proof of Theorem 3.1, and K_0 is set to the suggested value in the proof of Theorem 3.1 for the scaling factor of the training loss.

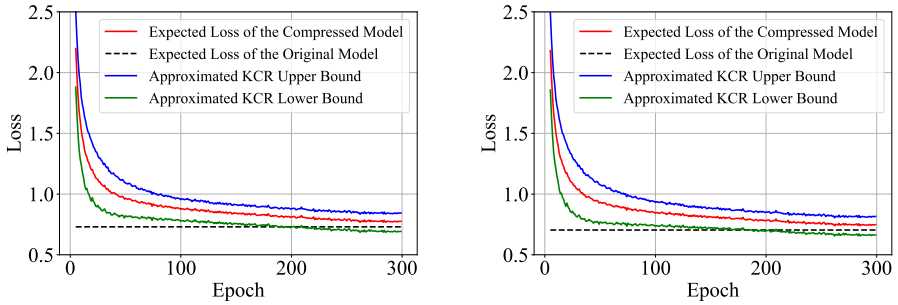
Base Model	η	–	0.01	0.05	0.2	0.4	0.6	0.8	1.0	1.5	2.0	2.5
Swin-B	KC	0.46	0.30	0.26	0.16	0.16	0.15	0.14	0.14	0.14	0.12	0.12
	Top-1 (%)	83.5	83.9	84.3	**84.7**	84.6	84.5	84.4	84.5	84.5	84.4	84.3
	Training Loss	0.62	0.56	0.42	**0.35**	0.36	0.37	0.40	0.39	0.37	0.40	0.44
	KCR Upper Bound	1.39	1.14	0.89	**0.69**	0.70	0.71	0.74	0.73	0.70	0.72	0.78
	KCR Lower Bound	0.01	0.12	0.06	0.10	0.11	0.13	0.16	0.15	0.14	0.18	0.21
ViT-B	KC	0.42	0.33	0.30	0.27	0.23	0.19	0.17	0.16	0.15	0.15	0.14
	Top-1 (%)	83.7	83.8	84.0	84.2	84.4	84.5	84.5	**84.6**	84.5	84.4	84.5
	Training Loss	0.58	0.50	0.48	0.45	0.45	0.37	0.31	**0.28**	0.31	0.34	0.32
	KCR Upper Bound	1.29	1.08	1.02	0.95	0.91	0.75	0.64	**0.58**	0.62	0.66	0.62
	KCR Lower Bound	0.02	0.05	0.06	0.07	0.11	0.09	0.06	0.05	0.08	0.11	0.10

D.16 Additional Results on the KCR Upper and Lower Bounds

Complementing Figure 1 in Section 4.6, Figure 3 illustrates the expected loss computed over the training and validation sets of the ImageNet-1K and the approximated KCR upper/lower bounds computed at different training epochs for Swin-T and ViT-S trained on the ImageNet-1K dataset. Similar to Swin-B and ViT-B, the approximated KCR upper/lower bounds are tightly correlated to the expected loss, revealing the tightness of the upper/lower bounds for the generalization error of the DNNs with KCR-Transformer blocks.

D.17 Study on the Effectiveness and Efficiency of the KCR Regularization

To study the effect of the two-stage training procedure independently from the contribution of the KCR regularization, we perform an ablation study in which the KCR regularization is disabled by setting $\eta = 0$ while keeping the same two-stage training pipeline for channel selection. This baseline isolates the impact of the KCR term itself. It is observed in Table 22 that the two-stage training alone does not lead to an improvement in model performance. In contrast, incorporating the KCR regularization yields consistent performance improvements across different architectures, demonstrating that the improvement primarily stems from it, which is aligned with the theoretical analysis in Theorem 3.1. In particular, reducing the KC through the KCR regularization effectively



(a) Swin-T (b) ViT-S

Figure 3: Illustration of the expected loss and the approximated KCR upper/lower bounds over different training epochs for Swin-T and ViT-S compressed by KCR.

lowers the upper bound for the generalization error, leading to better generalization capability and classification accuracy.

Furthermore, we compare the training efficiency of KCR-Transformers and their ablation variants without KCR regularization in Table 22. Here, "KCR-ViT-B without KCR regularization" denotes an ablation model that uses the same searched KCR-Transformer architecture and channel-pruned ViT-B backbone as KCR-ViT-B, but is retrained using only the standard cross-entropy loss by setting the KCR regularization weight to $\eta = 0$. Therefore, this ablation isolates the effect of the proposed KCR regularization from the effect of the searched compact architecture. It is observed that KCR regularization introduces only marginal training overhead, benefiting from the Nyström approximation of the TNN. For example, KCR-ViT-B requires only 17.2% more training time than its ablation counterpart, KCR-ViT-B without KCR regularization, while improving the top-1 accuracy by 1.5%.

Table 22: Study on the effectiveness and efficiency of the KCR regularization with ViT-B and Swin-B as the base models. "KCR-ViT-B without KCR regularization" denotes an ablation model that uses the same searched KCR-Transformer architecture and channel-pruned ViT-B backbone as KCR-ViT-B, but is retrained using only the standard cross-entropy loss by setting the KCR regularization weight to $\eta = 0$.

Models	Params	FLOPs	Training Efficiency (Minutes/Epoch)	Top-1 (%)
ViT-B	86.5 M	17.6 G	33.5	83.7
KCR-ViT-B without KCR Regularization	69.5 M	14.5 G	29.6	83.1
KCR-ViT-B	69.5 M	14.5 G	34.7	**84.6**
Swin-B	88.0 M	15.4 G	38.3	83.5
KCR-Swin-B without KCR Regularization	70.2 M	12.6 G	34.3	83.0
KCR-Swin-B	70.2 M	12.6 G	39.4	**84.7**

D.18 Comparison with Random and Magnitude-Based Channel Selection at Matched Sparsity

In this section, we study whether the accuracy retained by KCR-Transformers stems from the proposed KC-guided mechanism or from parameter redundancy in the original uncompressed models. We first revisit the ablation in Section D.17, and then compare KCR with random channel selection and magnitude-based channel selection at matched sparsity.

KCR-Transformers without KCR regularization. As described in Section D.17, "KCR-ViT-B without KCR regularization" and "KCR-Swin-B without KCR regularization" use the same searched KCR-Transformer architectures and channel-pruned backbones as KCR-ViT-B and KCR-Swin-B, but are retrained using only the standard cross-entropy loss by setting the KCR regularization weight to $\eta = 0$. It is observed in Table 22 that, without the KCR term, the compressed models perform worse than the original uncompressed baselines (83.1% vs. 83.7% for ViT-B, and 83.0% vs. 83.5% for Swin-B). Thus, parameter redundancy and the two-stage training procedure alone do not pre-

serve accuracy. With KCR regularization, the compressed models achieve 84.6% and 84.7% top-1 accuracy, respectively, exceeding their original uncompressed baselines. These results show that the improvement mainly stems from the proposed KC-guided regularization, which is consistent with Theorem 3.1.

Random and magnitude-based channel selection. To further separate the effect of KCR from parameter redundancy, we compare KCR with two channel selection baselines at matched sparsity. Random selection uniformly samples the retained channels, following the random channel pruning baseline in (Li et al., 2022b). Magnitude-based selection ranks the output channels according to the ℓ_1 norm of their weight vectors and retains the channels with the largest magnitudes, following (Li et al., 2017). For a controlled comparison, random selection and magnitude-based selection retain the same number of channels in each layer as KCR, so that all the compressed models have the same numbers of parameters and FLOPs. All the compressed models also follow the same retraining protocol, where the random and magnitude-based baselines are retrained with the standard cross-entropy (CE) loss, and KCR is retrained with the training loss in Eq. (2).

Table 23: Top-1 accuracy (%) on ImageNet-1K of ViT-B and Swin-B compressed by different channel selection methods at matched sparsity. For each backbone, random selection (Li et al., 2022b) and magnitude-based selection (Li et al., 2017) retain the same number of channels in each layer as KCR, and all the compressed models have the same numbers of parameters and FLOPs as KCR-ViT-B (69.5M parameters and 14.5G FLOPs) or KCR-Swin-B (70.2M parameters and 12.6G FLOPs). "KCR-Transformer without KCR regularization" denotes the ablation models KCR-ViT-B and KCR-Swin-B without KCR regularization in Table 22.

Selection Method (Matched Sparsity)	ViT-B Top-1 (%)	Swin-B Top-1 (%)
Random selection + CE retraining	82.1	81.7
Magnitude-based selection + CE retraining	82.4	82.2
KCR-Transformer without KCR regularization ($\eta = 0$)	83.1	83.0
KCR (Ours)	**84.6**	**84.7**
Original uncompressed baseline	83.7	83.5

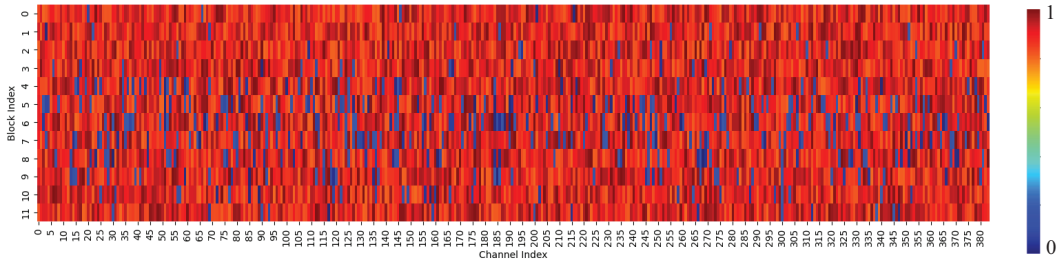
As shown in Table 23, random selection and magnitude-based selection perform worse than the original uncompressed baselines at the same sparsity, achieving 82.1% and 82.4% top-1 accuracy on ViT-B and 81.7% and 82.2% on Swin-B, respectively. In contrast, KCR achieves 84.6% and 84.7% top-1 accuracy on ViT-B and Swin-B, respectively, outperforming both the original uncompressed baselines and all the other compressed models with the same numbers of parameters and FLOPs. This comparison, together with the ablation without KCR regularization, demonstrates that neither parameter redundancy nor the two-stage retraining alone can explain the gains, and that the improvement comes from the proposed KC-guided mechanism.

D.19 Visualization of Channel Selection Masks

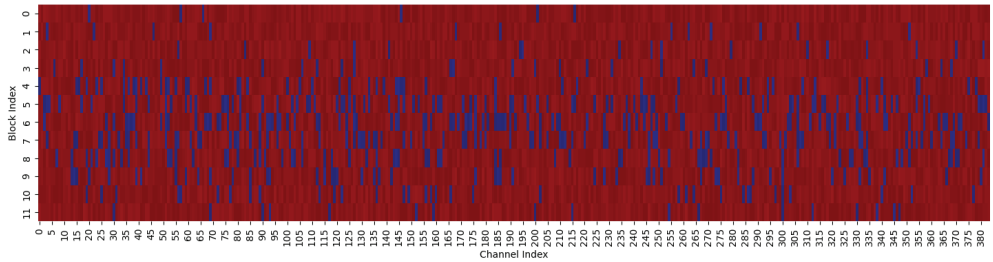
To further improve the interpretability of the channel selection mechanism by the Gumbel-Softmax operation, we illustrate the learned channel-selection masks for KCR-ViT-S and KCR-ViT-B in Figure 4. Each row in the heatmap corresponds to a layer, and each column corresponds to a channel of that layer. As the Gumbel-Softmax temperature decreases in the search phase, the learned masks evolve toward near-binary channel-selection masks, with most probabilities pushed close to either 0 or 1 at the end of the search phase. Furthermore, we observe that the shallow layers retain a larger proportion of channels, whereas the middle layers undergo a more aggressive reduction in the number of channels preserved.

D.20 t-SNE Visualization Analysis of the Image Representations Learned by KCR-Transformers

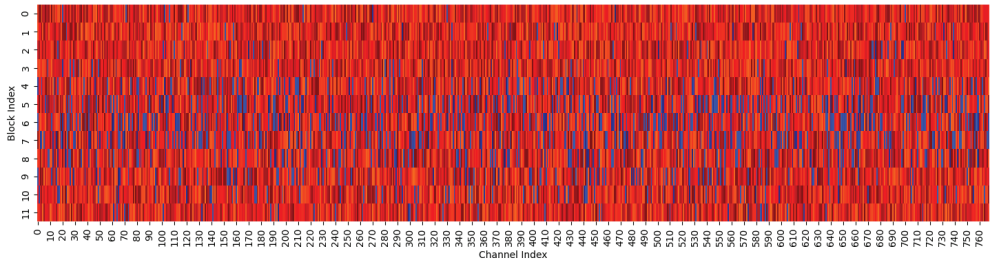
To further investigate the quality of the representations learned by KCR-Transformers, we visualize the learned image representations using t-SNE. Specifically, we randomly sample 10 ImageNet classes and then randomly sample 200 images from each selected class, and project the learned representations into a two-dimensional space. As illustrated in Figure 5, the representa-



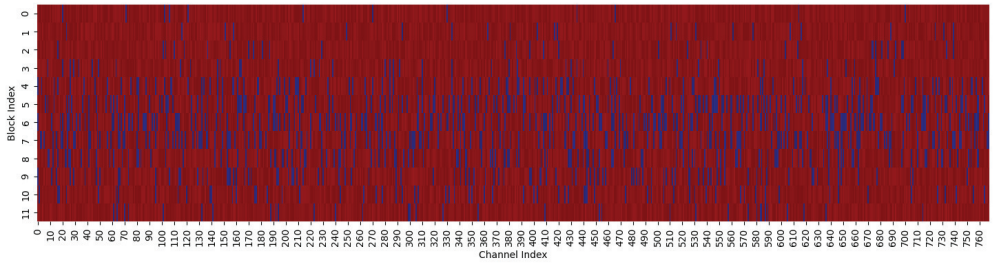
(a) KCR-ViT-S Channel Selection Masks (100 Epochs)



(b) KCR-ViT-S Channel Selection Masks (200 Epochs)



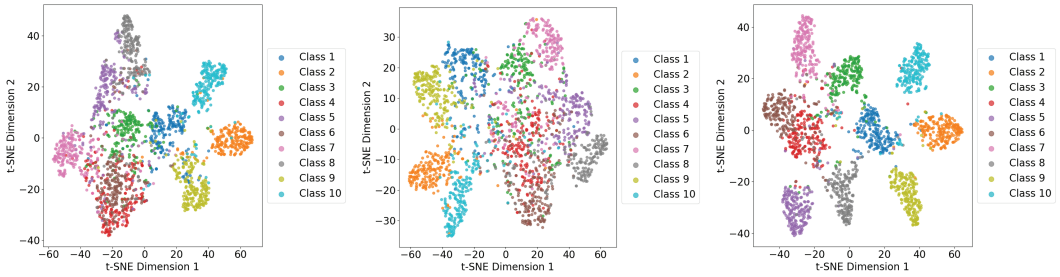
(c) KCR-ViT-B Channel Selection Masks (100 Epochs)



(d) KCR-ViT-B Channel Selection Masks (200 Epochs)

Figure 4: Visualization of learned Gumbel–Softmax channel-selection masks for KCR-ViT-S and KCR-ViT-B. Figures (a) and (b) illustrate the channel selection masks for ViT-S after 100 and 200 epochs in the search phase for KCR-ViT-S. Figures (c) and (d) illustrate the channel selection masks for ViT-B after 100 and 200 epochs in the search phase for KCR-ViT-B. The entire search phase takes 200 epochs.

tions learned by KCR-ViT-B exhibit clearer class separation than those learned by NTK-SAP-ViT-B and DeepCompress-ViT-B. In particular, KCR-ViT-B forms more compact intra-class clusters and larger margins between different classes, while NTK-SAP-ViT-B and DeepCompress-ViT-B show more overlap among classes and less coherent cluster structures. Such visualization results provide qualitative evidence that the proposed KCR regularization improves the discriminative quality of the learned feature space, benefiting from the improved generalization induced by the proposed KCR regularization.



(a) NTK-SAP-ViT-B (b) DeepCompress-ViT-B (c) KCR-ViT-B

Figure 5: t-SNE visualization of image representations learned by NTK-SAP-ViT-B, DeepCompress-ViT-B, and KCR-ViT-B. For each model, 10 ImageNet classes are randomly sampled, and 200 images are randomly sampled from each selected class. Compared with NTK-SAP-ViT-B and DeepCompress-ViT-B, KCR-ViT-B produces more compact intra-class clusters and better-separated inter-class distributions, indicating that KCR regularization leads to more discriminative image representations.

Table 24: Comparison of the KC between KCR-Transformers and models compressed by the baselines using EfficientViT-B1, Swin-T, and Swin-B. The value of KC is scaled with a constant multiplier of $6K_0$ as in the proof of Theorem 3.1, and K_0 is set to the suggested value in the proof of Theorem 3.1.

Model	# Params	FLOPs	Top-1 (%)	KC
EfficientViT-B1 (Cai et al., 2023)	9.1 M	0.52 G	79.4	0.53
MLP-Fusion-EfficientViT-B1 (Wei et al., 2023)	7.9 M	0.48 G	79.1	0.48
NTK-SAP-EfficientViT-B1 (Wang et al., 2023)	8.0 M	0.49 G	79.4	0.48
LPViT-EfficientViT-B1 (Xu et al., 2024)	7.9 M	0.46 G	79.1	0.37
STAR-EfficientViT-B1 (Zhang et al., 2024)	8.0 M	0.48 G	79.3	0.41
DeepCompress-EfficientViT-B1 (Cai et al., 2023)	7.9 M	0.46 G	79.2	0.40
KCR-EfficientViT-B1 (Cai et al., 2023)	7.8 M	0.44 G	**80.4**	**0.19**
Swin-T (Liu et al., 2021)	29.0 M	4.5 G	81.3	0.52
MLP-Fusion-Swin-T (Wei et al., 2023)	24.8 M	4.1 G	81.0	0.45
NTK-SAP-Swin-T (Wang et al., 2023)	25.5 M	4.2 G	81.2	0.28
LPViT-Swin-T (Xu et al., 2024)	24.8 M	4.0 G	81.1	0.41
STAR-Swin-T (Zhang et al., 2024)	24.6 M	4.0 G	81.4	0.38
DeepCompress-Swin-T (Ahmed et al., 2025)	24.8 M	4.1 G	81.1	0.40
KCR-Swin-T (Ours)	24.6 M	3.9 G	**82.4**	**0.13**
Swin-B (Liu et al., 2021)	88.0 M	15.4 G	83.5	0.46
MLP-Fusion-Swin-B (Wei et al., 2023)	70.8 M	13.3 G	83.2	0.38
NTK-SAP-Swin-B (Wang et al., 2023)	72.6 M	13.2 G	83.2	0.37
LPViT-Swin-B (Xu et al., 2024)	70.8 M	13.5 G	83.3	0.43
STAR-Swin-B (Zhang et al., 2024)	70.2 M	13.4 G	83.6	0.44
DeepCompress-Swin-B (Ahmed et al., 2025)	71.5 M	13.0 G	83.1	0.44
KCR-Swin-B (Ours)	70.2 M	12.6 G	**84.7**	**0.16**

D.21 Additional Results for KC Comparison

We further study the effectiveness of KCR-Transformer in reducing the KC across three representative vision transformer architectures, including EfficientViT-B1, Swin-T, and Swin-B. As shown in Table 24, the KCR-enhanced models consistently achieve the lowest KC while maintaining competitive or lower parameter counts, reduced FLOPs, and higher top-1 classification accuracy on

ImageNet-1K. For instance, KCR-Swin-T reduces the KC from 0.52 to 0.13 and improves the top-1 accuracy from 81.3% to 82.4%, while also reducing the FLOPs from 4.5G to 3.9G. Similarly, KCR-EfficientViT-B1 reduces the KC from 0.53 to 0.19 and improves the accuracy from 79.4% to 80.4%, and KCR-Swin-B reduces the KC from 0.46 to 0.16 with a top-1 accuracy improvement from 83.5% to 84.7%. These results demonstrate that our approximate TNN regularization effectively reduces the KC of vision backbones, which leads to improved generalization capability. In contrast, existing compression methods, including MLP-Fusion (Wei et al., 2023), NTK-SAP (Wang et al., 2023; Ahmed et al., 2025), LPViT (Xu et al., 2024), STAR (Zhang et al., 2024), and DeepCompress (Ahmed et al., 2025), generally produce models with larger KC and lower top-1 accuracy than KCR-Transformers under comparable model sizes and FLOPs.

D.22 Statistical Significance on ImageNet-1K

To further assess whether the improvements of KCR-Transformers over the strongest competing methods are statistically significant, we conduct a one-sided Student's t-test between each KCR model and the best non-KCR baseline under the same base architecture in Table 1. For each base model, the best baseline is selected according to the highest top-1 accuracy among the corresponding original and compressed models. We report the mean and standard deviation over 5 independent runs for both KCR and the selected best baseline. As shown in Table 25, KCR achieves statistically significant improvements across all base models, with all p-values below 0.01, which indicate that the gains of KCR-Transformers over the strongest baselines are consistent and unlikely to be caused by random variation.

Table 25: One-sided Student's t-test between KCR-Transformers and the best non-KCR baseline for each base model on ImageNet-1K. We report the mean and standard deviation of top-1 accuracy over 5 independent runs.

Base Model	Best Baseline	Baseline Top-1 (%)	KCR Top-1 (%)	p-value
EfficientViT-B1	NTK-SAP-EfficientViT-B1	79.4 ± 0.36	80.4 ± 0.31	0.0013
ViT-S	STAR-ViT-S	81.2 ± 0.42	82.2 ± 0.35	0.0021
ViT-B	ViT-B	83.7 ± 0.38	84.6 ± 0.34	0.0026
Swin-T	STAR-Swin-T	81.4 ± 0.40	82.4 ± 0.36	0.0019
Swin-B	STAR-Swin-B	83.6 ± 0.44	84.7 ± 0.37	0.0015