UPOP: UNIFIED AND PROGRESSIVE PRUNING FOR COMPRESSING VISION-LANGUAGE TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Data from the real world contains a vast amount of multimodal information, among which vision and language are the two most representative modalities. Moreover, in recent years, 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) that compresses vision-language Transformers via pruning. UPop incorporates 1) unifiedly searching multimodal subnets in a continuous optimization space from the original model; 2) progressively retraining the subnet while searching. Specifically, to ease the progress of pruning, we design *Unified Pruning* to automatically assign the appropriate pruning ratio to each compressible component, which comprises Self-Attentions, Cross-Attentions, and MLPs in both vision and language branches, instead of manually assigning each component a pruning ratio. Furthermore, to attain a higher compression ratio, we propose *Progressive Pruning* to maintain convergence between the search and retrain. In addition, UPop enables zero-cost subnet extraction after the search, and the searched subnet can even be used without further retraining. Experiments on multiple generative and discriminative vision-language tasks demonstrate the effectiveness and versatility of the proposed UPop. For example, we achieve $2\times$ compression on Image Caption with 0.5 SPICE drop and $4\times$ compression on VQA with 2.9% accuracy drop.

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 model compression exceedingly critical for deploying the increasingly heavier models on edge devices. There are lots of approaches that can be used to compress or accelerate deep learning models, such as weight sharing (Lan et al., 2019), low-rank factorization (Yu et al., 2017), pruning (He et al., 2017), quantization (Tao et al., 2022), parameter bootstrapping Chen et al. (2022), and knowledge distillation (Yang et al., 2022).

Recently, compression approaches dedicated to the Transformers (Vaswani et al., 2017) have also attracted much attention. According to the compressed components, these approaches can be summarized into two categories. The first category is token compression. By eliminating the number of input tokens, these approaches (Goyal et al., 2020; Rao et al., 2021) can reduce the FLOPs of models. The second category is model compression. By reducing the model size, these approaches (Wang et al., 2020; 2021) 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 prevalent situations where humans need to receive and process information from multiple modalities, among which vision and language are the two most representative ones. There are lots of multimodal tasks that have been extensively studied, including but not limited to Image Caption (Lin et al., 2014) that requires generating a text description for a given image, Text-Image Retrieval (Jia et al., 2015) that requires selecting one image from the candidate list based on a given text description, and NLVR2 (Suhr et al., 2018) that requires predicting whether a given sentence correctly describes a pair of given images.

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 representative multimodal models among them. 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 kinds of multimodal tasks.

Although compression on unimodal tasks has been widely investigated, how to compress multimodal models, especially vision-language Transformers, is still under-explored. Only a few works (Jin et al., 2021; Fang et al., 2021; Wang et al., 2022b) have paid attention to this problem, while all of them have been trying to conduct compression from knowledge distillation. In this paper, we propose a novel multimodal compression approach, Unified and Progressive Pruning (UPop).

A straightforward design of multimodal compression is to compress each modality separately via the unimodal compression 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. To overcome this shortcoming, we propose to unifiedly search on different modalities and different structures, which enables our approach to adaptively assign compression ratios among all compressible components given a total compression ratio. The second challenge is that the traditional two-stage compression paradigm (i.e., retraining after search) fails when the compression ratio is high. The significant gap of parameter weights between the searched model (i.e., model after the search phase) and the pruned subnet to be retrained severely degrades the final performance and even causes it hard to converge. Consequently, we propose an improved compression paradigm that conducts search and retraining progressively and simultaneously, which can effectively eliminate the gap mentioned above.

Our main contributions can be summarized as

- For the first time, we propose a multimodal pruning approach UPop for vision-language Transformers. UPop searches multimodal models in continuous optimization space, and a round of search can yield numerous multimodal subnets.
- The proposed *Unified Pruning* enables adaptive compression ratio assignment among all compressible components. *Progressive Pruning* proposes an improved compression paradigm that gains better convergence and supports higher compression ratios.
- As a deployment-friendly pruning approach, 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)). UPop is also evaluated on the unimodal task (e.g., image classification on ImageNet-1k Deng et al. (2009)).

2 RELATED WORK

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 interaction between vision and language modalities, 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), headsMichel et al. (2019), or channels (Zhu et al., 2021), and unstructured pruning (Yang et al., 2021; Chen et al., 2021b) that removes individual weights. The closest work to ours is ViT-Slimming (Chavan et al., 2022), a SOTA unimodal pruning approach applied to image classification. ViT-Slimming inserts trainable masks into the original model for searching subnets and retraining the searched subnets. Compared with ViT-Slimming, the proposed UPop is different in 3 aspects: 1) *Unified Pruning* enables adaptively instead of manually assigning the appropriate pruning ratio to each compressible component, and *Progressive Pruning* gains better convergence and performance at high compression ratios. 2) The subnets searched by UPop support real deployment without specific hardware requirements. 3) UPop focuses on the compression of vision-language tasks and can also be applied to unimodal tasks, like image classification.

3 METHODOLOGY

In this section, we first illustrate a straightforward approach for compressing vision-language Transformers, which we denote as *Multimodal Slimming* in Section 3.1. We then discuss its weakness and accordingly propose *Unified and Progressive Pruning* as illustrated in Figure 1 of Section 3.2. Necessary notations and their corresponding descriptions are listed in Table 1.

Table 1: 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
θ	Parameters of the orignal model	ζ	Parameters of the trainable mask
w	Regularization coefficient in searching	$\mathcal{F}_p$	p% compressed model $\mathcal{F}_p(x\|\theta,\zeta)$
$\alpha,\ \beta$	Learning rate during{search, retrain}	$T_{\{s,r\}}$	Iterations in {search, retrain} phase

3.1 PRELIMINARY

Multimodal Slimming straightforwardly applies the unimodal slimming to the multimodal scenario. Typically, we consider a multimodal model consisting of a ViT as vision encoder and a Bert as language encoder. *Multimodal Slimming* compresses ViT and Bert separately via the unimodal slimming approach, consisting of a *search* phase and a *retraining* phase. At the beginning of the search, trainable masks ζ are initialized to $\mathbf{1}$ and inserted into the Self-Attention, Cross-Attention, and MLP of each Transformer layer in each modality.

Search To search on Self-Attentions of Vision Transformer, denote the input of Self-Attention in the l^{th} layer as $a_l \in \mathbb{R}^{N\times D}$. Every head h in the Self-Attention will transform a_l into query $q_{l,h} \in \mathbb{R}^{N\times d}$, key $k_{l,h} \in \mathbb{R}^{N\times d}$, and value $v_{l,h} \in \mathbb{R}^{N\times d}$. Then trainable mask $\zeta_a^v \in \mathbb{R}^{L\times 1\times d}$ will be inserted into the original model, and the attention map of each head can be derived from

$$A_{l,h} = Softmax((q_{l,h} \odot \zeta^v_{a,l}) \times (k_{l,h} \odot \zeta^v_{a,l})^T/\sqrt{d}). \tag{1}$$

The output of each head h can be derived from

$$O_{l,h} = A_{l,h} \times (v_{l,h} \odot \zeta^v_{a,l}) \in \mathbb{R}^{N\times d}. \tag{2}$$

And the final output can be obtained by concatenating all heads. Note that more fine-grained mask with the shape of $\mathbb{R}^{L\times H\times d}$, like ViT-Slimming uses, results in pruned heads within a layer has different dimensions, and matrix computation of attention map becomes unfeasible on regular devices.

To search on MLPs of Vision Transformer, denote the input of MLP in the l^{th} layer as $m_l \in \mathbb{R}^{N\times D}$. Then trainable mask $\zeta_m^v \in \mathbb{R}^{L\times D_I}$, where D_I is the intermediate size of MLP, will be inserted, and the output of MLP can be derived from

$$a_{l+1} = f_2(f_1(m_l) \odot \zeta^v_{m,l}) \in \mathbb{R}^{N\times D}, \tag{3}$$

where f_1 and f_2 are the first and second fully connected layers in MLP. Search on Cross-Attentions and Language Transformer can be derived similary.

Besides, the ℓ_1-norm of masks ζ are added as additional loss items to drive the magnitude of masks smaller and smaller while searching:

$$\mathcal{L} = \mathcal{L}_{\mathcal{O}} + w_a \sum\nolimits_{\zeta_i\in\zeta_a} \|\zeta_i\|_1 + w_m \sum\nolimits_{\zeta_i\in\zeta_m} \|\zeta_i\|_1 \tag{4}$$

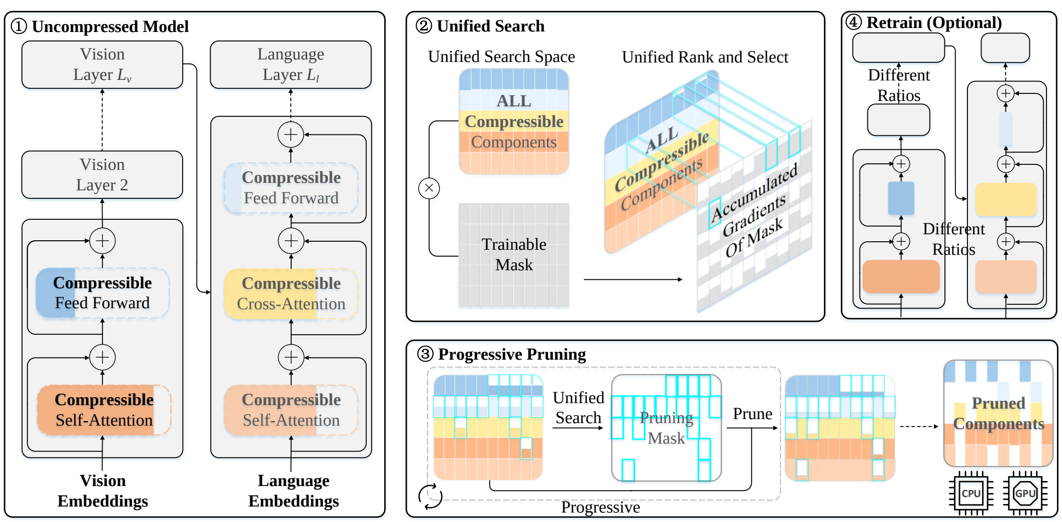
where $\mathcal{L}_{\mathcal{O}}$ is the loss to learn a multimodal model that typically contains contrastive loss and matching loss, and w_a and w_m are coefficients to balance the magnitude of loss items. It means that the model parameters θ and trainable masks ζ are optimized jointly in *search* phase.

Retraining After the search, the subnet can be pruned from the searched model based on mask ζ. The magnitude of the mask is used as the metric to evaluate the importance of corresponding neurons. Neurons in the compressible component 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 *Multimodal Slimming* is two-fold: 1) the mask $\zeta_i \in \zeta$ 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

Figure 1: Diagram of *Unified and Progressive Pruning* (UPop). (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.



3.2.1 UNIFIED PRUNING

The core idea of *Unified Pruning* is to unifiedly instead of separately searching on different modalities and structures. This enables *Unified Pruning* to adaptively assign the appropriate pruning ratio to each compressible component, instead of manually assigning each component a pruning ratio like *Multimodal Slimming* does.

Unified Search on Different Modalities *Unified Pruning* groups the pruning masks with respect to the same computation mechanisms. For typical vision-language Transformers, we divide the masks $\zeta = \{\zeta_{att}^v,\ \zeta_{att}^l,\ \zeta_{att}^c, \zeta_{mlp}^v,\ \zeta_{mlp}^l\}$ into two groups:

$$\zeta_a = \{\zeta_{att}^v,\ \zeta_{att}^l,\ \zeta_{att}^c\}, \quad \zeta_m = \{\zeta_{mlp}^v,\ \zeta_{mlp}^l\}. \tag{5}$$

One group ζ_a for different attention modules and another ζ_m for different MLP modules. The ranking and selection of masks are performed within each group.

Instead of searching on each $\zeta_i \in \zeta$ separately as *Multimodal Slimming* does:

$$M_i \leftarrow \text{TopKMask}({\zeta_i}^{(T_s)},\ p \cdot \text{Size}(\zeta_i)) \ \text{ for } \ \zeta_i \in \zeta, \tag{6}$$

where M_i is a binary mask used for pruning components of the subnet from the searched model. M_i is obtained by ranking and binarizing trainable mask ζ_i at the final iteration T_s, which keeps the most important $p \cdot \text{Size}(\zeta_i)$ parameters.

Unified Pruning searches on different modalities within each group which ranks important weights across different components.:

$$M_a \leftarrow \text{TopKMask}(\{{\zeta_i}^{(T_s)} | \zeta_i \in\ \zeta_a\},\ p \cdot \text{Size}(\zeta_a)), \tag{7}$$

$$M_m \leftarrow \text{TopKMask}(\{{\zeta_i}^{(T_s)} | \zeta_i \in\ \zeta_m\},\ p \cdot \text{Size}(\zeta_m)), \tag{8}$$

where M_a and M_m are binary masks used for pruning Attention and MLP structures, respectively. M_a and M_m are obtained by ranking and binarizing the corresponding trainable masks ζ_a and ζ_m at the final iteration T_s of search phase, respectively. Unified search on different modalities enables *Unified Pruning* to automatically assign the appropriate pruning ratio to each modality within the structures with the same computation mechanisms.

Unified Search on Different Structures We notice that simply uniting different structures degrades performance, and the reason why simple union fails is that the magnitude of the learned masks ζ_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 ζ_i used for different structures can be comparable. For the simplicity of implementation, we individually transform the mean and standard deviation of magnitudes distributions of different structures' mask to the 0 and 1 by z-score standardization, respectively.:

$$\zeta_a^{(T_s)} \leftarrow \frac{\zeta_a^{(T_s)} - \mathbb{E}[\zeta_a^{(T_s)}]}{\sqrt{\mathbb{E}[[\zeta_a^{(T_s)} - \mathbb{E}[\zeta_a^{(T_s)}]]^2]}}, \quad \zeta_m^{(T_s)} \leftarrow \frac{\zeta_m^{(T_s)} - \mathbb{E}[\zeta_m^{(T_s)}]}{\sqrt{\mathbb{E}[[\zeta_m^{(T_s)} - \mathbb{E}[\zeta_m^{(T_s)}]]^2]}}. \tag{9}$$

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

$$M \leftarrow \text{TopKMask}(\{{\zeta_i}^{(T_s)} | \zeta_i \in\ \zeta\},\ p \cdot \text{Size}(\zeta)), \tag{10}$$

where M is a binary mask used for pruning all compressible components, and M is obtained by ranking and binarizing the whole trainable masks ζ at the final iteration T_s of the search phase. Unified search on different modalities further enables *Unified Pruning* to automatically assign appropriate pruning ratios to all compressible components.

3.2.2 Progressive Pruning

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

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

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

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

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

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

where p_t is the current compression ratio when the iteration number is t. And the update strategy for optimizing ζ in each iteration of the search phase can be written as

$$\zeta^{(t+1)} \leftarrow (1 - M_i^t) + (1 - \frac{p_t}{p})M_i^t, \tag{13}$$

which ensures that as p_t progressively increases to p, each magnitude of mask ζ 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.

Algorithm 1 UPop: Unified and Progressive Pruning

Input: $\zeta,\ \zeta_a,\ \zeta_m,\ \theta,\ \mathcal{F},\ p,\ T_s,\ T_r,\ \alpha,\ \beta$

1 **for** $t \leftarrow 0$ **to** $T_r - 1$ **do**

2   **if** $t < T_s$ **then**

3         $\mathcal{L} \leftarrow \mathcal{L}_{\mathcal{O}} + w_a \sum_{\zeta_i \in \zeta_a} \|\zeta_i\|_1 + w_m \sum_{\zeta_i \in \zeta_m} \|\zeta_i\|_1$

4         $\theta^{(t+1)} \leftarrow \theta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \mathcal{L}(\theta^{(t)}, \zeta^{(t)})$

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

6         $G_a^{(t)} \leftarrow \frac{G_a^{(t)} - \mathbb{E}[G_a^{(t)}]}{\sqrt{\mathbb{E}[[G_a^{(t)} - \mathbb{E}[G_a^{(t)}]]^2]}}, \quad G_m^{(t)} \leftarrow \frac{G_m^{(t)} - \mathbb{E}[G_m^{(t)}]}{\sqrt{\mathbb{E}[[G_m^{(t)} - \mathbb{E}[G_m^{(t)}]]^2]}}$

7         $p_t = p\sqrt{(1 - \cos(\frac{\pi t}{T_s - 1}))\frac{1}{2}}$

8         $M^t \leftarrow \text{TopKMask}(\sum_{i=0}^{t} G^{(i)},\ p_t \cdot \text{Size}(\zeta))$

9         $\zeta^{(t+1)} \leftarrow (1 - M^t) + (1 - \frac{p_t}{p})M^t$

10         $\mathcal{F}_{p_{t+1}} \leftarrow \mathcal{F}_{p_t}(x|\theta^{(t+1)}, \zeta^{(t+1)})$

11   **else**

12         $\theta^{(t+1)} \leftarrow \theta^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \mathcal{L}_{\mathcal{O}}(\theta^{(t)})$

13 **return** $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x|\theta^{(T_r)})$

The proposed UPop combines unified search and progressive pruning as outlined in Algorithm 1. Line 2 $\sim$ 10 implements the search phase, and Line 12 implements the optional retraining phase. During the search phase, Line 3 computes the loss function consisting of the original loss and additional regularization items of trainable mask ζ. Line 4 normally updates the parameters θ of the original model with the original optimizer. Line 5 $\sim$ 9 updates the parameter of the trainable mask ζ with a customed optimizer. Specifically, Line 5 computes gradient of the loss function $\mathcal{L}$ with respect to the ζ. Line 6 conducts z-score standardization introduced in Section 3.2.1 to make G_a and G_m comparable. Line 7 computes the current compression ratio p_t to be achieved (p_t is a function of the current iteration number t, and a detailed discussion is provided in Appendix B.3). Line 8 generates the current pruning mask M_t by ranking and selecting the top p_t percent of positions based on accumulated gradient $\sum_{i=0}^{t} G^{(i)}$. Line 9 progressively compresses ζ based on M_t and accordingly Line 10 progressively compresses $\mathcal{F}_{p_t}$ to $\mathcal{F}_{p_{t+1}}$. The search phase ends after T_s cycles of Line 3 $\sim$ 10. After the search phase, Line 12 provides an optional retrain phase to further finetune the pruned subnet by the normal optimizer of the original model.

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. In addition, due to space constraints, we provide more ablation studies of the proposed *Unified and Progressive Pruning*, and the application on the unimodal classification task in the Appendix B.

4.1 COMPRESSION EXPERIMENTS ON THE VISUAL REASONING TASK

NLVR2 is a typical binary classification visual reasoning task with two images and a text description as inputs. To quantitatively evaluate the proposed UPop, we compress the BLIP model fine-tuned

Table 2: Compression results on the NLVR2. Bold indicates the best performance at the same compression ratio. The "Reduce" column indicates the compression times. The marker ✓ or ✗ indicates whether the model converges at the current compression times.

Approach	Reduce	Status	Dev Acc(↑)	Test Acc(↑)	Params(M)	FLOPs(G)
Uncompressed	1×	✓	82.48	83.08	259.45	132.54
Multimodal Slimming	2×	✓	75.74	76.44	146.18	66.88
	3×	✗	✗	✗	✗	✗
Unified Pruning (Section 3.2.1)	2×	✓	79.50	80.32	149.90	95.01
	3×	✓	71.25	71.66	106.33	68.19
	4×	✗	✗	✗	✗	✗
Unified and Progressive Pruning (Section 3.2)	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**	$54.48_{\downarrow 79\%}$	$19.08_{\downarrow 86\%}$

on this task at a ratio of 2, 3, 4, 5, and 10 times, respectively. 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 $\zeta = \{\zeta_a^v, \zeta_m^v, \zeta_a^l, \zeta_m^l, \zeta_a^{c0}, \zeta_a^{c1}\}$. We compress the original model with three aforementioned multimodal compression approaches, Multimodal Slimming, Unified Pruning, and UPop, respectively. Experimental results are shown in Table 2. It is worth noting that at a compression ratio of N times, the total number of parameters of the compressed model 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 of the image encoder, the word embedding module of the text encoder, 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 $2\times$ compression ratio, Table 2 shows that compared to the Multimodal Slimming, 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 $3\times$ compression ratio, while Multimodal Slimming does not. We provide visualization results and more analyses in Appendix B.1 and B.2

4.3 Effect of Progressive Pruning

As shown in Table 2, 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 Multimodal Slimming and Unified Pruning fail.

To further illustrate how Progressive Pruning strengthens the convergence capability of the compressed model, we compare the performance of pruned subnets in the situation of search without any retraining or search with only one epoch retraining. Tabel 3 shows that the model compressed by UPop can converge without any retraining while the other two compression approaches fail. Furthermore, with only one epoch retraining, the model compressed by UPop converges at significantly superior performance to the other two approaches. The experiments in Tabel 3 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 Effect of Unified and Progressive Pruning

Unified Pruning and Progressive Pruning boost the performance of Multimodal Slimming in two aspects, respectively. At the same and relatively low compression ratio, Unified Pruning gains significant performance improvements by adaptively assigning appropriate compression ratios among

Table 3: Performance of the compressed model while searching without any retraining or with only one epoch retraining.

Approach	Reduce	Search Only		One Epoch Retrain	
		Dev Acc(↑)	Test Acc(↑)	Dev Acc(↑)	Test Acc(↑)
Multimodal Slimming	2×	✗	✗	62.82	63.35
Unified Pruning	2×	✗	✗	75.42	75.30
UPop	2×	**76.89**	**77.84**	**79.08**	**80.08**

all compressible components. As the compression ratio rises, the gap in parameter weights between the searched model and the pruned subnet to be retrained becomes larger and larger. Therefore the compressed model will be increasingly difficult to converge. In such a situation, Progressive Pruning plays the role of maintaining the convergence capability of the compressed model. Combined Progressive Pruning with Unified Pruning, the UPop gains the ability to achieve better performance at the same compression ratio and push the limit of compression ratio to a greater extent.

4.5 Compression 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 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 ζ corresponding to the compressible components on this model is $\zeta = \{\zeta_a^v,\ \zeta_m^v,\ \zeta_a^l,\ \zeta_m^l,\ \zeta_a^c\}$. Table 4 shows that UPop also achieves superior performance on the Image Caption task.

Table 4: Compression results on the Image Caption task and the Visual Question Answering task. The higher the CIDEr, SPICE, test-dev, and test-std, the better the model performance. 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
Multimodal Slimming	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 (Section 3.2.1)	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 (Section 3.2)	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**	$133.3_{\downarrow 63\%}$	$62.3_{\downarrow 67\%}$

4.6 Compression Experiments on the Visual Question Answering Task

We also conducted experiments on the Visual Question Answering 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 ζ corresponding to the compressible components on this model is $\zeta = \{\zeta_a^v,\ \zeta_m^v,\ \zeta_a^{l,en},\ \zeta_m^{l,en},\ \zeta_a^{l,de},\ \zeta_m^{l,de}\}$. Table 4 shows the improved performance of UPop on the Visual Question Answering task.

4.7 Compression Experiments on the Image-Text Retrieval Task

We also conducted experiments on the Image-Text 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 ζ

corresponding to the compressible components on this model is $\zeta = \{\zeta_a^v,\ \zeta_m^v,\ \zeta_a^l,\ \zeta_m^l,\ \zeta_a^c\}$. Table 5 shows the improved performance of UPop on the Image-Text Retrieval task.

Table 5: Compress BLIP on the COCO and Flickr30K datasets of the Image-Text Retrieval task. The higher the R@1, R@5, and R@10, the better the model performance. 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
	Multimodal Slimming	2×	61.7	85.0	91.1	46.0	73.2	82.6	249.5	77.3
		4×	✗	✗	✗	✗	✗	✗	✗	✗
	Unified Pruning (Section 3.2.1)	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 (Section 3.2)	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
	Multimodal Slimming	2×	78.9	92.7	95.5	63.8	85.1	90.1	249.3	77.2
		4×	✗	✗	✗	✗	✗	✗	✗	✗
	Unified Pruning (Section 3.2.1)	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 (Section 3.2)	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$_{\downarrow 67\%}$

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 6 shows that UPop is able to achieve comparable effectiveness to BLIP on CLIP. It is worth noting that we use the momentum distillation method proposed by BLIP to finetune CLIP on the Image-Text Retrieval task. Due to the introduction of momentum models, the number of parameters and FLOPs in Table 6 are approximately twice as high as the original CLIP, respectively.

Table 6: Compress CLIP on the COCO and Flickr30K datasets of the Image-Text Retrieval task. Notations are the same as in Table 5.

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 (Section 3.2)	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 (Section 3.2)	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$_{\downarrow 74\%}$

5 CONCLUSION

This paper proposes a multimodal compression approach, Unified and Progressive Pruning (UPop), for vision-language Transformers. UPop unifiedly searches on all compressible components, which consists 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 conducts search and retraining progressively and simultaneously, which effectively strengthens the convergence capability of the compressed model and enables higher compression ratios. Finally, UPop is a practically deployable compression approach that physically extracts the pruned subnet from the original model.

REFERENCES

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433, 2015.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4931–4941, 2022.

Boyu Chen, Peixia Li, Chuming Li, Baopu Li, Lei Bai, Chen Lin, Ming Sun, Junjie Yan, and Wanli Ouyang. 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.

Dongsheng Chen, Chaofan Tao, Lu Hou, Lifeng Shang, Xin Jiang, and Qun Liu. Litevl: Efficient video-language learning with enhanced spatial-temporal modeling. *arXiv preprint arXiv:2210.11929*, 2022.

Tianlong Chen, Yu Cheng, Zhe Gan, Lu Yuan, Lei Zhang, and Zhangyang Wang. Chasing sparsity in vision transformers: An end-to-end exploration. *Advances in Neural Information Processing Systems*, 34:19974–19988, 2021b.

Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. 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.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

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

Zhiyuan Fang, Jianfeng Wang, Xiaowei Hu, Lijuan Wang, Yezhou Yang, and Zicheng Liu. Compressing visual-linguistic model via knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1428–1438, 2021.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. Power-bert: Accelerating bert inference via progressive word-vector elimination. In *International Conference on Machine Learning*, pp. 3690–3699. PMLR, 2020.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

Yan Huang, Wei Wang, and Liang Wang. 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.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pp. 4904–4916. PMLR, 2021.

Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. Guiding long-short term memory for image caption generation, 2015. URL https://arxiv.org/abs/1509.04942.

Woojeong Jin, Maziar Sanjabi, Shaoliang Nie, Liang Tan, Xiang Ren, and Hamed Firooz. Msd: Saliency-aware knowledge distillation for multimodal understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3557–3569, 2021.

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

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

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

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *arXiv preprint arXiv:2201.12086*, 2022.

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pp. 121–137. Springer, 2020.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

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

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

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.

Jing Shao, Siyu Chen, Yangguang Li, Kun Wang, Zhenfei Yin, Yinan He, Jianing Teng, Qinghong Sun, Mengya Gao, Jihao Liu, et al. Intern: A new learning paradigm towards general vision. *arXiv preprint arXiv:2111.08687*, 2021.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Vitas: vision transformer architecture search. In *European Conference on Computer Vision*, pp. 139–157. Springer, 2022.

Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018.

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

Chaofan Tao, Lu Hou, Wei Zhang, Lifeng Shang, Xin Jiang, Qun Liu, Ping Luo, and Ngai Wong. Compression of generative pre-trained language models via quantization. *arXiv preprint arXiv:2203.10705*, 2022.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

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

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

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2140–2151, 2021.

Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022a.

Zhecan Wang, Noel Codella, Yen-Chun Chen, Luowei Zhou, Xiyang Dai, Bin Xiao, Jianwei Yang, Haoxuan You, Kai-Wei Chang, Shih-fu Chang, et al. Multimodal adaptive distillation for leveraging unimodal encoders for vision-language tasks. *arXiv preprint arXiv:2204.10496*, 2022b.

Huanrui Yang, Hongxu Yin, Pavlo Molchanov, Hai Li, and Jan Kautz. Nvit: Vision transformer compression and parameter redistribution. *arXiv preprint arXiv:2110.04869*, 2021.

Zhendong Yang, Zhe Li, Mingqi Shao, Dachuan Shi, Zehuan Yuan, and Chun Yuan. Masked generative distillation. *arXiv preprint arXiv:2205.01529*, 2022.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 21–29, 2016.

Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. 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.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. 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.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. 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.

Mingjian Zhu, Yehui Tang, and Kai Han. Vision transformer pruning. *arXiv preprint arXiv:2104.08500*, 2021.

A Implementation Details

A.1 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.2 Deployability

UPop is a deployable pruning approach that allows the compressed model to be physically extracted from the original model and can further be deployed in real scenarios, whiles some pruning approaches are non-deployable. For example, 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 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, which enables UPop to support real deployment without specific hardware requirements.

A.3 Hyperparameter Settings

Table 7: Training hyperparameters for compressing BLIP-based models.

Hyperparameters	BLIP-NLVR	BLIP-Caption	BLIP-VQA	BLIP-Retrieval	
	NLVR2	COCO	VQAv2	COCO	Flickr30K
Optimizer	AdamW				
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
Rtrain epochs	15	5	10	6	12
Rrtrain 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 8: Training hyperparameters for compressing CLIP and DeiT.

Hyperparameters	CLIP		DeiT
	COCO	Flickr30K	ImageNet
Optimizer	AdamW		
AdamW β	(0.9, 0.999)		
Weight decay	0.2	0.2	0.05
Batch size	256	256	4096
Search epochs	6	12	60
Search LR	1e-5	1e-5	8e-4
Rtrain epochs	6	12	300
Rrtrain LR	1e-5	1e-5	8e-4
Search LR schedule	N/A		
Retrain LR schedule	CosineLRScheduler (Loshchilov & Hutter, 2016)		
Data augmentation	RandomAugment (Cubuk et al., 2020)		RepeatedAugment (Touvron et al., 2021)

Table 9: Structure hyperparameters for all models used in our experiments. "*" indicates 2 Transformers share parameters.

Model	Input resolution	Vision Transformer				Language Transformer			
		number	layers	width	heads	number	layers	width	heads
BLIP-NLVR	384×384	2*	12	768	12	1	12	768	12
BLIP-Caption	384×384	1	12	768	12	1	12	768	12
BLIP-VQA	480×480	1	12	768	12	2	12	768	12
BLIP-Retrieval	384×384	2	12	768	12	2	12	768	12
CLIP	336×336	2	24	1024	16	2	12	768	12
DeiT	224×224	1	12	384	6	0	-	-	-

A.4 IMPLEMENTATION OF MULTIMODAL SLIMMING

Algorithm 2 Multimodal Slimming

Input: $\zeta,\ \zeta_a,\ \zeta_m,\ \theta,\ \mathcal{F},\ p,\ T_s,\ T_r,\ \alpha,\ \beta$

for $t \leftarrow 0$ **to** $T_s - 1$ **do**

  $\mathcal{L} \leftarrow \mathcal{L}_{\mathcal{O}} + w_a \sum_{\zeta_i \in \zeta_a} \|\zeta_i\|_1 + w_m \sum_{\zeta_i \in \zeta_m} \|\zeta_i\|_1$

  $\theta^{(t+1)} \leftarrow \theta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \mathcal{L}(\theta^{(t)}, \zeta^{(t)})$

  $\zeta^{(t+1)} \leftarrow \zeta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\zeta \mathcal{L}(\theta^{(t)}, \zeta^{(t)})$

for $\zeta_i \in \zeta$ **do**

  $M_i \leftarrow \text{TopKMask}({\zeta_i}^{(T_s)},\ p \cdot \text{Size}(\zeta_i))$

$\hat{\theta} \leftarrow \{\theta_i^{(T_s)} | M_i = 1\}, \quad \mathcal{F}_p \leftarrow \mathcal{F}(x|\hat{\theta}, \zeta^{(T_s)})$

for $t \leftarrow 0$ **to** $T_r - 1$ **do**

  $\hat{\theta}^{(t+1)} \leftarrow \hat{\theta}^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_{\hat{\theta}} \mathcal{L}_{\mathcal{O}}(\hat{\theta}^{(t)})$

return $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x|\hat{\theta}^{(T_r)})$

A.5 IMPLEMENTATION OF UNIFIED PRUNING

Algorithm 3 Unified Pruning

Input: $\zeta,\ \zeta_a,\ \zeta_m,\ \theta,\ \mathcal{F},\ p,\ T_s,\ T_r,\ \alpha,\ \beta$

for $t \leftarrow 0$ **to** $T_s - 1$ **do**

  $\mathcal{L} \leftarrow \mathcal{L}_{\mathcal{O}} + w_a \sum_{\zeta_i \in \zeta_a} \|\zeta_i\|_1 + w_m \sum_{\zeta_i \in \zeta_m} \|\zeta_i\|_1$

  $\theta^{(t+1)} \leftarrow \theta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \mathcal{L}(\theta^{(t)}, \zeta^{(t)})$

  $\zeta^{(t+1)} \leftarrow \zeta^{(t)} - \alpha \frac{1}{n} \sum_{i=1}^{n} \nabla_\zeta \mathcal{L}(\theta^{(t)}, \zeta^{(t)})$

$\zeta_a^{(T_s)} \leftarrow \frac{\zeta_a^{(T_s)} - \mathbb{E}[\zeta_a^{(T_s)}]}{\sqrt{\mathbb{E}[[\zeta_a^{(T_s)} - \mathbb{E}[\zeta_a^{(T_s)}]]^2]}}, \quad \zeta_m^{(T_s)} \leftarrow \frac{\zeta_m^{(T_s)} - \mathbb{E}[\zeta_m^{(T_s)}]}{\sqrt{\mathbb{E}[[\zeta_m^{(T_s)} - \mathbb{E}[\zeta_m^{(T_s)}]]^2]}}$

$M \leftarrow \text{TopKMask}(\zeta^{(T_s)},\ p \cdot \text{Size}(\zeta))$

$\hat{\theta} \leftarrow \{\theta_i^{(T_s)} | M_i = 1\}, \quad \mathcal{F}_p \leftarrow \mathcal{F}(x|\hat{\theta}, \zeta^{(T_s)})$

for $t \leftarrow 0$ **to** $T_r - 1$ **do**

  $\hat{\theta}^{(t+1)} \leftarrow \hat{\theta}^{(t)} - \beta \frac{1}{n} \sum_{i=1}^{n} \nabla_{\hat{\theta}} \mathcal{L}_{\mathcal{O}}(\hat{\theta}^{(t)})$

return $\mathcal{F}^\star \leftarrow \mathcal{F}_p(x|\hat{\theta}^{(T_r)})$

B SUPPLEMENTARY EXPERIMENTS AND ANALYSES

B.1 RETAINED COMPONENTS IN THE COMPRESSION MODEL

Figure 2: The proportion of all compressible components retained in the compression model. 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 ζ_i, and the number in cells represents the retained proportion of a certain component's certain layer.

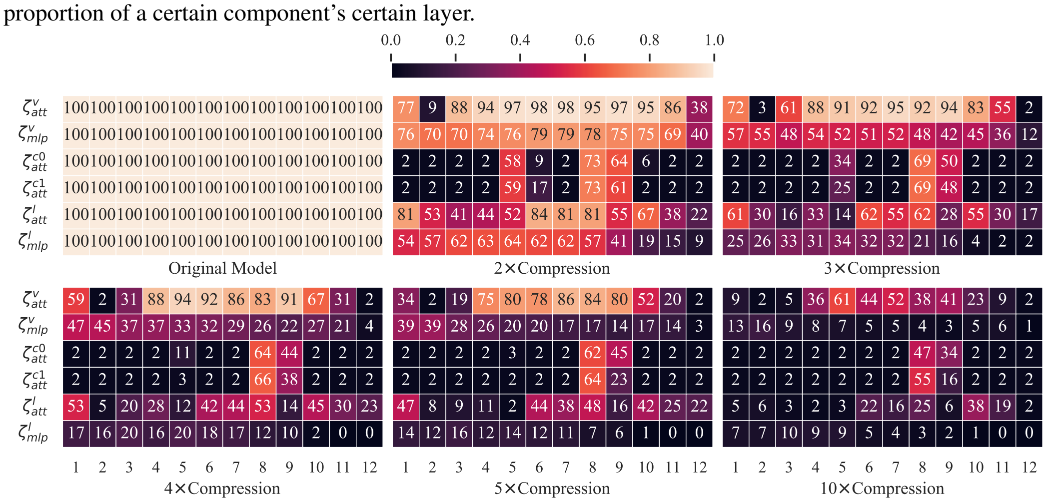
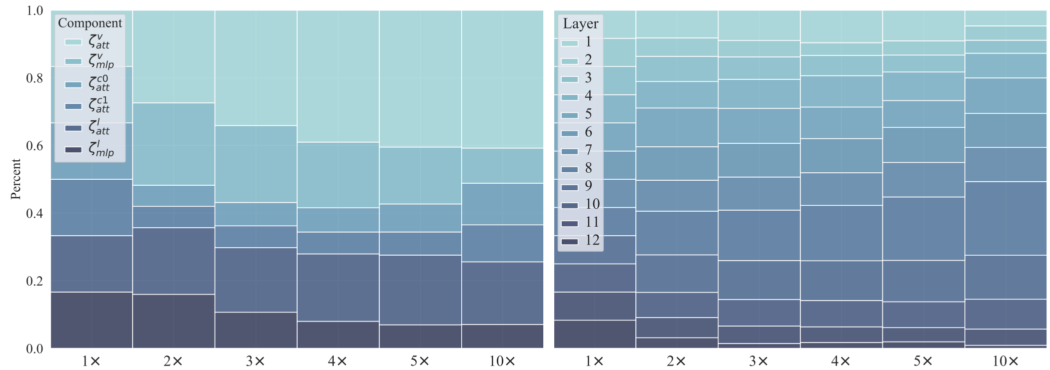


Figure 2 visualizes the proportion of all compressible components retained in the compression model. It can be observed from the figure that retained proportion of each compressible component has significantly different trends as the compression ratio increases. Moreover, there are obviously unbalanced compression assignments in different layers at different compression ratios.

B.2 VARIATION OF COMPRESSIBLE COMPONENTS AND LAYERS

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 3. 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

Figure 3: The left subfigure: variation of compressible components as the compression ratio increases. The right subfigure: variation of layers as the compression ratio increases.



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 3 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.

B.3 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

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

is a function that satisifies all requirements.

Table 10: Study on how the update strategy of compression ratio p_t affects the model 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**
$p\sqrt{(1-\cos(\frac{\pi t}{T_s-1}))\frac{1}{2}}$	80.33	**81.13**

Table 10 shows the performance of the 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 $p\sqrt{(1-\cos(\frac{\pi t}{T_s-1}))\frac{1}{2}}$. 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.

B.4 Study on the frequency of Updating Compression Mask ζ

We also explore how the frequency of updating mask ζ affects the model performance. Experimental results on the BLIP-NLVR model are reported in Table 11. Update compression mask ζ 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 11 that updating the ζ too frequently causes the compressed model to tend to overfit on the validation set.

Table 11: Study on how the frequency of updating compression mask ζ 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 ζ each time the model parameters θ are updated, while frequency 100 means updating ζ once every 100 times the model parameters θ are updated. Consequently, frequency 50 is adopted by us, 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 in different situations. Empirically, setting the frequency to the number of iterations corresponding to the 1% compression ratio will be appropriate. For example, if we aim to accomplish 50% compression ratio in 10,00 iterations, then a frequency about $1000 \times \frac{1}{50} = 20$ is recommended.

B.5 Compression 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 reported in Table 12 and illustrated in Figure 4, we conduct unimodal DeiT (Touvron et al., 2021) compression on the ImageNet dataset, and UPop can also achieve competitive performance compared to other unimodal compression SOTA approaches.

Table 12: Compress DeiT on the ImageNet dataset. The units of Params and FLOPs are M and G, respectively. "*" 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
$\text{UPop}_{1.11\times}$	81.1$_{\uparrow 1.2}$	95.4$_{\uparrow 0.4}$	19.9$_{\downarrow 10\%}$	4.1$_{\downarrow 11\%}$
$\text{UPop}_{1.25\times}$	80.8$_{\uparrow 0.9}$	95.4$_{\uparrow 0.4}$	17.8$_{\downarrow 19\%}$	3.7$_{\downarrow 20\%}$
$\text{UPop}_{1.42\times}$	80.2$_{\uparrow 0.3}$	95.1$_{\uparrow 0.1}$	15.7$_{\downarrow 29\%}$	3.2$_{\downarrow 30\%}$
$\text{UPop}_{1.67\times}$	79.6$_{\downarrow 0.3}$	94.8$_{\downarrow 0.2}$	13.5$_{\downarrow 39\%}$	2.8$_{\downarrow 39\%}$
$\text{UPop}_{2\times}$	78.9$_{\downarrow 1.0}$	94.6$_{\downarrow 0.4}$	11.4$_{\downarrow 48\%}$	2.3$_{\downarrow 50\%}$

Figure 4: Comparison of DeiT compressed by various approaches listed in Table 12. The left subfigure illustrates the Accuracy-FLOPs trade-off, and the right subfigure illustrates the Accuracy-Parameter trade-off. 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.