UPop: Unified and Progressive Pruning for Compressing Vision-Language Transformers

Dachuan Shi [1 2] **Chaofan Tao** [3] **Ying Jin** [4] **Zhendong Yang** [1] **Chun Yuan** [1 †] **Jiaqi Wang** [2 †]

Abstract

Real-world data contains a vast amount of multimodal information, among which vision and language are the two most representative modalities. Moreover, increasingly heavier models, *e.g.*, Transformers, have attracted the attention of researchers to model compression. However, how to compress multimodal models, especially vison-language Transformers, is still under-explored. This paper proposes the **U**nified and **Pro**gressive **P**runing (***UPop***) as a universal vison-language Transformer compression framework, which incorporates 1) unifiedly searching multimodal subnets in a continuous optimization space from the original model, which enables automatic assignment of pruning ratios among compressible modalities and structures; 2) progressively searching and retraining the subnet, which maintains convergence between the search and retrain to attain higher compression ratios. Experiments on various tasks, datasets, and model architectures demonstrate the effectiveness and versatility of the proposed UPop framework. The code is available at https://github.com/sdc17/UPop.

1. Introduction

The number of parameters and FLOPs of deep learning models (Devlin et al., 2018; Shoeybi et al., 2019; Brown et al., 2020; Shao et al., 2021; Smith et al., 2022) have proliferated in recent years, which makes compression exceedingly critical for deploying the increasingly heavier models on edge devices. There are lots of approaches for model compression, such as weight sharing (Lan et al., 2019), low-rank factorization (Yu et al., 2017), quantization (Tao et al., 2022), parameter bootstrapping (Chen et al., 2022), knowledge distillation (Yang et al., 2022), and pruning (Han et al., 2015b). As the paradigm this paper focuses on, pruning approaches not only benefit from inheriting well-optimized parameters of the original model but also provide flexible design space for various architectures.

Table 1: Overview of experimental results at 2× compression. The proposed UPop framework is efficient and effective on various tasks, datasets, and architectures. **Bold** indicates the best post-compression performance. Mask-based Pruning is extended from the SOTA pruning method ViT-Slimming (Chavan et al., 2022).

Method	Visual Reason	Image Caption	Visual QA	Retrieval COCO	Retrieval Flickr	Image Classification
Original Model	83.1	23.8	77.5	81.9	96.8	79.9
Mask-based Pruning	76.4$_{\downarrow 6.7}$	21.0$_{\downarrow 2.8}$	71.6$_{\downarrow 5.9}$	61.7$_{\downarrow 20}$	78.9$_{\downarrow 18}$	77.9$_{\downarrow 2.0}$
UPop (Ours)	**81.1**$_{\downarrow 2.0}$	**23.3**$_{\downarrow 0.5}$	**76.3**$_{\downarrow 1.2}$	**77.4**$_{\downarrow 4.5}$	**94.0**$_{\downarrow 2.8}$	**78.9**$_{\downarrow 1.0}$

Recently, pruning approaches dedicated to the Transformers (Vaswani et al., 2017) have attracted much attention. According to the pruned components, these approaches can be summarized into two categories. 1) Token Pruning: By eliminating the number of input tokens, these approaches (Goyal et al., 2020; Rao et al., 2021) can reduce the FLOPs of models. 2) Model Pruning. By reducing the model size, these approaches (Chen et al., 2021b; Su et al., 2022) can reduce both the parameters and FLOPs of models. This paper focuses on model compression so that the parameters and FLOPs of models can be reduced simultaneously.

In real applications, there are lots of multimodal tasks that have been extensively studied, including but not limited to Visual Question Answer (Antol et al., 2015), Image Caption (Lin et al., 2014), and Image-Text Retrieval (Jia et al., 2015). To tackle these multimodal tasks, various multimodal models (Kiros et al., 2014; Karpathy et al., 2014; Antol et al., 2015; Vinyals et al., 2015; Yang et al., 2016; Huang et al., 2017) have been proposed accordingly. Furthermore, as Transformer (Vaswani et al., 2017) has been more and more popular among deep models, transformer-based models (Tan & Bansal, 2019; Lu et al., 2019; Zhou et al., 2020; Li et al., 2020; Kim et al., 2021; Jia et al., 2021; Yu et al., 2022; Wang et al., 2022a) have also dominated the recent studies of multimodal models. For example, CLIP (Radford et al., 2021) and BLIP (Li et al., 2022) are some of the most repre-

[1]Tsinghua University [2]Shanghai AI Laboratory [3]The University of Hong Kong [4]The Chinese University of Hong Kong. Work was done when Dachuan Shi was an intern at Shanghai AI Laboratory. [†]Correspondence to: Chun Yuan <yuanc@sz.tsinghua.edu.cn>, Jiaqi Wang <wjqdev@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

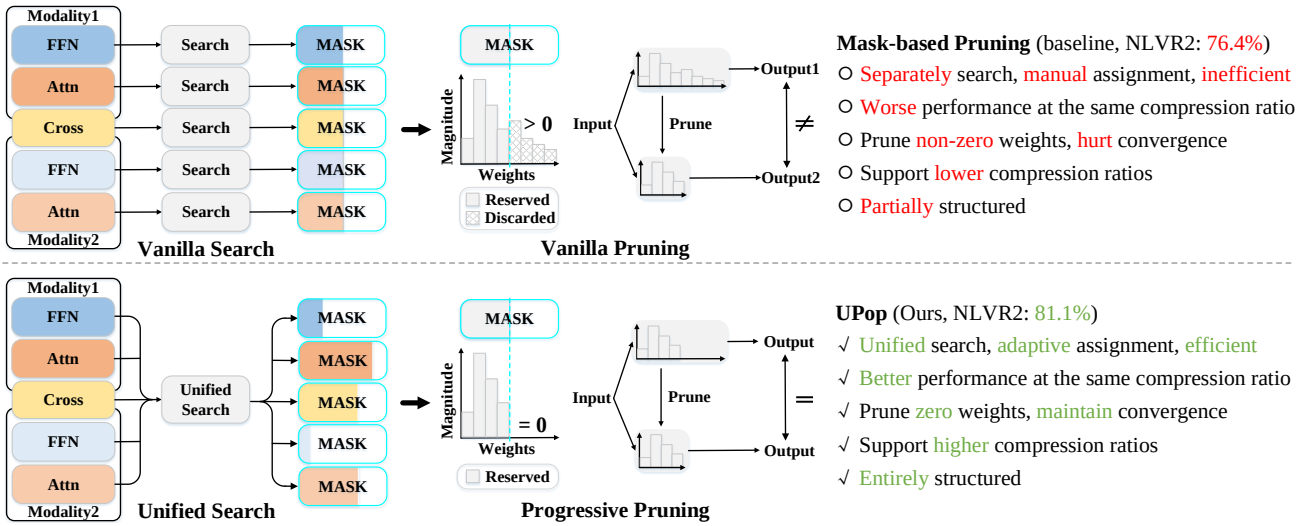

Figure 1: Comparison between the Mask-based Pruning (*e.g.*, extending ViT-Slimming (Chavan et al., 2022) to the multimodal scenario) and our UPop framework. Mask-based Pruning manually assigns each compressible component with a predefined compression ratio, which is inefficient and sub-optimal. Moreover, the vanilla pruning paradigm fails when it comes to higher compression ratios. UPop enables adaptively assigning the pruning ratio to each compressible component, which achieves significant performance improvements at the same overall compression ratio. Moreover, the progressive pruning paradigm eliminates the weight gap between the searched model and the pruned subnet to be retrained, therefore gaining better convergence and performance, especially at high compression ratios.

sentative ones. Benefiting from massive image-text pairs as pre-training datasets, they can learn joint representations of multiple modalities and can be further used to fine-tune on downstream tasks.

Although compression on unimodal tasks has been widely investigated, how to compress multimodal models, especially vision-language Transformers, is still relatively underexplored. In this paper, we propose a novel multimodal pruning framework, Unified and Progressive Pruning (UPop).

A straightforward design of multimodal pruning is to compress each modality separately via the unimodal pruning approach. However, there exist two main challenges. One of the challenges is that we have to manually explore suitable compression ratios for different components in different modalities, which is inefficient, especially when the model has multiple types of modules (these modules may comprise Self-Attentions, Cross-Attentions, and Feed-Forward Nets in both vision and language branches for typical vision-language Transformers). Moreover, when given a total compression ratio, the optimal compression ratio for different modalities and modules may vary, and therefore manual assignment is most likely sub-optimal. To overcome this shortcoming, we propose to unifiedly search on different modalities and different structures, which enables our approach to adaptively assign appropriate compression ratios among all compressible components given a total compression ratio. Comparison is illustrated in Figure 1 (Vanilla Search vs. Unified Search).

The second challenge is that the traditional two-stage pruning paradigm (*i.e.*, retraining after searching) fails when the compression ratio is high. After the search stage, unimportant neurons are going to be removed. However, many of them have non-zero weights, and suddenly binarizing them to zero after searching harms the convergence of the pruned subnet. In other words, the significant gap of parameter weights between the searched model (*i.e.*, model after the searching stage) and the pruned subnet to be retrained cause it is hard to converge and severely degrades the final performance. Consequently, we propose an improved pruning paradigm that conducts searching and retraining progressively and simultaneously, which ensures weights of removed neurons will progressively converge to zero before the end of the search stage, and therefore effectively eliminate the gap mentioned above. Comparison is illustrated in Figure 1 (Vanilla Pruning vs. Progressive Pruning).

Our main contributions can be summarized as

- In this paper, we propose a novel and universal multimodal pruning framework UPop for compressing vision-language Transformers.
- The proposed *Unified Pruning* enables adaptive compression ratio assignment among all compressible components. *Progressive Pruning* proposes an improved pruning paradigm that gains better convergence and supports higher compression ratios.
- As a structured pruning framework, UPop's effectiveness and versatility are validated on various multimodal tasks, datasets, and model architectures (*e.g.*, dual-stream CLIP (Radford et al., 2021) and mixed-stream BLIP (Li et al., 2022)), and also evaluated on unimodal tasks (*e.g.*, image classification and segmentation).

Table 2: Here we list the notations table. In the later part of the article, superscript $^{\{v,l,c\}}$ indicates notations for vision, language, and cross-modality, respectively, subscript $_{\{a,m\}}$ indicates notations for Attention and MLP structure, respectively.

NOTATION	DESCRIPTION	NOTATION	DESCRIPTION
L	Number of layers	H	Number of heads
N	Number of patches / Sequence length	D	Embedding size
d	Embedding size of each head	p	Total compression ratio
$\boldsymbol{\theta}$	Parameters of the original model	$\boldsymbol{\zeta}$	Parameters of the trainable mask
w	Regularization coefficient in searching	$\mathcal{F}_p$	$p\%$ compressed model $\mathcal{F}_p(x\|\boldsymbol{\theta},\boldsymbol{\zeta})$
$\alpha,\ \beta$	Learning rate during{search, retrain}	$T_{\{s,r\}}$	Iterations in {search, retrain} phase

2. Related Works

Vision-Language Transformer Recently, significant progress in vision-language tasks has been achieved by various Vision-Lanauge Transformers (Radford et al., 2021; Yu et al., 2022; Wang et al., 2022a), among which BLIP (Li et al., 2022) is one of the most representative models. BLIP is a pure transformed-based multimodal model, which employs a Bert (Devlin et al., 2018) and a ViT (Dosovitskiy et al., 2020) as text encoder and image encoder, respectively. To allow multimodal interaction, BLIP injects vision information from the image encoder into the text encoder by inserting an additional cross-attention layer after the self-attention layer of each transformer block in the text encoder.

Transformer Pruning There are several works exploring Transformers pruning on unimodal tasks. For example, structured pruning that removes layers (Fan et al., 2019), heads (Michel et al., 2019), or channels (Zhu et al., 2021), unstructured pruning (Yang et al., 2021; Chen et al., 2021b; Sanh et al., 2020; Chen et al., 2020; Liu et al., 2022b) that removes individual weights, and the intersection of structured and unstructured pruning such as ViT-Slimming (Chavan et al., 2022) that removes a different number of individual weights for different heads. UPop is a structured pruning approach whose minimum granularity is an entire row or column in the weights of model parameters.

Multimodal Transformer Compression A few works have investigated the compression of multimodal Transformers. For example, MiniVLM (Wang et al., 2020) suggest an efficient feature extractor and a compact BERT (Devlin et al., 2018) as basic components of visual-language models. DistillVLM (Fang et al., 2021) proposes that knowledge distillation can be used to mimic attention distributions from large vision-language models. The prior work (Gan et al., 2022) directly applies the unimodal pruning method (Han et al., 2015a) on the multimodal scenario to verify whether the lottery tickets hypothesis also exists on multimodal models. The difference between UPop and (Gan et al., 2022) are 3 aspects: 1) *Unified Pruning* enables adaptively instead of manually assigning the appropriate pruning ratio to each compressible component, 2) *Progressive Pruning* gains better convergence and performance at high compression ratios. 3) UPop is structured and relatively easier to deploy, while (Gan et al., 2022) is unstructured and harder to deploy.

Supplementary Due to the space constraint, we provide more related works about global pruning, iterative pruning, and parameter-efficient tuning in Appendix B.

3. Methodology

We propose *Unified and Progressive Pruning* as illustrated in Figure 2. Necessary notations are listed in Table 2. We start by revisiting *Mask-based Pruning* and straightforwardly extend it to the multimodal scenario.

3.1. Mask-based Pruning

Extended *Mask-based Pruning* compresses vision and language Transformers separately via unimodal *Mask-based Pruning*, consisting of a search phase and a retraining phase. Detailed implementation refers to Algorithm 2 in Appendix.

Search Take searching on Self-Attentions of Vision Transformer as an example. Denote the input of Self-Attention in the l^{th} layer as $\boldsymbol{a}_l \in \mathbb{R}^{N\times D}$, and every head h in the Self-Attention will transform $\boldsymbol{a}_l$ into query $\boldsymbol{q}_{l,h} \in \mathbb{R}^{N\times d}$, key $\boldsymbol{k}_{l,h} \in \mathbb{R}^{N\times d}$, and value $\boldsymbol{v}_{l,h} \in \mathbb{R}^{N\times d}$. The trainable mask $\boldsymbol{\zeta}_a^v \in \mathbb{R}^{L\times 1\times d}$ will be initialized to ones and inserted into Self-Attentions of each layer.[1] Then attention map of each head can be derived from

$$\boldsymbol{A}_{l,h} = \text{Softmax}\left((\boldsymbol{q}_{l,h} \odot \boldsymbol{\zeta}_{a,l}^v) \times (\boldsymbol{k}_{l,h} \odot \boldsymbol{\zeta}_{a,l}^v)^\top / \sqrt{d}\right). \quad (1)$$

The output of each head h can be derived from

$$\boldsymbol{O}_{l,h} = \boldsymbol{A}_{l,h} \times (\boldsymbol{v}_{l,h} \odot \boldsymbol{\zeta}_{a,l}^v) \in \mathbb{R}^{N\times d}. \quad (2)$$

Search on other structures (*e.g.*, Cross-Attentions, FFNs) and modalities (*e.g.*, vision, language) can be conducted similarly. Besides, the ℓ_1-norm of masks $\boldsymbol{\zeta}$ are added as additional loss items to drive the magnitude of masks smaller:

$$\mathcal{L} = \mathcal{L}_{\mathcal{O}} + w_a \sum\nolimits_{\boldsymbol{\zeta}_i\in\boldsymbol{\zeta}_a} \|\boldsymbol{\zeta}_i\|_1 + w_m \sum\nolimits_{\boldsymbol{\zeta}_i\in\boldsymbol{\zeta}_m} \|\boldsymbol{\zeta}_i\|_1 \quad (3)$$

where $\mathcal{L}_{\mathcal{O}}$ is the original loss to learn a multimodal model, and w_a and w_m are coefficients to balance the magnitude of loss items. It means that the model parameters $\boldsymbol{\theta}$ and trainable masks $\boldsymbol{\zeta}$ are optimized jointly in the search phase.

[1] More fine-grained mask shape $\mathbb{R}^{L\times H\times d}$ will result in pruned heads within a layer has different dimensions, and thus matrix computation of attention map becomes unfeasible on regular devices.

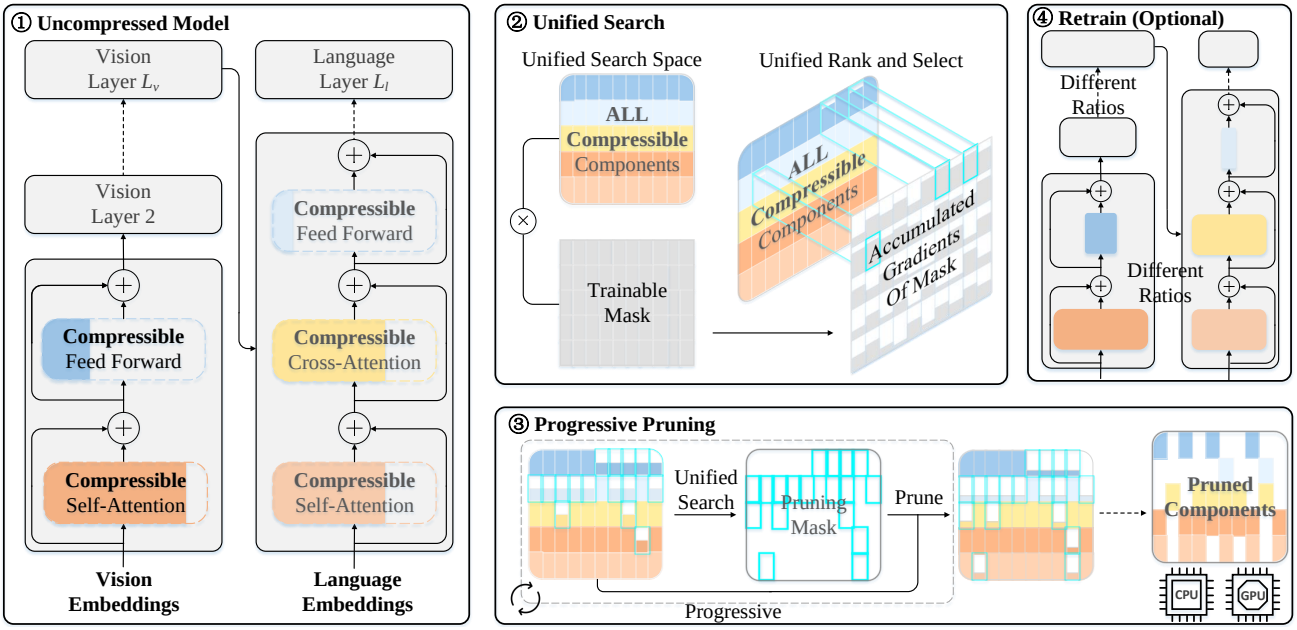

Figure 2: Diagram of *Unified and Progressive Pruning* (UPop) framework. (1) Trainable masks are initialized to ones and inserted into Self-Attention, Cross-Attention, and MLP (Feed Forward Network) in each modality. (2) Combine all compressible components and trainable masks as a unified search space. Then, the current pruning mask is generated based on unified ranking and selecting the importance metric (*i.e.*, accumulated gradients of the trainable masks). (3) Repeat the cycle consisting of unified search and progressive pruning until the target total compression ratio is reached. (4) Pruned subnet can be further fine-tuned to achieve better performance.

Retraining After the search, the subnet can be pruned from the searched model based on mask $\boldsymbol{\zeta}$. The magnitude of the mask is used as the metric to evaluate the importance of corresponding neurons. Neurons with the smallest magnitude of $p\%$ in the mask are removed (*i.e.*, binarized as zero during retraining) from the searched model. The obtained subnet is retrained to get the final compressed model.

The major weakness of *Mask-based Pruning* is two-fold: 1) the mask $\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}$ on each module is assigned with a compression ratio manually, which is inefficient and sub-optimal, especially when the modules are usually various in a multimodal model; 2) for those neurons to be removed after search, their corresponding magnitude in the searched mask is not guaranteed to be zero. There are a lot of non-zero neurons with relatively small mask magnitudes, and suddenly binarizing them to zero after search harms the convergence of the pruned subnet. We tackle the aforementioned issues with *Unified Pruning* and *Progressive Pruning*, respectively.

3.2. Unified and Progressive Pruning

3.2.1. Unified Pruning

The core idea of *Unified Pruning* is to unifiedly instead of separately search on different modalities and structures, which enables adaptively instead of manually assigning the appropriate pruning ratio to each compressible component. Detailed implementation refers to Algorithm 3 in Appendix.

Unified Search on Different Modalities *Unified Pruning* groups the pruning masks by computation mechanisms. For typical vision-language Transformers, we divide the masks $\boldsymbol{\zeta} = \{\boldsymbol{\zeta}^v_{att}, \boldsymbol{\zeta}^l_{att}, \boldsymbol{\zeta}^c_{att}, \boldsymbol{\zeta}^v_{mlp}, \boldsymbol{\zeta}^l_{mlp}\}$ into two groups:

$$\boldsymbol{\zeta}_a = \{\boldsymbol{\zeta}^v_{att}, \boldsymbol{\zeta}^l_{att}, \boldsymbol{\zeta}^c_{att}\}, \quad \boldsymbol{\zeta}_m = \{\boldsymbol{\zeta}^v_{mlp}, \boldsymbol{\zeta}^l_{mlp}\}. \tag{4}$$

One group $\boldsymbol{\zeta}_a$ for different attention modules and another $\boldsymbol{\zeta}_m$ for different MLP modules. The ranking and selection of masks are performed within each group. Instead of searching on each $\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}$ separately:

$$\boldsymbol{M}_i \leftarrow \texttt{TopKMask}(\boldsymbol{\zeta}_i^{(T_s)}, \ p \cdot \text{Size}(\boldsymbol{\zeta}_i)) \ \text{ for } \ \boldsymbol{\zeta}_i \in \boldsymbol{\zeta}, \tag{5}$$

where $\boldsymbol{M}_i$ is a binary mask used for pruning components of the subnet from the searched model. $\boldsymbol{M}_i$ is obtained by ranking and binarizing trainable mask $\boldsymbol{\zeta}_i$ at the final iteration T_s, which keeps the most important $p\%$ parameters. *Unified Pruning* searches on different modalities within each group which ranks weights across different components:

$$\boldsymbol{M}_a \leftarrow \texttt{TopKMask}(\{\boldsymbol{\zeta}_i^{(T_s)} | \boldsymbol{\zeta}_i \in \ \boldsymbol{\zeta}_a\}, \ p \cdot \text{Size}(\boldsymbol{\zeta}_a)), \tag{6}$$

$$\boldsymbol{M}_m \leftarrow \texttt{TopKMask}(\{\boldsymbol{\zeta}_i^{(T_s)} | \boldsymbol{\zeta}_i \in \ \boldsymbol{\zeta}_m\}, \ p \cdot \text{Size}(\boldsymbol{\zeta}_m)), \tag{7}$$

Unified Search on Different Structures We notice that simply uniting different structures degrades performance, and the reason is that the magnitude of the learned masks $\boldsymbol{\zeta}_i$ used for different structures vary greatly.

Intuitively, it is feasible to conduct unified searching after transforming the magnitudes distributions of different structures' masks to have the same mean and standard deviation, and thus masks $\boldsymbol{\zeta}_i$ used for different structures can be comparable. For the simplicity of implementation, we individually transform the mean and standard deviation of

Algorithm 1 UPop: Unified and Progressive Pruning

Input: Original model $\mathcal{F}$, parameters of the original model $\boldsymbol{\theta}$, parameters of the trainable mask $\boldsymbol{\zeta}$, total compression ratio p, iterations in the search stage T_s and retrain stage T_r, learning rate for the search stage α and retrain stage β
Output: Model $\mathcal{F}^\star$ after the search and retrain

1 **for** $t \leftarrow 0$ **to** $T_r - 1$ **do**
2 **if** $t < T_s$ **then**
3 # Calculate the loss $\mathcal{L}$, and normally update $\boldsymbol{\theta}$ with the original optimizer
4 $\mathcal{L} \leftarrow \mathcal{L}_\mathcal{O} + w_a \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_a} \|\boldsymbol{\zeta}_i\|_1 + w_m \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_m} \|\boldsymbol{\zeta}_i\|_1, \quad \boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)})$
5 # Calculate the gradient of loss $\mathcal{L}$ with respect to the trainable mask $\boldsymbol{\zeta}$
6 $\boldsymbol{G}^{(t)} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\zeta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)})$
7 # Conduct z-score standardization to make the current $\boldsymbol{G}_a$ and $\boldsymbol{G}_m$ comparable
8 $\boldsymbol{G}_a^{(t)} \leftarrow (\boldsymbol{G}_a^{(t)} - \mathbb{E}[\boldsymbol{G}_a^{(t)}]) / (\mathbb{E}[[\boldsymbol{G}_a^{(t)} - \mathbb{E}[\boldsymbol{G}_a^{(t)}]]^2])^{\frac{1}{2}}, \quad \boldsymbol{G}_m^{(t)} \leftarrow (\boldsymbol{G}_m^{(t)} - \mathbb{E}[\boldsymbol{G}_m^{(t)}]) / (\mathbb{E}[[\boldsymbol{G}_m^{(t)} - \mathbb{E}[\boldsymbol{G}_m^{(t)}]]^2])^{\frac{1}{2}}$
9 # Generate pruning mask $\boldsymbol{M}_t$ by ranking and selecting on accumulated gradient $\sum_{i=0}^{t} \boldsymbol{G}^{(i)}$
10 $p_t \leftarrow p(\frac{1}{2}(1 - \cos(\frac{\pi t}{T_s - 1})))^{\frac{1}{2}}, \quad \boldsymbol{M}^t \leftarrow \texttt{TopKMask}(\sum_{i=0}^{t} \boldsymbol{G}^{(i)},\ p_t \cdot \texttt{Size}(\boldsymbol{\zeta}))$
11 # Progressively compress $\boldsymbol{\zeta}_t$ based on M_t and accordingly progressively compress $\mathcal{F}$
12 $\boldsymbol{\zeta}^{(t+1)} \leftarrow (1 - \boldsymbol{M}^t) + (1 - \frac{p_t}{p}) \boldsymbol{M}^t, \quad \mathcal{F}_{p_{t+1}} \leftarrow \mathcal{F}_{p_t}(x | \boldsymbol{\theta}^{(t+1)}, \boldsymbol{\zeta}^{(t+1)})$
13 **else**
14 # Optional further finetune the pruned subnet $\mathcal{F}(x | \hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$ with the original optimizer
15 $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \mathcal{L}_\mathcal{O}(\boldsymbol{\theta}^{(t)})$
16 **return** $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x | \boldsymbol{\theta}^{(T_r)})$

magnitudes distributions of different structures' mask to the 0 and 1 by z-score standardization, respectively:

$$\boldsymbol{\zeta}_a^{(T_s)} \leftarrow (\boldsymbol{\zeta}_a^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_a^{(T_s)}]) / (\mathbb{E}[[\boldsymbol{\zeta}_a^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_a^{(T_s)}]]^2])^{\frac{1}{2}}, \quad (8)$$

$$\boldsymbol{\zeta}_m^{(T_s)} \leftarrow (\boldsymbol{\zeta}_m^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_m^{(T_s)}]) / (\mathbb{E}[[\boldsymbol{\zeta}_m^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_m^{(T_s)}]]^2])^{\frac{1}{2}}. \quad (9)$$

Then search on different modalities of different structures can be feasible:

$$\boldsymbol{M} \leftarrow \texttt{TopKMask}(\{\boldsymbol{\zeta}_i^{(T_s)} | \boldsymbol{\zeta}_i \in \boldsymbol{\zeta}\},\ p \cdot \texttt{Size}(\boldsymbol{\zeta})), \quad (10)$$

where M is a binary mask used for pruning all compressible components, and M is obtained by ranking and binarizing the whole trainable masks $\boldsymbol{\zeta}$ at the final iteration T_s.

3.2.2. Progressive Pruning

Retrain the pruned model after the search is a traditional two-stage paradigm. However, this paradigm fails when it comes to high compression ratios, because there is no guarantee that the magnitude of searched mask $\boldsymbol{\zeta}^{(T_s)}$ corresponding to the eliminated neurons in compressible components will converge to 0, which makes the pruned subnet with the parameters $\hat{\boldsymbol{\theta}}$ sliced from $\boldsymbol{\theta}^{(T_s)}$ difficult to converge. When the compression ratio becomes higher, the eliminated non-zero neurons from the parameters $\boldsymbol{\theta}^{(T_s)}$ is more, and the gap between $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\theta}^{(T_s)}$ is larger, thereby increasing the difficulty for the pruned subnet $\mathcal{F}(x | \hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$ to converge.

To address the above issue, we further propose the *Progressive Pruning*, whose core idea is to ensure each magnitude of the trainable mask $\boldsymbol{\zeta}$ corresponding to the eliminated neurons in compressible components converges to 0. This is achieved by updating trainable mask $\boldsymbol{\zeta}$ with a customed optimizer that is a function of the current iteration number t, instead of updating trainable mask $\boldsymbol{\zeta}$ with the same optimizer as the parameter $\boldsymbol{\theta}$ of the original model used.

Specifically, gradients $\boldsymbol{G}^{(t)}$ of $\boldsymbol{\zeta}$ in each iteration of the search phase is first collected:

$$\boldsymbol{G}^{(t)} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\zeta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)}), \quad (11)$$

where n is the number of batch size. Then the accumulated gradients $\sum_{i=0}^{t} \boldsymbol{G}^{(i)}$ can be used as a new metric to evaluate the importance of corresponding neurons. And the pruning mask $\boldsymbol{M}^t$ at this iteration can be generated accordingly:

$$\boldsymbol{M}^t \leftarrow \texttt{TopKMask}(\sum_{i=0}^{t} \boldsymbol{G}^{(i)},\ p_t \cdot \texttt{Size}(\boldsymbol{\zeta})), \quad (12)$$

where p_t is the current compression ratio when the iteration number is t. And the update strategy for optimizing $\boldsymbol{\zeta}$ in each iteration of the search phase can be written as

$$\boldsymbol{\zeta}^{(t+1)} \leftarrow (1 - \boldsymbol{M}_i^t) + (1 - \frac{p_t}{p}) \boldsymbol{M}_i^t, \quad (13)$$

which ensures that as p_t progressively increases to p, each magnitude of mask $\boldsymbol{\zeta}$ corresponding to the removed neurons in compressible components will exactly converge to 0. *Progressive Pruning* eliminates the parameter gap between the searched model and the pruned subnet to be retrained, therefore gaining better convergence and performance, especially at high compression ratios.

The proposed *UPop* framework combines *Unified Pruning* and *Progressive Pruning* as outlined in Algorithm 1. Line 2 $\sim$ 12 implements the search phase where Line 10 calculates the current compression ratio p_t to be achieved (detailed discussion is provided in Appendix C.2), and Line 13 $\sim$ 15 implements an optional retrain phase.

Table 3: Compression results on the NLVR2. Bold indicates the best performance at the same compression ratio. Reduce indicates compression times. The marker ✓ or ✗ indicates whether the model converges at the current compression times. The units of Params and FLOPs are M and G, respectively.

Approach	Reduce	Status	Dev Acc	Test Acc	Params	FLOPs
Uncompressed	1×	✓	82.48	83.08	259.45	132.54
Mask-based Pruning	2×	✓	75.74	76.44	146.18	66.88
	3×	✗	✗	✗	✗	✗
Unified Pruning (Ours)	2×	✓	79.50	80.32	149.90	95.01
	3×	✓	71.25	71.66	106.33	68.19
	4×	✗	✗	✗	✗	✗
Unified and Progressive Pruning (Ours)	2×	✓	**80.33**	**81.13**	150.15$_{\downarrow 42\%}$	89.36$_{\downarrow 33\%}$
	3×	✓	**76.89**	**77.61**	109.01$_{\downarrow 58\%}$	65.29$_{\downarrow 51\%}$
	4×	✓	**72.85**	**73.55**	88.61$_{\downarrow 66\%}$	50.35$_{\downarrow 62\%}$
	5×	✓	**68.71**	**68.76**	76.81$_{\downarrow 70\%}$	39.93$_{\downarrow 70\%}$
	10×	✓	**57.17**	**57.79**	54.48$_{\downarrow 79\%}$	19.08$_{\downarrow 86\%}$

Table 4: Performance of the 2× compressed BLIP model on the NLVR2 while searching only and without any retraining. The marker ✗ indicates the model fails converging.

Approach w/o Retrain	Dev Acc	Test Acc
Mask-based Pruning	✗	✗
Unified Pruning (Ours)	✗	✗
UPop (Ours)	**76.89**	**77.84**

Table 5: Performance of the 2× compressed BLIP model on the NLVR2 while retraining only one epoch after searching.

Approach w/ Retrain	Dev Acc	Test Acc
Mask-based Pruning	62.82	63.35
Unified Pruning (Ours)	75.42	75.30
UPop (Ours)	**79.08**	**80.08**

Table 6: Comparisons of the 2 × compressed BLIP model on the NLVR2. The superscript *: The original approach is unstructured, which has the finest compression granularity, and therefore we report the performance at the same granularity as ours to achieve a fair comparison. The STE †: use straight-through estimator (Bengio et al., 2013) to approximate gradients.

Approach	Dev Acc	Test Acc	Params
Uncompressed (Li et al., 2022)	82.48	83.08	259.45
Iterative Magnitude-based Pruning* (Gan et al., 2022)	66.88	67.19	151.86
Mask-based Pruning (Chavan et al., 2022)	75.74	76.44	146.18
Mask-based Pruning (Chavan et al., 2022) w/ Iterative Pruning with STE† (Sanh et al., 2020)	78.05	77.68	146.11
Unified Pruning (Ours)	79.50	80.32	149.90
Unified Pruning (Ours) w/ Iterative Pruning with STE† (Sanh et al., 2020)	80.01	80.54	145.86
Unified Pruning (Ours) w/ Progressive Pruning (Ours)	**80.33**	**81.13**	150.15

4. Experiments

We report the performance of UPop on a series of multimodal tasks, including Visual Reasoning, Image Captioning, Visual Question Answer, and Image-Text Retrieval. Due to the space constraint, we provide more ablation studies and experiments on unimodal tasks in Appendix C.

4.1. Experiments on the Visual Reasoning Task

NLVR2 is a binary classification visual reasoning task with two images and a text description as inputs. To quantitatively evaluate the proposed UPop, we compress the fine-tuned BLIP model on this task at a ratio of 2, 3, 4, 5, and 10 times, respectively.[2] The model consists of two weight-shared ViT as image encoder and a Bert with two cross-attention as text encoder, therefore the mask ζ corresponding to the compressible components on this model is $\zeta = \{\zeta_a^v, \zeta_m^v, \zeta_a^l, \zeta_m^l, \zeta_a^{c0}, \zeta_a^{c1}\}$. As shown in Table 3 and 6, we compress the original model with Mask-based Pruning, Magnitude-based Pruning, Iterative Pruning, Unified Pruning, their combinations, and UPop, respectively.

[2]Note that at N times compression, the total number of parameters will not be strictly equal to the $\frac{1}{N}$ of the original model. This is because some modules of the original model are not covered by the mask ζ, such as the patch embedding module, the word embedding module, and the classification head. In addition, at the same compression ratio, different searched masks will also lead to different structures and FLOPs of the compressed model.

4.2. Effect of Unified Pruning

At the 2× compression ratio, Table 3 shows that compared to the Mask-based Pruning, Unified Pruning gains 3.76% and 3.88% accuracy improvement on the dev set and test set, respectively. Furthermore, Unified Pruning converges successfully at the 3× compression ratio, while Mask-based Pruning does not. Table 6 shows that Unified Pruning also outperforms Magnitude-based Pruning under the same setting of the compression ratio and granularity.

Unified Pruning enables the model to adaptively assign appropriate compression ratios among different compressible components. Table 7 demonstrates that Unified Pruning can rescue us from the burden of repeated experiments (*e.g.*, doing grid search) for searching the optimal compression ratio assignment. Furthermore, Figure 3 visualizes the proportion of all compressible components retained in the compressed

Table 7: Comparisons between manually and adaptively assigning different compression ratios to different components of different modalities. Experiments are conducted with the 2 × compressed BLIP model on the NLVR2. The percentages in the table indicate the remaining parameters. Note that although the total compression ratio is the same for all experiments, the summation of individual percentages per row is different because the proportion of each component to the total number of model parameters is different.

Assignment	Vision Attention	Language Attention	Cross Attention1	Cross Attention2	Vision FFN	Language FFN	Test Acc
Maunal	50%	50%	50%	50%	50%	50%	76.44
Maunal	22%	62%	26%	75%	59%	53%	73.75
Maunal	44%	80%	17%	27%	83%	41%	80.01
Maunal	26%	73%	70%	63%	12%	64%	61.31
Maunal	58%	81%	19%	18%	47%	72%	78.80
Adaptive (Ours)	81%	58%	18%	19%	72%	47%	**81.13**

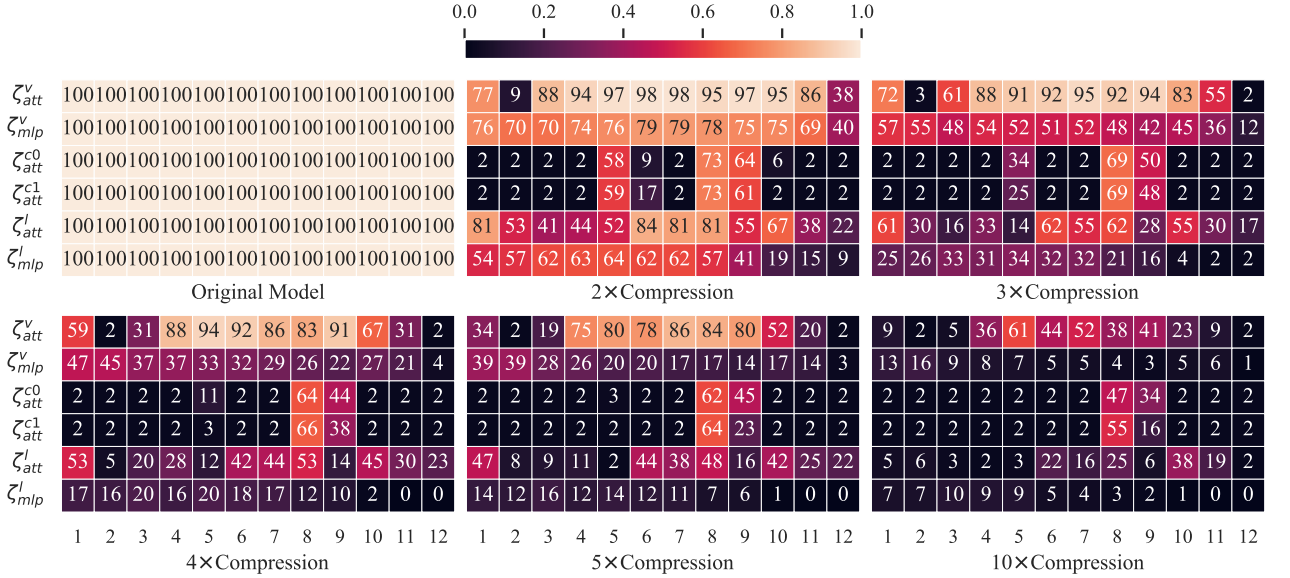

Figure 3: The proportion of all compressible components retained in the compressed BLIP model on the NLVR2. These six subfigures represent the original model and the compressed model at the $2\times$, $3\times$, $4\times$, $5\times$, and $10\times$ compression ratio, respectively. In each subfigure, the horizontal axis represents the layer number, the vertical axis represents the compressible components corresponding to each $\boldsymbol{\zeta}_i$, and the number in cells represents the retained proportion of a certain component's certain layer.

model. It can be observed from the figure that retained proportion of each compressible component has significantly different trends as the compression ratio increases. Moreover, there are obviously unbalanced compression assignments in different layers at different compression ratios.

4.3. Effect of Progressive Pruning

As shown in Table 3, at the $2\times$ compression ratio, the Unified and Progressive Pruning (UPop) gains further 0.83% and 0.81% accuracy improvement on the dev set and test set compared to the Unified Pruning. Moreover, at the $3\times$ compression, the improvements are extended to 5.64% and 5.95%, respectively. At the higher $4\times$, $5\times$, and $10\times$ compression ratio, the Progressive Pruning can still enable the compressed model to converge successfully, while both Mask-based Pruning and Unified Pruning fail. Furthermore, Table 6 shows that the proposed Progressive Pruning also outperforms Iterative Pruning with STE under the same setting of the compression ratio and granularity.

To further illustrate how Progressive Pruning strengthens the convergence capability of the compressed model, we compare the performance of pruned subnets in the situation that search without any retraining or search with only one epoch retraining. Table 4 shows that the model compressed by UPop can converge without any retraining while the other two compression approaches fail. Furthermore, Table 5 shows that with only one epoch retraining, the model compressed by UPop converges at significantly superior performance to the other two approaches. The experiments in Table 4 and 5 indicate that Progressive Pruning maintains the convergence capability of the compressed model by initializing the pruned subnet to be retrained with better parameter weights.

4.4. Experiments on the Image Caption Task

To validate the versatility of the proposed UPop, we further conducted experiments on the Image Caption task. We compress the fine-tuned BLIP model on the COCO dataset

Table 8: Compression results on the Image Caption task and the Visual Question Answering task. The CIDEr, SPICE, test-dev, and test-std are the higher the better. The units of Params and FLOPs are M and G, respectively.

Approach	Reduce	Image Caption				Visual Question Answering			
		CIDEr	SPICE	Params	FLOPs	test-dev	test-std	Params	FLOPs
Uncompressed	1×	133.3	23.8	224.0	65.7	77.4	77.5	361.6	186.1
Mask-based Pruning	2×	112.9	21.0	124.9	33.2	71.6	71.6	205.8	96.4
	4×	60.7	12.8	75.4	17.1	69.2	69.3	128.4	51.7
Unified Pruning (Ours)	2×	127.9	23.1	124.7	44.2	75.2	75.4	216.4	118.7
	4×	100.3	19.1	77.5	25.6	73.5	73.6	135.3	77.3
UPop (Ours)	2×	**128.9**	**23.3**	127.1$_{\downarrow 43\%}$	39.8$_{\downarrow 39\%}$	**76.3**	**76.3**	211.3$_{\downarrow 42\%}$	109.4$_{\downarrow 41\%}$
	4×	**117.4**	**21.7**	76.5$_{\downarrow 66\%}$	22.2$_{\downarrow 66\%}$	**74.5**	**74.6**	133.3$_{\downarrow 63\%}$	62.3$_{\downarrow 67\%}$

Table 9: Compress BLIP on the COCO and Flickr30K datasets of the Image-Text Retrieval task. The R@1, R@5, and R@10 are the higher the better. The units of Params and FLOPs are M and G, respectively.

Dataset	Approach	Reduce	Image → Text			Text → Image			Params	FLOPs
			R@1	R@5	R@10	R@1	R@5	R@10		
COCO (5K test set)	Uncompressed	1×	81.9	95.4	97.8	64.3	85.7	91.5	447.6	153.2
	Mask-based Pruning	2×	61.7	85.0	91.1	46.0	73.2	82.6	249.5	77.3
		4×	✗	✗	✗	✗	✗	✗	✗	✗
	Unified Pruning (Ours)	2×	75.4	92.9	96.3	57.6	81.9	88.7	253.1	103.4
		4×	40.3	69.3	80.2	31.3	58.8	70.7	148.7	61.4
	UPop (Ours)	2×	**77.4**	**93.4**	**97.0**	**59.8**	**83.1**	**89.8**	248.9$_{\downarrow 44\%}$	88.3$_{\downarrow 42\%}$
		4×	**62.9**	**86.2**	**92.3**	**47.4**	**74.8**	**83.9**	147.9$_{\downarrow 67\%}$	50.2$_{\downarrow 67\%}$
Flickr30K (1K test set)	Uncompressed	1×	96.8	99.9	100.0	86.9	97.3	98.7	447.6	153.2
	Mask-based Pruning	2×	78.9	92.7	95.5	63.8	85.1	90.1	249.3	77.2
		4×	✗	✗	✗	✗	✗	✗	✗	✗
	Unified Pruning (Ours)	2×	92.2	99.0	**99.8**	78.5	93.7	96.1	252.3	104.1
		4×	50.0	76.1	84.3	40.8	68.1	77.0	148.7	60.8
	UPop (Ours)	2×	**94.0**	**99.5**	99.7	**82.0**	**95.8**	**97.6**	250.5$_{\downarrow 44\%}$	91.0$_{\downarrow 41\%}$
		4×	**85.8**	**97.4**	**98.4**	**71.3**	**91.0**	**94.8**	147.6$_{\downarrow 67\%}$	51.0$_{\downarrow 67\%}$

at a ratio of 2 and 4 times, respectively. The model consists of a ViT as the image encoder and a Bert with cross-attention as the text decoder. Therefore the mask $\boldsymbol{\zeta}$ corresponding to the compressible components on this model is $\boldsymbol{\zeta} = \{\zeta_a^v, \zeta_m^v, \zeta_a^l, \zeta_m^l, \zeta_a^c\}$. Table 8 shows that UPop also achieves superior performance on the Image Caption task.

4.5. Experiments on the Visual QA Task

We compress the fine-tuned BLIP model on the VQA2.0 dataset at a ratio of 2 and 4 times, respectively. The model consists of a ViT as the image encoder, a Bert with cross-attention as the text encoder, and a Bert with cross-attention as the text decoder. Therefore the mask $\boldsymbol{\zeta}$ corresponding to the compressible components on this model is $\boldsymbol{\zeta} = \{\zeta_a^v, \zeta_m^v, \zeta_a^{l,en}, \zeta_m^{l,en}, \zeta_a^{l,de}, \zeta_m^{l,de}\}$. Table 8 shows the improved performance of UPop on the VQA task.

4.6. Experiments on the Retrieval Task

We compress the fine-tuned BLIP model on the COCO and Flickr30K datasets at a ratio of 2 and 4 times, respectively. The model consists of a ViT as the image encoder, a Bert with cross-attention as the text encoder, an extra ViT as the momentum image encoder, and an extra Bert with cross-attention as the momentum text encoder. Since the momentum models are updated by taking the moving average of normal models, we do not add the compression mask into the momentum models. Therefore the mask $\boldsymbol{\zeta}$ corresponding to the compressible components on this model is $\boldsymbol{\zeta} = \{\zeta_a^v, \zeta_m^v, \zeta_a^l, \zeta_m^l, \zeta_a^c\}$. Table 9 shows the improved performance of UPop on the Image-Text Retrieval task.

To further validate the versatility of UPop on different model architectures, we also compressed the dual-stream architecture, CLIP (Radford et al., 2021), on the Image-Text Retrieval task. Table 10 shows that UPop is able to achieve

Table 10: Compress CLIP on the COCO and Flickr30K datasets of the Image-Text Retrieval task. Notations are the same as in Table 9.

Dataset	Approach	Reduce	Image → Text			Text → Image			Params	FLOPs
			R@1	R@5	R@10	R@1	R@5	R@10		
COCO (5K test set)	Uncompressed	1×	71.5	90.8	95.4	56.8	80.7	87.6	856.0	395.7
	UPop (Ours)	2×	70.8	90.8	95.2	53.1	79.9	87.3	473.7$_{\downarrow 45\%}$	196.3$_{\downarrow 50\%}$
		4×	56.1	82.4	90.2	41.1	71.0	81.4	280.2$_{\downarrow 67\%}$	105.9$_{\downarrow 73\%}$
Flickr30K (1K test set)	Uncompressed	1×	96.8	100.0	100.0	86.6	97.8	99.1	856.0	395.7
	UPop (Ours)	2×	93.2	99.4	99.8	80.5	95.4	97.6	474.3$_{\downarrow 45\%}$	201.1$_{\downarrow 49\%}$
		4×	82.9	95.7	97.8	67.3	89.5	93.5	278.5$_{\downarrow 67\%}$	102.6$_{\downarrow 74\%}$

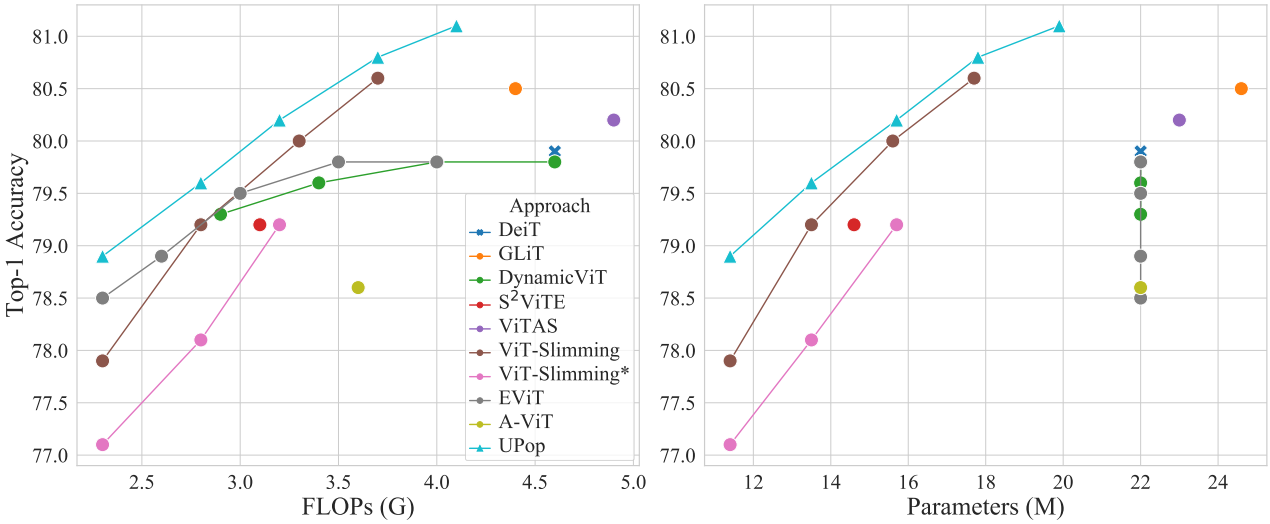

Figure 4: The left and right subfigures illustrate the Accuracy-FLOPs and Accuracy-Parameter trade-off, respectively. * indicates the performance of the deployable model if the original model is non-deployable. Two subfigures demonstrate that the proposed UPop (marked with the blue triangle) achieves better performance on both trade-offs. Note that token-specific compression approaches only reduce FLOPs and not the number of parameters. Therefore they are vertical lines in the Accuracy-Parameter trade-off figure.

comparable effectiveness to BLIP on CLIP. [3]

4.7. Experiments on the Image Classification Task

In addition to the multimodal tasks that UPop mainly focuses on, UPop can also be adapted to unimodal tasks by combining Unified Search on different structures and Progressive Pruning. As illustrated in Figure 4 and reported in Appendix Table 16, we conduct DeiT (Touvron et al., 2021) compression on ImageNet dataset (Deng et al., 2009), and UPop can also achieve competitive performance compared to other unimodal compression SOTA approaches.

5. Conclusion

This paper proposes a novel multimodal pruning framework, Unified and Progressive Pruning (UPop), for vision-language Transformers. UPop unifiedly searches on all compressible components consisting of Self-Attentions, MLPs, and Cross-Attentions of all modalities, and thus can adaptively assign appropriate compression ratios for all components. Moreover, analysis of masks indicates that the importance of components for compression varies. Therefore, the proposed unified search is a better choice than manually assigning compression ratios among different components, which is inefficient and sub-optimal. Furthermore, UPop progressively conducts search and retraining, which effectively strengthens the convergence capability of the compressed model and enables higher compression ratios.

6. Acknowledgements

This work was supported by the National Key R&D Program of China (2022YFB4701400/4701402), SZSTC Grant (JCYJ20190809172201639, WDZC20200820200655001), Shenzhen Key Laboratory (ZDSYS20210623092001004), Beijing Key Lab of Networked Multimedia, and Shanghai AI Laboratory.

[3] Note that we use the momentum distillation to finetune CLIP on the Image-Text Retrieval task. Due to the introduction of momentum models, the number of parameters and FLOPs in Table 10 are approximately twice as high as the original CLIP, respectively.

References

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Bousselham, W., Thibault, G., Pagano, L., Machireddy, A., Gray, J., Chang, Y. H., and Song, X. Efficient self-ensemble for semantic segmentation. *arXiv e-prints*, pp. arXiv–2111, 2021.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chavan, A., Shen, Z., Liu, Z., Liu, Z., Cheng, K.-T., and Xing, E. P. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4931–4941, 2022.

Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., and Ouyang, W. Glit: Neural architecture search for global and local image transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–21, 2021a.

Chen, D., Tao, C., Hou, L., Shang, L., Jiang, X., and Liu, Q. Litevl: Efficient video-language learning with enhanced spatial-temporal modeling. *arXiv preprint arXiv:2210.11929*, 2022.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., and Carbin, M. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

Chen, T., Cheng, Y., Gan, Z., Yuan, L., Zhang, L., and Wang, Z. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021b.

Contributors, M. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Fan, A., Grave, E., and Joulin, A. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

Fang, Z., Wang, J., Hu, X., Wang, L., Yang, Y., and Liu, Z. Compressing visual-linguistic model via knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1428–1438, 2021.

Gan, Z., Chen, Y.-C., Li, L., Chen, T., Cheng, Y., Wang, S., Liu, J., Wang, L., and Liu, Z. Playing lottery tickets with vision and language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 652–660, 2022.

Goyal, S., Choudhury, A. R., Raje, S., Chakaravarthy, V., Sabharwal, Y., and Verma, A. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pp. 3690–3699. PMLR, 2020.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015b.

He, R., Liu, L., Ye, H., Tan, Q., Ding, B., Cheng, L., Low, J.-W., Bing, L., and Si, L. On the effectiveness of adapter-based tuning for pretrained language model adaptation. *arXiv preprint arXiv:2106.03164*, 2021.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Huang, Y., Wang, W., and Wang, L. Instance-aware image and sentence matching with selective multimodal lstm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2310–2318, 2017.

Hyeon-Woo, N., Ye-Bin, M., and Oh, T.-H. Fedpara: Low-rank hadamard product for communication-efficient federated learning. *arXiv preprint arXiv:2108.06098*, 2021.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pp. 4904–4916. PMLR, 2021.

Jia, X., Gavves, E., Fernando, B., and Tuytelaars, T. Guiding long-short term memory for image caption generation, 2015. URL https://arxiv.org/abs/1509.04942.

Karpathy, A., Joulin, A., and Fei-Fei, L. F. Deep fragment embeddings for bidirectional image sentence mapping. *Advances in neural information processing systems*, 27, 2014.

Kim, W., Son, B., and Kim, I. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pp. 5583–5594. PMLR, 2021.

Kiros, R., Salakhutdinov, R., and Zemel, R. S. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Li, J., Li, D., Xiong, C., and Hoi, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *arXiv preprint arXiv:2201.12086*, 2022.

Li, X., Yin, X., Li, C., Zhang, P., Hu, X., Zhang, L., Wang, L., Hu, H., Dong, L., Wei, F., et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pp. 121–137. Springer, 2020.

Liang, Y., Ge, C., Tong, Z., Song, Y., Wang, J., and Xie, P. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Liu, X., Ji, K., Fu, Y., Tam, W. L., Du, Z., Yang, Z., and Tang, J. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021a.

Liu, X., Ji, K., Fu, Y., Tam, W., Du, Z., Yang, Z., and Tang, J. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 61–68, 2022a.

Liu, Y., Meng, F., Lin, Z., Li, J., Fu, P., Cao, Y., Wang, W., and Zhou, J. A win-win deal: Towards sparse and robust pre-trained language models. *arXiv preprint arXiv:2210.05211*, 2022b.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021b.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022c.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Lu, J., Batra, D., Parikh, D., and Lee, S. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

Michel, P., Levy, O., and Neubig, G. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., and Hsieh, C.-J. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Sanh, V., Wolf, T., and Rush, A. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389, 2020.

Shao, J., Chen, S., Li, Y., Wang, K., Yin, Z., He, Y., Teng, J., Sun, Q., Gao, M., Liu, J., et al. Intern: A new learning paradigm towards general vision. *arXiv preprint arXiv:2111.08687*, 2021.

Shi, D., Liu, R., Tao, L., He, Z., and Huo, L. Multi-encoder parse-decoder network for sequential medical image segmentation. In *2021 IEEE international conference on image processing (ICIP)*, pp. 31–35. IEEE, 2021.

Shi, D., Liu, R., Tao, L., and Yuan, C. Heuristic dropout: An efficient regularization method for medical image segmentation models. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1101–1105. IEEE, 2022.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

Strudel, R., Garcia, R., Laptev, I., and Schmid, C. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7262–7272, 2021.

Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., and Xu, C. Vitas: vision transformer architecture search. In *European Conference on Computer Vision*, pp. 139–157. Springer, 2022.

Suhr, A., Zhou, S., Zhang, A., Zhang, I., Bai, H., and Artzi, Y. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018.

Sung, Y.-L., Cho, J., and Bansal, M. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5227–5237, 2022.

Tan, H. and Bansal, M. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

Tao, C., Hou, L., Zhang, W., Shang, L., Jiang, X., Liu, Q., Luo, P., and Wong, N. Compression of generative pre-trained language models via quantization. *arXiv preprint arXiv:2203.10705*, 2022.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Touvron, H., Cord, M., and Jégou, H. Deit iii: Revenge of the vit. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIV*, pp. 516–533. Springer, 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

Wang, J., Hu, X., Zhang, P., Li, X., Wang, L., Zhang, L., Gao, J., and Liu, Z. Minivlm: A smaller and faster vision-language model. *arXiv preprint arXiv:2012.06946*, 2020.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 568–578, 2021.

Wang, W., Bao, H., Dong, L., Bjorck, J., Peng, Z., Liu, Q., Aggarwal, K., Mohammed, O. K., Singhal, S., Som, S., et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022a.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022b.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances*

in Neural Information Processing Systems, 34:12077–12090, 2021.

Yang, H., Yin, H., Molchanov, P., Li, H., and Kautz, J. Nvit: Vision transformer compression and parameter redistribution. *arXiv preprint arXiv:2110.04869*, 2021.

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 21–29, 2016.

Yang, Z., Li, Z., Shao, M., Shi, D., Yuan, Z., and Yuan, C. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022.

Yin, H., Vahdat, A., Alvarez, J. M., Mallya, A., Kautz, J., and Molchanov, P. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10809–10818, 2022.

Young, P., Lai, A., Hodosh, M., and Hockenmaier, J. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Yu, J., Wang, Z., Vasudevan, V., Yeung, L., Seyedhosseini, M., and Wu, Y. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Yu, X., Liu, T., Wang, X., and Tao, D. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7370–7379, 2017.

Yuan, Y., Chen, X., and Wang, J. Object-contextual representations for semantic segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pp. 173–190. Springer, 2020.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.

Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J., and Gao, J. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 13041–13049, 2020.

Zhu, M., Tang, Y., and Han, K. Vision transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021.

A. Implementation Details

A.1. Hyperparameter Settings

Table 11: Training hyperparameters for compressing BLIP-based models.

Hyperparameters	BLIP-NLVR (Li et al., 2022)	BLIP-Caption (Li et al., 2022)	BLIP-VQA (Li et al., 2022)	BLIP-Retrieval (Li et al., 2022)	
	NLVR2 (Suhr et al., 2018)	COCO (Lin et al., 2014)	VQAv2 (Goyal et al., 2017)	COCO (Lin et al., 2014)	Flickr30K (Young et al., 2014)
Optimizer	AdamW (Loshchilov & Hutter, 2017)				
AdamW β	(0.9, 0.999)				
Weight decay	0.05				
Batch size	256				
Search epochs	15	5	10	6	12
Search LR	3e-6	1e-5	2e-5	1e-5	1e-5
Retrain epochs	15	5	10	6	12
Retrain LR	3e-6	1e-5	2e-5	1e-5	1e-5
Search LR schedule	N/A				
Retrain LR schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)				
Data augmentation	RandomAugment (Cubuk et al., 2020)				

Table 12: Training hyperparameters for compressing CLIP, DeiT, and Segmenter.

Hyperparameters	CLIP (Radford et al., 2021)		DeiT (Touvron et al., 2021)	Segmenter (Strudel et al., 2021)
	COCO (Lin et al., 2014)	Flickr30K (Young et al., 2014)	ImageNet (Deng et al., 2009)	ADE20k (Zhou et al., 2017)
Optimizer	AdamW (Loshchilov & Hutter, 2017)			SGD (Robbins & Monro, 1951)
Optimizer settings	AdamW β (0.9, 0.999)			SGD momentum 0.9
Weight decay	0.2	0.2	0.05	0
Batch size	256	256	4096	64
Search epochs	6	12	60	16
Search LR	1e-5	1e-5	8e-4	4e-3
Retrain epochs	6	12	300	64
Retrain LR	1e-5	1e-5	8e-4	4e-3
Search LR schedule	N/A			
Retrain LR schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)			PolynomialLR (Strudel et al., 2021)
Data augmentation	RandomAugment (Cubuk et al., 2020)		RepeatedAugment (Touvron et al., 2021)	Standard pipline from MMSegmentation (Contributors, 2020)

Table 13: Structure hyperparameters for all models used in our experiments. The superscript * indicates 2 Transformers share parameters. The superscript †: 12 layers for the encoder and 2 layers for the decoder.

Model	Input resolution	Vision Transformer number	Vision Transformer layers	Vision Transformer width	Vision Transformer heads	Language Transformer number	Language Transformer layers	Language Transformer width	Language Transformer heads
BLIP-NLVR (Li et al., 2022)	384×384	2*	12	768	12	1	12	768	12
BLIP-Caption (Li et al., 2022)	384×384	1	12	768	12	1	12	768	12
BLIP-VQA (Li et al., 2022)	480×480	1	12	768	12	2	12	768	12
BLIP-Retrieval (Li et al., 2022)	384×384	2	12	768	12	2	12	768	12
CLIP (Radford et al., 2021)	336×336	2	24	1024	16	2	12	768	12
DeiT (Touvron et al., 2021)	224×224	1	12	384	6	0	-	-	-
Segmenter (Strudel et al., 2021)	512×512	2	(12, 2)†	384	6	0	-	-	-

A.2. Scope of Compressible Components

Self-Attentions, Cross-Attentions, and MLPs are widely used components in multimodal transformer layers. Consequently, the scope of compressible components in our experiments includes Self-Attentions, MLPs, and Cross-Attentions of both Vision Transformers and Language Transformers. Note that Cross-Attention only needs to be compressed if it exists. In early multimodal Transformers, *e.g.*, LXMERT (Tan & Bansal, 2019) and ViLBERT (Lu et al., 2019), Cross-Attention exists within both vision and language Transformers. In some more modern works, Cross-Attention exists in only one of the modalities, such as CoCa (Yu et al., 2022) and BLIP (Li et al., 2022). In addition, there are also a few models, such as CLIP (Radford et al., 2021), that do not have explicit Cross-Attention but only conduct cross-modality interaction by maximizing the cosine similarity of outputs from different modalities.

A.3. Compression Granularity and Deployability

UPop is a structured pruning approach whose minimum granularity is an entire row or column in the weights of model parameters, and a deployable pruning approach that allows the compressed model to be physically extracted from the original model. Generally speaking, structured approaches such as UPop are relatively easier to deploy, while unstructured approaches such as (Gan et al., 2022) are relatively hard to deploy. More specifically, suppose we are going to prune a fully connected layer with the parameter $\boldsymbol{\theta} \in \mathbb{R}^{w_1 \times w_2}$, then unstructured approaches will add a binary mask of the same shape $w_1 \times w_2$ on the $\boldsymbol{\theta}$, and every row and column of the mask may have a different amount of 0 after pruning, which results in difficulty for extracting all the weights with a mask of 1 to constitute a legal and smaller parameter matrix for the pruned layer. However, UPop will add a binary mask of the shape w_1 on the output of the fully connected layer. Hence each position in the mask of UPop corresponds to an entire row of the parameter matrix $\boldsymbol{\theta}$. It is simple to constitute a legal and smaller parameter matrix for the pruned layer by physically removing the entire rows from the original parameter matrix $\boldsymbol{\theta}$.

Besides, ViT-Slimming (Chavan et al., 2022) compress heads of Self-Attentions with unrestricted compression ratio, and thus the compressed model may have different embedding sizes of heads within a layer. However, the matrix computation of the attention map on regular hardware (*e.g.*, GPU cards) requires the query and key of each head within a layer to have the same embedding size. By restricting each head within the same layer to have the same compression ratio, UPop frees from non-deployable matrix computation and becomes structured across heads within individual layers.

A.4. Implentation of Mask-based Pruning

The *Mask-based Pruning* is outlined in Algorithm 2. Line $1 \sim 10$ implements the search phase, and Line $11 \sim 13$ implements the retrain phase.

Algorithm 2 Mask-based Pruning

Input: Original model $\mathcal{F}$, parameters of the original model $\boldsymbol{\theta}$, parameters of the trainable mask $\boldsymbol{\zeta}$, total compression ratio p, iterations in the search stage T_s and retrain stage T_r, learning rate for the search stage α and retrain stage β

Output: Model $\mathcal{F}^\star$ after the search and retrain

1 **for** $t \leftarrow 0$ **to** $T_s - 1$ **do**
2 $\quad$ `# Calculate the loss` $\mathcal{L}$
3 $\quad$ $\mathcal{L} \leftarrow \mathcal{L}_{\mathcal{O}} + w_a \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_a} \|\boldsymbol{\zeta}_i\|_1 + w_m \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_m} \|\boldsymbol{\zeta}_i\|_1$
4 $\quad$ `# Normally update both` $\boldsymbol{\theta}$ `and` $\boldsymbol{\zeta}$ `with the original optimizer`
5 $\quad$ $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)}), \quad \boldsymbol{\zeta}^{(t+1)} \leftarrow \boldsymbol{\zeta}^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\zeta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)})$
6 `# Individually generate each pruning mask` $\boldsymbol{M}_i$ `by ranking and selecting on each` $\boldsymbol{\zeta}_i$
7 **for** $\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}$ **do**
8 $\quad$ $\boldsymbol{M}_i \leftarrow \texttt{TopKMask}(\boldsymbol{\zeta}_i^{(T_s)}, p \cdot \texttt{Size}(\boldsymbol{\zeta}_i))$
9 `# Compress each` $\boldsymbol{\theta}_i$ `based on each` M_i `and accordingly compress` $\mathcal{F}$
10 $\hat{\boldsymbol{\theta}} \leftarrow \{\boldsymbol{\theta}_i^{(T_s)} | \boldsymbol{M}_i = 1\}, \quad \mathcal{F}_p \leftarrow \mathcal{F}(x|\hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$
11 `# Further finetune the pruned subnet` $\mathcal{F}(x|\hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$ `with the original optimizer`
12 **for** $t \leftarrow 0$ **to** $T_r - 1$ **do**
13 $\quad$ $\hat{\boldsymbol{\theta}}^{(t+1)} \leftarrow \hat{\boldsymbol{\theta}}^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_{\hat{\boldsymbol{\theta}}} \mathcal{L}_{\mathcal{O}}(\hat{\boldsymbol{\theta}}^{(t)})$
14 **return** $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x|\hat{\boldsymbol{\theta}}^{(T_r)})$

A.5. Implentation of Unified Pruning

The *Unified Pruning* is outlined in Algorithm 3. Line 1 $\sim$ 11 implements the search phase, and Line 12 $\sim$ 14 implements the retrain phase.

Algorithm 3 Unified Pruning

Input: Original model $\mathcal{F}$, parameters of the original model $\boldsymbol{\theta}$, parameters of the trainable mask $\boldsymbol{\zeta}$, total compression ratio p, iterations in the search stage T_s and retrain stage T_r, learning rate for the search stage α and retrain stage β
Output: Model $\mathcal{F}^\star$ after the search and retrain

1 **for** $t \leftarrow 0$ **to** $T_s - 1$ **do**
2 $\quad$ # Calculate the loss $\mathcal{L}$
3 $\quad$ $\mathcal{L} \leftarrow \mathcal{L}_{\mathcal{O}} + w_a \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_a} \|\boldsymbol{\zeta}_i\|_1 + w_m \sum_{\boldsymbol{\zeta}_i \in \boldsymbol{\zeta}_m} \|\boldsymbol{\zeta}_i\|_1$
4 $\quad$ # Normally update both $\boldsymbol{\theta}$ and $\boldsymbol{\zeta}$ with the original optimizer
5 $\quad$ $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)}), \quad \boldsymbol{\zeta}^{(t+1)} \leftarrow \boldsymbol{\zeta}^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\zeta}} \mathcal{L}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\zeta}^{(t)})$
6 # Conduct z-score standardization to make $\boldsymbol{\zeta}_a$ and $\boldsymbol{\zeta}_m$ after the search comparable
7 $\boldsymbol{\zeta}_a^{(T_s)} \leftarrow (\boldsymbol{\zeta}_a^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_a^{(T_s)}]) / (\mathbb{E}[[\boldsymbol{\zeta}_a^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_a^{(T_s)}]]^2])^{\frac{1}{2}}, \quad \boldsymbol{\zeta}_m^{(T_s)} \leftarrow (\boldsymbol{\zeta}_m^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_m^{(T_s)}]) / (\mathbb{E}[[\boldsymbol{\zeta}_m^{(T_s)} - \mathbb{E}[\boldsymbol{\zeta}_m^{(T_s)}]]^2])^{\frac{1}{2}}$
8 # Generate pruning mask $\boldsymbol{M}$ by ranking and selecting on $\boldsymbol{\zeta}$
9 $\boldsymbol{M} \leftarrow \texttt{TopKMask}(\boldsymbol{\zeta}^{(T_s)}, p \cdot \texttt{Size}(\boldsymbol{\zeta}))$
10 # Compress $\boldsymbol{\theta}$ based on M and accordingly compress $\mathcal{F}$
11 $\hat{\boldsymbol{\theta}} \leftarrow \{\boldsymbol{\theta}^{(T_s)} | \boldsymbol{M} = 1\}, \quad \mathcal{F}_p \leftarrow \mathcal{F}(x | \hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$
12 # Further finetune the pruned subnet $\mathcal{F}(x | \hat{\boldsymbol{\theta}}, \boldsymbol{\zeta}^{(T_s)})$ with the original optimizer
13 **for** $t \leftarrow 0$ **to** $T_r - 1$ **do**
14 $\quad$ $\hat{\boldsymbol{\theta}}^{(t+1)} \leftarrow \hat{\boldsymbol{\theta}}^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_{\hat{\boldsymbol{\theta}}} \mathcal{L}_{\mathcal{O}}(\hat{\boldsymbol{\theta}}^{(t)})$
15 **return** $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x | \hat{\boldsymbol{\theta}}^{(T_r)})$

B. Supplementary Related Works

Global pruning Movement Pruning (Sanh et al., 2020) prunes unimodal BERT (Devlin et al., 2018) and investigates the global pruning, which conducts ranking on the whole model. The differences between it and UPop are: 1) It only prunes across different structures in unimodality language, while UPop not only prunes across different structures but also prunes across modalities. Therefore, it and UPop are complementary in terms of scope. 2) It finds local (ranking inside each weight matrix) and global pruning perform similarly on language tasks. However, UPop finds separate and unified pruning performs notably differently on multimodal tasks. And the different conclusions may be attributed to the intrinsic properties of different modalities and could be a meaningful topic for future research. 3) It treats all weights equally when ranks mask of different structures (*i.e.*, Self-Attentions and MLPs). However, UPop notices that simply unified search space of different structures fail and proposes a solution based on z-score standardization to tackle this problem.

Iterative pruning Iterative pruning is an existing technique used by some prior works (Sanh et al., 2020; Liu et al., 2022b). The differences between it and UPop are: 1) It divides weights into multiple groups, and each time only binarizes a smaller number of non-zero weights to 0. For example, to achieve a 50% total compression ratio, it divides weights into five groups, and each time only binarizes 10% non-zero weights to 0. On the other hand, UPop prunes each weight to make them from the original value progressively converge to zero. For example, suppose there is a single weight with an original value of 2.8. Then UPop will progressively prune this value to 0 (*e.g.*, 2.8 to 2.7 to 2.6 to $\cdots$ to 0.1 to 0) while iterative pruning will directly prune it from 2.8 to 0. 2) Its binary masks take value from the set $\{0, 1\}$ while real-value masks of UPop take value from the interval $[0, 1]$. 3) Iterative pruning in (Sanh et al., 2020; Liu et al., 2022b) has to use the straight-through estimator (Bengio et al., 2013) to approximate gradients of binary masks to ensure weights can be normally updated, while this is not needed for real-value masks used by UPop.

Parameter-efficient Tuning Parameter-efficient tuning includes but not limited to low-rank adaptation (Hu et al., 2021; Hyeon-Woo et al., 2021), prompt tuning (Liu et al., 2022a; 2021a), parameter sharing (Lan et al., 2019; Shi et al., 2021), adapters (He et al., 2021; Sung et al., 2022), and dropout (Fan et al., 2019; Shi et al., 2022). The difference between them and UPop is that they are used for reducing learnable parameters during tuning instead of inference. In contrast, UPop is used for reducing the parameters and FLOPs during inference instead of tuning.

C. Supplementary Experiments and Analyses

C.1. Variation of Compressible Components and Layers

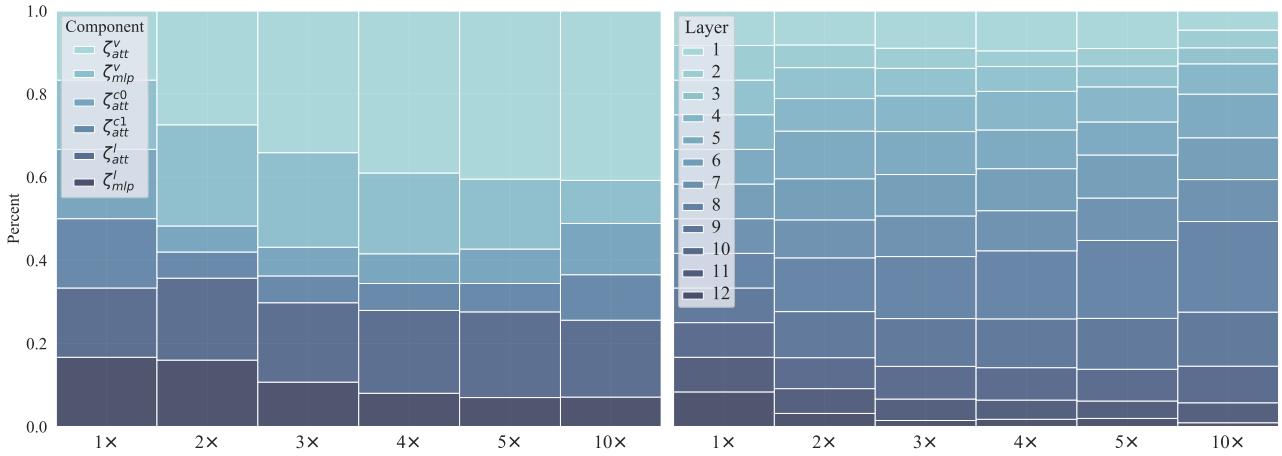

Figure 5: The left subfigure: variation of compressible components as the compression ratio increases. The right subfigure: variation of layers as the compression ratio increases.

Unified Pruning enables the model to adaptively assign appropriate compression ratios among different compressible components. Accordingly, we demonstrate the variation of all components and layers as the total compression ratio increases in Figure 5. The left subfigure shows that the retained percentage of Self-Attention of ViT and Self-Attention of Bert among all compressible components significantly increases as the compression ratio increases. In contrast, the retained percentage of MLP of ViT and MLP of Bert decreases. This indicates that Self-Attentions have higher importance than MLPs when the number of parameters is limited. It can also be observed that vision modality is more important than language modality in this task. The trend of the retained percentage of Cross-Attention generally decreases and then increases. This phenomenon indicates that at low compression ratios, the parameters of the visual and language modalities are relatively adequate. Therefore cross-attention is less important at this time. At high compression ratios, the vision and language modality lacks sufficient parameters, and cross-attention becomes more critical.

Similarly, the right subfigure of Figure 5 demonstrates the variation of all layers as the total compression ratio increases. It can be observed that the middle layers occupy an increasing proportion as the total compression ratio increases, which indicates that the majority of modalities' information is generated in the middle layers of the model. In the earlier layers, the information is not detailed enough. In contrast, in the last several layers, the refinement of the information becomes less critical when the number of parameters is limited.

C.2. Study on Update Strategy of Compression Ratio

Compression ratio p_t is a monotonically increasing function of iteration number t, and an intuitive design for updating p_t is to increase p_t evenly as t increases, *i.e.*:

$$p_t = p\frac{t}{T_s - 1} \tag{14}$$

It is worth noting that according to the implementation of Algorithm 1, the current compression ratio p_t of t^{th} iteration means that $p_t\%$ of embeddings has been compressed by $\frac{p_t}{p}\%$. As a consequence, the actual compression ratio a_t should be the ratio of the compressed embedding size multiplied by the ratio of each embedding that is compressed:

$$a_t = p_t \times \frac{p_t}{p} = \frac{p_t^2}{p} \tag{15}$$

In addition to the monotonically increasing property, a more appropriate update strategy than a uniform update strategy also needs to satisfy:

- On the one hand, the actual compression ratio should increase relatively slowly at the beginning of searching. Because when the iteration number t is small, the cumulative gradients are relatively volatile, and the generated mask is relatively inaccurate.

- On the other hand, the actual compression ratio should also increase relatively slowly toward the end of searching. Because as the current compression ratio gradually increases, the difficulty of compression also increases.

Formally speaking, a_t is supposed to satisfy:

$$\begin{cases} a_0 = 0 \\ a_{T_s-1} = p \\ \frac{\mathrm{d}a_t}{\mathrm{d}t} \geq 0, \ \forall t \in [0, T_s - 1] \\ \exists \ t_0 \in (0, T_s) \ \ s.t. \ \frac{\mathrm{d}^2 a_t}{\mathrm{d}t^2} > 0, \ \forall t \in (0, t_0), \text{and } \frac{\mathrm{d}^2 a_t}{\mathrm{d}t^2} < 0, \ \forall t \in (t_0, T_s - 1) \end{cases} \tag{16}$$

For example, the integration of trigonometric function $f(x) = \sin \frac{\pi x}{T_s - 1}$ defined on interval $[0, T_s - 1]$ satisfies the latter two requirements of the Equation 16. To further satisfy the first two properties, we only need to let

$$p \frac{\int_0^t \sin \frac{\pi x}{T_s-1} \mathrm{d}x}{\int_0^{T_s-1} \sin \frac{\pi x}{T_s-1} \mathrm{d}x} = \frac{p}{2}(1 - \cos \frac{\pi t}{T_s - 1}) = a_t = \frac{p_t^2}{p} \tag{17}$$

And thus

$$p_t = p(\frac{1}{2}(1 - \cos(\frac{\pi t}{T_s - 1})))^{\frac{1}{2}} \tag{18}$$

is a function that satisifies all requirements.

Table 14: Study on how the update strategy of compression ratio p_t affects the performance. The last one is adopted as our update strategy.

p_t	Dev Acc	Test Acc
$p\frac{t}{T_s-1}$	79.94	80.84
$p\frac{(2T_s-t+1)t}{((T_s+1)T_s)}$	**80.38**	**81.13**
$p(\frac{1}{2}(1 - \cos(\frac{\pi t}{T_s-1})))^{\frac{1}{2}}$	80.33	**81.13**

Table 14 shows the performance of the 2 $\times$ compressed BLIP-NLVR model with different p_t update strategies. The first one is the uniform update, while the last one is the strategy we adopted. There is obvious performance improvement when replacing the uniform update with $p(\frac{1}{2}(1 - \cos(\frac{\pi t}{T_s-1})))^{\frac{1}{2}}$. Besides, the last one is not the only feasible strategy, and other update strategies that satisfy requirements in Equation 16 should also achieve better performance than uniform update. For example, the second strategy $p\frac{(2T_s-t+1)t}{((T_s+1)T_s)}$ also satisfies requirements and also achieves comparable performance to the strategy we adopted.

C.3. Study on the frequency of Updating Compression Mask ζ

We also explore how the frequency of updating mask $\boldsymbol{\zeta}$ affects the model performance. Experimental results on the 2$\times$ compressed BLIP-NLVR model are reported in Table 15. Update compression mask $\boldsymbol{\zeta}$ at intervals has two benefits:

- On the one hand, it can reduce a small amount of computation during searching.
- On the other hand, it can be observed from Table 15 that updating the $\boldsymbol{\zeta}$ too frequently causes the compressed model to tend to overfit on the validation set.

Table 15: Study on how the frequency of updating compression mask $\boldsymbol{\zeta}$ affects the model performance. Frequency 50 is adopted by us.

Frequency	Dev Acc	Test Acc
1	**80.97**	80.14
10	80.48	80.86
50	80.33	**81.13**

The frequency 1 means updating $\boldsymbol{\zeta}$ each time the model parameters $\boldsymbol{\theta}$ are updated, while frequency 10 means updating $\boldsymbol{\zeta}$ once every 10 times the model parameters $\boldsymbol{\theta}$ are updated. Consequently, frequency 50 is adopted by us for the 2$\times$

compressed BLIP-NLVR model, which mitigates the overfitting in the validation set and improves the performance on the test set. It is worth noting that the appropriate frequency varies for different models and tasks. Empirically, setting the frequency to the number of iterations corresponding to the 1% compression ratio is more likely to be appropriate. For example, if we aim to accomplish 50% compression ratio in 1000 iterations, then a frequency about $1000 \times \frac{1}{50} = 20$ should be recommended.

C.4. Experiments on the Image Classification Task

Table 16: Compress DeiT on the ImageNet dataset. The units of Params and FLOPs are M and G, respectively. The superscript * indicates the performance of the deployable model if the original model is non-deployable. For fairness of comparison, all reported experimental results, including UPop, do not use knowledge distillation.

Approach	Top-1 (%)	Top-5 (%)	Params	FLOPs
DeiT (Touvron et al., 2021)	79.9	95.0	22.0	4.6
GLiT (Chen et al., 2021a)	80.5	-	24.6	4.4
DynamicViT (Rao et al., 2021)	79.3	-	22.0	2.9
S^2ViTE (Chen et al., 2021b)	79.2	-	14.6	3.1
ViTAS (Su et al., 2022)	80.2	95.1	23.0	4.9
ViT-Slimming (Chavan et al., 2022)	77.9	94.1	11.4	2.3
ViT-Slimming* (Chavan et al., 2022)	77.1	93.6	11.4	2.3
EViT (Liang et al., 2022)	78.5	94.2	22.0	2.3
A-ViT (Yin et al., 2022)	78.6	-	22.0	3.6
DeiT with UPop$_{1.11\times}$ (Ours)	81.1$_{\uparrow 1.2}$	95.4$_{\uparrow 0.4}$	19.9$_{\downarrow 10\%}$	4.1$_{\downarrow 11\%}$
DeiT with UPop$_{1.25\times}$ (Ours)	80.8$_{\uparrow 0.9}$	95.4$_{\uparrow 0.4}$	17.8$_{\downarrow 19\%}$	3.7$_{\downarrow 20\%}$
DeiT with UPop$_{1.42\times}$ (Ours)	80.2$_{\uparrow 0.3}$	95.1$_{\uparrow 0.1}$	15.7$_{\downarrow 29\%}$	3.2$_{\downarrow 30\%}$
DeiT with UPop$_{1.67\times}$ (Ours)	79.6$_{\downarrow 0.3}$	94.8$_{\downarrow 0.2}$	13.5$_{\downarrow 39\%}$	2.8$_{\downarrow 39\%}$
DeiT with UPop$_{2.00\times}$ (Ours)	78.9$_{\downarrow 1.0}$	94.6$_{\downarrow 0.4}$	11.4$_{\downarrow 48\%}$	2.3$_{\downarrow 50\%}$

C.5. Experiments on the Image Segmentation Task

Table 17: Compress Segmenter on the ADE20k dataset. The units of Params and FLOPs are M and G, respectively. The SS and MS mean single-scale and multi-scale testing for the mIoU metric, respectively. With and Without superscript * means CNN-based and Transformer-based models, respectively.

Approach	mIoU (SS)	mIoU (MS)	Params	FLOPs
Segmenter (Strudel et al., 2021)	45.3	46.9	26.4	38.6
Swin Transformer (Liu et al., 2021b)	44.5	46.1	60.0	236.0
SenFormer (Bousselham et al., 2021)	-	46.0	59.0	179.0
SegFormer (Xie et al., 2021)	46.5	47.5	27.5	62.4
PVT (Wang et al., 2021)	39.8	-	28.2	44.5
PVTv2 (Wang et al., 2022b)	45.2	-	29.1	45.8
DeiT III (Touvron et al., 2022)	45.6	46.8	41.7	-
DeeplabV3+ (ResNet-101)* (Chen et al., 2018)	45.5	46.4	63.0	255.0
OCRNet (HRNet-W48)* (Yuan et al., 2020)	-	45.7	70.5	164.8
ConvNeXt * (Liu et al., 2022c)	-	46.7	60.0	-
Segmenter with Mask-based Pruning (Chavan et al., 2022)	38.7	41.0	18.9	27.3
Segmenter with UPop$_{1.10\times}$ (Ours)	45.9$_{\uparrow 0.6}$	47.6$_{\uparrow 0.7}$	23.9$_{\downarrow 10\%}$	33.7$_{\downarrow 13\%}$
Segmenter with UPop$_{1.16\times}$ (Ours)	45.5$_{\uparrow 0.6}$	47.3$_{\uparrow 0.4}$	22.7$_{\downarrow 14\%}$	32.0$_{\downarrow 17\%}$
Segmenter with UPop$_{1.23\times}$ (Ours)	45.3$_{\uparrow 0.0}$	47.1$_{\uparrow 0.2}$	21.5$_{\downarrow 19\%}$	30.4$_{\downarrow 21\%}$
Segmenter with UPop$_{1.39\times}$ (Ours)	44.4$_{\downarrow 0.9}$	46.3$_{\downarrow 0.6}$	18.9$_{\downarrow 28\%}$	27.6$_{\downarrow 29\%}$

We also conduct experiments on the unimodal Segmenter-S model and ADE20k dataset for the semantic segmentation task as shown in Table 17, where we compare Segmenter pruned by UPop with the uncompressed Segmenter and other models with similar params/FLOPs to our pruned models. Furthermore, we also compare the performance of UPop on different unimodal tasks. Experimental results demonstrate:

- When comparing the model pruned by UPop with the uncompressed model, UPop can achieve more than $1.2\times$ loss-free compression and around $1.4\times$ compression with less than 1% mIoU loss for both single-scale and multi-scale testing.

- When comparing the model pruned by UPop with other models with similar params/FLOPs, UPop can achieve very competitive performance under Performance-Parameters and Performance-FLOPs trade-off constraints. For example, the $1.1\times$ compressed model can outperform all other models on multi-scale testing, and achieve a second place on single-scale testing by only using 54% FLOPs of the first place model.
- When comparing UPop's performance on the semantic segmentation task with on the image classification task (refers to Appendix Table 16), UPop can achieve around $1.5\times$ loss-free compression and $2\times$ compression with no more than 1% accuracy loss on the image classification task. We believe this ratio gap should be attributed to the intrinsic properties of different tasks. For example, classification models have more redundancy since many pixels (*e.g.*, background pixels) are unimportant for classification results, and therefore classification models can be pruned more easily. On the other hand, segmentation models have less redundancy since the model is expected to output the corresponding category for each pixel, and therefore pruning segmentation models is more difficult.