Once-Tuning-Multiple-Variants: Tuning Once and Expanded as Multiple Vision-Language Model Variants

Chong Yu[1] Tao Chen[2,*] Zhongxue Gan[1,*]
[1]Academy for Engineering and Technology, Fudan University, Shanghai 200433, China
[2]School for Information Science and Technology, Fudan University, Shanghai 200433, China
21110860050@m.fudan.edu.cn; {eetchen, ganzhongxue}@fudan.edu.cn

Abstract

Vision-language model (VLM) is one of the most important models for multi-modal tasks. Real industrial applications often meet the challenge of adapting VLMs to different scenarios, such as varying hardware platforms or performance requirements. Traditional methods involve training or fine-tuning to adapt multiple unique VLMs or using model compression techniques to create multiple compact models. These approaches are complex and resource-intensive. This paper introduces a novel paradigm called Once-Tuning-Multiple-Variants (OTMV). OTMV requires only a single tuning process to inject dynamic weight expansion capacity into the original VLM structure. This tuned VLM can then be expanded into multiple variants tailored for different scenarios in inference. The tuning mechanism of OTMV is inspired by the mathematical series expansion theorem, which helps to reduce the parameter size and memory requirements while maintaining accuracy for VLM. Experiment results show that OTMV-tuned models achieve comparable accuracy to baseline VLMs across various visual-language tasks. The experiments also demonstrate the dynamic expansion capability of OTMV-tuned VLMs, outperforming traditional model compression and adaptation techniques in terms of accuracy and efficiency.

1. Introduction

Deep neural networks have been proven successful in many mono-modal tasks, both in the vision domain (such as image classification [18, 33], object detection [42, 60], and segmentation [27, 45]) and in the language domain (such as summarization [17], machine translation [24], question answering [34], information extraction and retrieval [32]). Based on these foundational mono-modal capacities, prior arts started to explore the potential of deep neural models on multi-modal tasks [20]. The combination of two critical modalities, vision and language, achieves considerable progress. These vision-language models [3, 4, 30, 53] jointly process information from these two modalities and uniformly utilize them to fulfill the complex tasks [13]. For instance, visual question answering [1] provides an image and the corresponding question as models' input and requires the model to give the correct answer.

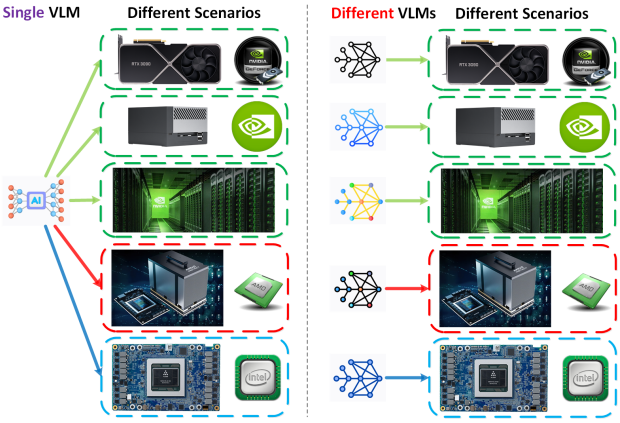

Figure 1. Comparison: Once-Tuning-Multiple-Variants (OTMV) (***Left***) only produces a single instance, while the typical manners (***Right***) all produce multiple different instances to meet the requirements of different scenarios.

In real-world applications, we always need to adapt the VLM model to different scenarios [30]. For example, we want to deploy the VLM model to different hardware platforms with quite different hardware features or software stacks. Another example is even deploying the VLM model on the same specific platform; we still want to adjust the VLM model itself to meet the different performance requirements in latency-sensitive or throughput-sensitive use cases. The typical manner is using multiple different VLM models to meet the requirements of different scenarios, respectively. Multiple different VLM models can be obtained using two tuning paradigms. One tuning paradigm introduces the differences in the training process, either in the

*Tao Chen and Zhongxue Gan are corresponding authors.

pre-training [12, 55] or downstream fine-tuning and adaption [8, 15, 19, 57] stages. The other tuning paradigm uses model compression techniques [7, 29, 52] to compress a single VLM model into multiple compact ones. On the other hand, we present a revolutionary paradigm called Once-Tuning-Multiple-Variants (OTMV). OTMV offers a game-changing solution, requiring only a single tuning of a VLM model to inject dynamic expansion capacity. The tuned VLM can then be expanded into multiple variants, perfectly tailored to meet the demands of different scenarios. This paradigm shift is illustrated in Figure 1, highlighting the contrast between OTMV and the traditional approach.

The naive VLM's adaptive capability to different scenarios is the poorest. Because the pre-training paradigm needs to train several unique VLMs to meet the various scenarios' requirements, the sum of parameter size is the largest, and the training cost is the highest among all the paradigms. The efficient fine-tuning, especially the parameter-efficient fine-tuning (PEFT) [9, 19], reduces the painful training consumption by introducing some low-rank adapters. However, different adapters also need to be trained for different scenarios. After fine-tuning, the adapter parameters will be merged with pre-trained VLM parameters to eliminate additional inference latency. So, the parameter size sum is still larger than a single naive pre-trained VLM. The compression paradigm starts from a pre-trained VLM. Then, it applies the model compression process, such as pruning, quantization, or low-rank decomposition, to generate multiple unique VLMs to handle different scenarios. Each compressed VLM has a smaller parameter size than the naive pre-trained VLM. However, compressing the VLMs without introducing a remarkable accuracy drop is a nontrivial effort; some extra training or fine-tining is necessary for accuracy compensation. So, its training cost is slightly higher than that of the single naive VLM. Moreover, the compression paradigm needs more extra deployment effort. The proposed OTMV paradigm only needs some extra lightweight training cost than the naive VLM, and there is no reliance on the model compression techniques, so there is no additional effort for deployment. OTMV will dynamically expand the model structure with the necessary layers among the original VLM model, so the parameter size is also lower than the baseline, i.e., a single naive VLM.

The fundamental principle for the training process of VLMs is finding a proper approximation to estimate the correlation in image-text pairs. The approximation function is determined by the model's weight tensors in VLM. In mathematics, the series expansion [46, 50] is a concept that expresses a function as an infinite sum of terms. If it is limited to expanding a finite number of terms, thus yielding an approximation of the function. **Motivated by this similarity**, we borrow the series expansion from mathematics into the VLM. We regard the individual weight tensors as the terms in series expansion. The typical VLM can dynamically expand or erase the individual weight tensors from its original structure by introducing a unique training mechanism. Similar to series expansion, the more weight tensors are expanded, the more accurate the approximation, i.e., the accuracy of the VLM is higher. At the same time, if only a small amount of the weight tensors are kept and expanded, OTMV also guarantees that this VLM variant has the least accuracy gap with the original VLM model.

Our main contributions include:

- We propose the brand-new tuning paradigm, OTMV, with the least extra training cost and effort for deployment and good adaption to different scenarios.
- OTMV utilizes the mathematical series expansion theorem to improve VLMs and designs a unique training mechanism for VLMs to inject dynamic weight expansion capacity with least accuracy influence.
- OTMV is proven to boost the tuning workflow and efficiency on various vision-language benchmarking tasks against the state-of-the-art competitors.

2. Related work

2.1. Vision-language model

Because the capability of the vision-language models (VLM) highly relies on the image and language modal alignment [59], the typical learning paradigm of VLM starts from pre-training with large-scale image-text pairs [28, 41, 51, 58]. Then, the pre-trained foundation VLM is further fine-tuned [25, 30, 43, 53] with task-specific annotated datasets to fit the downstream tasks. There are also some prior arts [21, 54] to explore the potential to substitute the task-specific fine-tuning stage with the zero-shot prediction manner, i.e., applying the pre-trained VLM directly to downstream tasks without fine-tuning. As compensation, more image-text pairs with instructive annotations must be learned in the pre-training stage. However, the success of these two VLM learning paradigms is highly reliant on the pre-training stage, which requires a lot of training data and computation consumption.

2.2. Efficient fine-tuning for VLM

As a further improvement, there are prior efficient fine-tuning explorations to reduce the traditional fine-tuning paradigm cost. Instruction tuning improves tuning efficiency [2, 48] by adapting VLM to learn while following task-specific instructions. Unlike the prior paradigm, which updates all the parameters of the pre-trained VLM, parameter-efficient tuning [22, 49] freezes the pre-trained parameters and only trains a small number of task-specific addition parameters for each task. As the representative, LoRA [19] proposes merging the trainable parameters in the task-specific adapters with the original frozen pre-trained

weights to avoid introducing extra inference latency. Although these prior arts can solve the adaption requirements to multiple tasks, the fine-tuning costs ***still grow linearly with the number of deployment tasks***. For the LoRA-style paradigm [9, 19], it is not ***straightforward*** to batch inputs to different tasks because multiple different adapters are also needed. ***Extra deployment overhead*** is required to choose the LoRA adapters dynamically.

There are also some all-in-one trials. OFA network [5] reduces the neural architecture search (NAS) cost by decoupling training and search. It mainly focuses on convolution networks (CNN). Moreover, it ***relies on the model compression method***, like pruning, to progressively shrink the model size in depth, width, kernel size, and resolution. Slimmable network [56] is another width-adjustable all-in-one method. It is designed particularly for CNN structures, and thus, ***extra handling of the batch normalization*** module for various width sizes is a nontrivial effort. MatFormer [10] is a pre-training-only all-in-one solution based on summation loss for Transformer-based models. MatFormer works only along the width dimension of the feed-forward layers in Transformer blocks and ***cannot expand to more than a few sub-models***. Flextron [6] utilizes a nested elastic structure and an input-adaptive manner to route tokens through its sub-structures for improved deployment efficiency. It mainly focuses on the large language model, and the ***accuracy drops are still obvious*** when it uses a 50% dynamic elastic structure.

2.3. Model compression for VLM

In real-world vision-language applications, hardware memory, computation resources, and latency constraints are critical. So, the large VLMs and the typical learning paradigm both challenge deployment. Some prior works started applying model compression techniques to relieve challenges.

DistillVLM [14] demonstrates that knowledge distillation can effectively compress large VLMs into smaller, high-performing models. By addressing the alignment issues in the teacher and student models, the proposed DistillVLM shows substantial performance gains in image captioning and visual question answering tasks. On this basis, EfficientVLM [52] further improved as a two-stage framework. The first stage shrinks a pre-trained large VLM using knowledge distillation. In the second stage, a modal-adaptive pruning algorithm is introduced to infer the importance of vision and language modalities for different downstream tasks and adaptively prunes redundant neurons. UPop [44] offered another versatile solution for VLM compression. It first searches for multi-modal subnets within a continuous optimization space, optimizing the overall compression ratio without manual intervention. Then, UPop progressively searches and retrains the subnets, to achieve better performance and higher compression ratios by eliminating the weight gap between the searched model and the pruned subnet.

Although these prior arts partially relieve VLMs' deployment challenges, ***implementing the model compression is a nontrivial effort***. No matter the knowledge distillation, modal-adaptive pruning, or the subnets search, all add complexity in preparing the compact VLM models for deployment. What's worse, if we need to prepare different VLM variants to fit various scenarios, many compression processes cannot be reused, i.e., ***the model compression efforts will be doubled and redoubled***.

3. Once-Tuning-Multiple-Variants

3.1. OTMV workflow

The typical vision-language model's structure contains the two-way backbones [3]. The language backbone consists of word embedding to cast the input sequence tokens into the language latent space, then applying multiple language modality transformer blocks to learn language modality features further. The vision backbone takes the image patches as the input tokens, following the patch embedding and position embedding [11] to cast them into the vision latent space. The stacked vision modality transformer blocks will further extract the features of vision modality. Such a design can encourage the VLM model to learn more modality-specific information [53]. After the two-way modality-specific backbones, several vision-language transformer blocks are connected to learn the alignment between the vision and language modalities and enable deep fusion for multiple multi-modal tasks.

The mathematical series expansion theorem [46] proves that the infinite sum of the series can express a nonlinear function and that the expansion of the first few finite series terms is a good approximation of the original function in most cases. Motivated by this concept, we choose several key weight tensors among all transformer blocks (***in analogy with the series first-order term***). The other weight tensors are substituted with the key weight tensors multiplied with the corresponding learnable OTMV coefficients (***in analogy with the series expanded high-order terms***) to ***approximate*** the original naive VLM model mutually. The extensive explanation of why the OTMV expanded VLM variants can approximate the original naive VLM with a negligible accuracy gap is given in the **Appendix**.

Because most typical VLM models are based on transformer structures, we use a transformer-based VLM as an example. Without loss of generality, a given VLM model can be divided into three parts: the vision modality backbone, the language modality backbone, and the vision-language cross-modality transformer part.

Assume there are M vision modality transformer blocks in the vision modality backbone. We regard the weight ten-

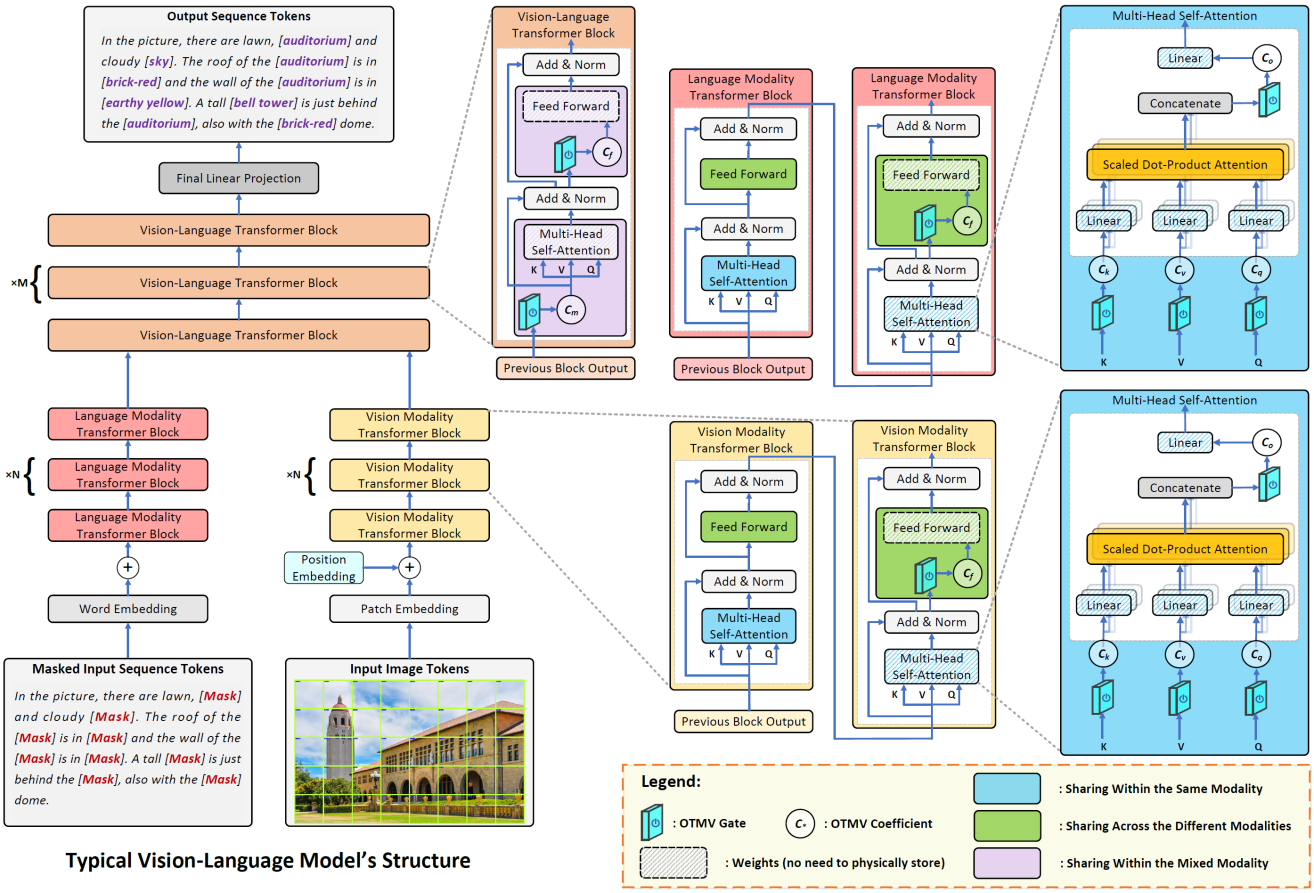

Figure 2. Once-Tuning-Multiple-Variants (OTMV) workflow. OTMV first chooses the key weight tensors, analogous to the first-order term in the mathematical series expansion theorem. The other weight tensors share the first-order weights with identical values and multiply them with one learnable scalar value, the OTMV coefficient, for compensation. So, there is ***no need to be physically stored in hardware memory***. An OTMV gate is attached to each weight tensor without physically stored values. The OTMV gate determines whether to expand and calculate the following weight tensor or bypass it. This is the critical component to guarantee that the OTMV workflow only requires one tuning process, while it can be expanded into multiple VLM variants to meet the different deployment scenarios. We only share the weight tensors ***within*** the same modality transformer blocks for the multi-head attention layers. For the feed-forward layers, we share the weight tensors ***across*** the language-modality and vision-modality transformer blocks. For the vision-language transformer blocks for modality alignment, both the multi-head attention and feed-forward layers are shared ***within*** these multi-modality transformer blocks.

sors (including query, key, value and output linear projection layers) of *Multi-Head Self-Attention* in the first transformer block as the key weight tensors, refer as $W_{*\,Atten(1)}^{\,V-Mod}$, $(* \in [q, k, v, o])$. Then, the weight tensors of *Multi-Head Self-Attention* in the other vision modality transformer blocks can be expanded as:

$$
\begin{aligned}
W_{q\,Atten(i)}^{\,V-Mod} &= C_{q\,Atten(i)}^{\,V-Mod} \cdot W_{q\,Atten(1)}^{\,V-Mod}, \quad (i = 2, \cdots, M) \\
W_{k\,Atten(i)}^{\,V-Mod} &= C_{k\,Atten(i)}^{\,V-Mod} \cdot W_{k\,Atten(1)}^{\,V-Mod}, \quad (i = 2, \cdots, M) \\
W_{v\,Atten(i)}^{\,V-Mod} &= C_{v\,Atten(i)}^{\,V-Mod} \cdot W_{v\,Atten(1)}^{\,V-Mod}, \quad (i = 2, \cdots, M) \\
W_{o\,Atten(i)}^{\,V-Mod} &= C_{o\,Atten(i)}^{\,V-Mod} \cdot W_{o\,Atten(1)}^{\,V-Mod}, \quad (i = 2, \cdots, M)
\end{aligned} \tag{1}
$$

where $C_{*\,Atten(i)}^{\,V-Mod}$ refers to the learnable OTMV coefficients for vision modality *Multi-Head Self-Attention* layers.

Assume there are N language modality transformer blocks in the language modality backbone. We regard the weight tensors (including query, key, value and output linear projection layers) of *Multi-Head Self-Attention* in the first transformer block as the key weight tensors, refer as $W_{*\,Atten(1)}^{\,L-Mod}$, $(* \in [q, k, v, o])$. Then, the weight tensors of *Multi-Head Self-Attention* in the other language modality transformer blocks can be expanded as:

$$
\begin{aligned}
W_{q\,Atten(i)}^{\,L-Mod} &= C_{q\,Atten(i)}^{\,L-Mod} \cdot W_{q\,Atten(1)}^{\,L-Mod}, \quad (i = 2, \cdots, N) \\
W_{k\,Atten(i)}^{\,L-Mod} &= C_{k\,Atten(i)}^{\,L-Mod} \cdot W_{k\,Atten(1)}^{\,L-Mod}, \quad (i = 2, \cdots, N) \\
W_{v\,Atten(i)}^{\,L-Mod} &= C_{v\,Atten(i)}^{\,L-Mod} \cdot W_{v\,Atten(1)}^{\,L-Mod}, \quad (i = 2, \cdots, N) \\
W_{o\,Atten(i)}^{\,L-Mod} &= C_{o\,Atten(i)}^{\,L-Mod} \cdot W_{o\,Atten(1)}^{\,L-Mod}, \quad (i = 2, \cdots, N)
\end{aligned} \tag{2}
$$

where $C_{*\,Atten(i)}^{\,L-Mod}$ refers to the learnable scalar coefficients for language modality *Multi-Head Self-Attention* layers.

In the vision modality backbone, we refer the weight tensors of *Feed Forward* layers in the first transformer block as $W_{FF(1)}^{V-Mod}$. In the language modality backbone, we regard the weight tensors of *Feed Forward* layers in the first transformer block as $W_{FF(1)}^{L-Mod}$. Then, we can merge these two key tensors into a unified cross-vision-language key tensor, refer as $W_{mg\,FF(1)}^{\;\;VL-Mod}$.

$$W_{mg\,FF(1)}^{\;\;VL-Mod} = Mean_E\left(W_{FF(1)}^{V-Mod}, W_{FF(1)}^{L-Mod}\right) \quad (3)$$

where $Mean_E$ is the operation to get the elementwise mean of the two given tensors. Then, the weight tensors of *Feed Forward* in the other vision modality and language modality transformer blocks can be expanded as:

$$\begin{aligned} W_{FF(i)}^{V-Mod} &= C_{FF(i)}^{V-Mod} \cdot W_{mg\,FF(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, M) \\ W_{FF(i)}^{L-Mod} &= C_{FF(i)}^{L-Mod} \cdot W_{mg\,FF(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, N) \end{aligned} \quad (4)$$

where $C_{FF(i)}^{V-Mod}$ and $C_{FF(i)}^{L-Mod}$ refer to the learnable scalar OTMV coefficients for vision and language modality *Feed Forward* layers, respectively.

Assume there are K vision-language transformer blocks in the vision-language cross-modality model part. We regard the weight tensors (including query, key, value and output linear projection layers) of *Multi-Head Self-Attention* in the first transformer block as the key weight tensors, refer as $W_{*\,Atten(1)}^{\;\;VL-Mod}$, $(* \in [q, k, v, o])$. Then, the weight tensors of *Multi-Head Self-Attention* in the other vision-language cross-modality transformer blocks can be expanded as:

$$\begin{aligned} W_{q\,Atten(i)}^{\;\;VL-Mod} &= C_{q\,Atten(i)}^{\;\;VL-Mod} \cdot W_{q\,Atten(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, K) \\ W_{k\,Atten(i)}^{\;\;VL-Mod} &= C_{k\,Atten(i)}^{\;\;VL-Mod} \cdot W_{k\,Atten(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, K) \\ W_{v\,Atten(i)}^{\;\;VL-Mod} &= C_{v\,Atten(i)}^{\;\;VL-Mod} \cdot W_{v\,Atten(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, K) \\ W_{o\,Atten(i)}^{\;\;VL-Mod} &= C_{o\,Atten(i)}^{\;\;VL-Mod} \cdot W_{o\,Atten(1)}^{\;\;VL-Mod}, \;\; (i = 2, \cdots, K) \end{aligned} \quad (5)$$

where $C_{*\,Atten(i)}^{\;\;VL-Mod}$ refers to the learnable scalar OTMV coefficients for vision-language cross-modality *Multi-Head Self-Attention* layers.

We refer the weight tensors of *Feed Forward* layers in the first transformer block as $W_{FF(1)}^{VL-Mod}$. Then, the weight tensors of *Feed Forward* in the other vision-language cross-modality transformer blocks can be expanded as:

$$W_{FF(i)}^{VL-Mod} = C_{FF(i)}^{VL-Mod} \cdot W_{FF(1)}^{VL-Mod}, \;\; (i = 2, \cdots, K) \quad (6)$$

where $C_{FF(i)}^{VL-Mod}$ refers to the learnable scalar coefficients for vision-language cross-modality *Feed Forward* layers.

In this manner, we ***only need to physically save the values of the key weight tensors*** (i.e., $W_{*\,Atten(1)}^{\;\;V-Mod}$, $W_{*\,Atten(1)}^{\;\;L-Mod}$, $W_{mg\,FF(1)}^{\;\;VL-Mod}$, $W_{*\,Atten(1)}^{\;\;VL-Mod}$, $W_{FF(1)}^{VL-Mod}$) ***and several OTMV coefficients***. The calculation and parameter size increase is negligible because each OTMV coefficient is just a single scalar value. The values of the other weight tensors are easily obtained with the key weight tensors and the corresponding OTMV coefficients. So, *there is no need to physically save them in the memory*, i.e., leading to ***apparent memory saving*** for the whole VLM model. The OTMV concept is shown in Figure 2.

3.2. OTMV training strategy

The major challenge with the prior tuning paradigms is the high cost of training, fine-tuning, or compressing the single VLM multiple times to obtain multiple variants.

We improve the training strategy in the OTMV tuning by introducing a ***high-order series skip mechanism***, which is controlled by the OTMV gate G. During the VLM's OTMV tuning process, each high-order series weight tensor (i.e., each weight tensor without physically stored values) is attached with an OTMV gate. These tensors connected by the OTMV gates may be randomly skipped in the calculation, both in the forward and backward paths, based on a set skip probability. The skip mechanism is implemented by adding a skip probability parameter to the OTMV gate, $G(p_{skip})$. For example, setting the skip probability to 0.15 means that the OTMV gate will be 15% randomly closed, and the connected weight tensor will also be 15% randomly skipped in the OTMV training process. A higher skip probability value can help the VLM better adapt to different expansions but may result in increased training difficulty and potentially lead to a more noticeable drop in final accuracy. Our experiments indicate that a skip probability between 0.05 and 0.1 usually provides an excellent dynamic adaptation capability for the OTMV-tuned VLM, with a good balance in accuracy degradation for all the varied VLM variants.

Through the high-order series skip mechanism, OTMV compels the VLM model to *learn with this regulation during the tuning process*, wherein ***some weight tensors are dynamically decided to expand or skip during inference***. Once the OTMV-tuned VLM reaches an acceptable accuracy level, we can dynamically determine which weight tensors are expanded or skipped in different VLM variants during the inference process. These VLM variants can maintain a good accuracy level without any additional preparation for deployment. The OTMV tuning workflow is summarized in Algorithm 1.

4. Experiments

For the experiments in this paper, we choose PyTorch [39] with version 1.13.0 as the framework to implement all methods. The results of the model training and fine-tuning experiments are obtained with V100 [35] and A100 [36] GPU clusters. All the reference models and algorithms use the default data type provided in public repositories.

Algorithm 1: Once-Tuning-Multiple-Variants (**OTMV**) tuning for the given VLM model

Input: Original target VLM model $\boldsymbol{VLM}_o$, Tuning dataset $\boldsymbol{D}_t$

Data: Skip probability of the attached OTMV gates p_{skip}, Overall tuning loss threshold δ_t

Output: **OTMV**-tuned VLM variant model $\boldsymbol{VLM}_{otmv}$. It can dynamically generate multiple VLM variants by expanding or skipping by OTMV gates during the inference process for deployment.

while *Overall OTMV tuning loss:* $\boldsymbol{L}_{otmv}$ *is larger than threshold* δ_t **do**

  // Substitute ***Multi-Head Self-Attention*** weight tensors with OTMV expansion form

  if *Weight tensors from the vision modality backbone* **then**

    $W_{*\,Atten(i)}^{\;V-Mod} = C_{*\,Atten(i)}^{\;V-Mod} \cdot W_{*\,Atten(1)}^{\;V-Mod} \cdot G\left(p_{skip}\right), \;\; (* \in [q,k,v,o]), \;\; (i = 2, \cdots, M)$

  else if *Weight tensors from the language modality backbone* **then**

    $W_{*\,Atten(i)}^{\;L-Mod} = C_{*\,Atten(i)}^{\;L-Mod} \cdot W_{*\,Atten(1)}^{\;L-Mod} \cdot G\left(p_{skip}\right), \;\; (* \in [q,k,v,o]), \;\; (i = 2, \cdots, N)$

  else if *Weight tensors from the vision-language cross-modality part* **then**

    $W_{*\,Atten(i)}^{\;VL-Mod} = C_{*\,Atten(i)}^{\;VL-Mod} \cdot W_{*\,Atten(1)}^{\;VL-Mod} \cdot G\left(p_{skip}\right), \;\; (* \in [q,k,v,o]), \;\; (i = 2, \cdots, K)$

  // Substitute ***Feed Forward*** weight tensors with OTMV expansion form

  Merge two key tensors into a unified cross-vision-language key tensor: $W_{mg\,FF(1)}^{\;\;VL-Mod} = Mean_E\left(W_{FF(1)}^{V-Mod}, W_{FF(1)}^{L-Mod}\right)$

  if *Weight tensors from the vision modality backbone* **then**

    $W_{FF(i)}^{V-Mod} = C_{FF(i)}^{V-Mod} \cdot W_{mg\,FF(1)}^{\;\;VL-Mod} \cdot G\left(p_{skip}\right), \;\; (i = 2, \cdots, M)$

  else if *Weight tensors from the language modality backbone* **then**

    $W_{FF(i)}^{L-Mod} = C_{FF(i)}^{L-Mod} \cdot W_{mg\,FF(1)}^{\;\;VL-Mod} \cdot G\left(p_{skip}\right), \;\; (i = 2, \cdots, N)$

  else if *Weight tensors from the vision-language cross-modality part* **then**

    $W_{FF(i)}^{VL-Mod} = C_{FF(i)}^{VL-Mod} \cdot W_{FF(1)}^{VL-Mod} \cdot G\left(p_{skip}\right), \;\; (i = 2, \cdots, K)$

  // OTMV training with the VLM model in OTMV expansion form and attaching OTMV gates

  Use the identical loss function from $\boldsymbol{VLM}_o$, calculate the overall OTMV tuning loss $\boldsymbol{L}_{otmv}$ on tuning dataset $\boldsymbol{D}_t$

  Minimize the overall OTMV tuning loss $\boldsymbol{L}_{otmv}$ w.r.t key weight tensors and OTMV coefficients of $\boldsymbol{VLM}_{otmv}$

end

4.1. VLM reference models

To evaluate the efficacy of OTMV tuning workflow and make the comparison with prior arts, BLIP[1] [28], VLMo[2] [3] and BEiT-3[3] [53] are chosen as the target VLM models for reference. For BLIP, we choose the model type which pre-trains with the ViT-Base backbone while using a dataset bootstrapped by captioner and filter with ViT-Large. For VLMo and BEiT-3, we both choose the base and large model types, respectively.

4.2. Efficacy for OTMV tuning

To evaluate the efficacy of the OTMV tuning workflow, we choose the typical VLM models and compare the accuracy of the baseline models with the OTMV-tuned models. To make *a fair comparison*, the accuracy reported in Table 1 comes from the OTMV-tuned VLM models with all weight tensors expanded, i.e., *the baseline and the OTMV-tuned versions have the same number of weight tensors*.

Visual Question Answering task requires the VLM to generate the language answers or choose the correct answers for the questions according to the input images. So it requires the VLM models to understand and interpret both visual content and natural language questions. All the reference VLMs and OTMV-tuned models go through the pre-training stages and fine-tune the pre-trained VLMs on the VQA v2.0 dataset [16]. All the VQA scores in Table 1 are reported on the VQA test-dev and test-standard split.

Visual Reasoning task requires the VLM reasoning about sets of objects, making comparisons, understanding spatial relations within the images, and determining whether the given sentences are true or false about the pairing images. To solve this task, VLMs must accurately interpret the visual content of the images and align it with the corresponding natural language descriptions. All the reference VLMs and OTMV-tuned models go through the pre-training stages and fine-tune the pre-trained VLMs on the NLVR2 dataset [47]. All the accuracy results in Table 1 are reported on the NLVR2 development and public test set (test-P), respectively.

Vision-Language Retrieval task involves finding relevant images based on textual descriptions (i.e., text-to-image retrieval) or retrieving textual descriptions corresponding to a given image (i.e., image-to-text retrieval). This task is a critical component of cross-modal retrieval, which aims to bridge the semantic gap between visual and textual information. All the reference VLMs and OTMV-tuned models are evaluated on two widely used benchmark datasets, COCO [31] and Flickr30K [40], and the Karpa-

[1] https://github.com/salesforce/BLIP

[2] https://github.com/microsoft/unilm/tree/master/vlmo

[3] https://github.com/microsoft/unilm/tree/master/beit3

Models	Tasks	Visual Question Answering		Visual Reasoning		Image-Text Retrieval				Image Captioning			
	Datasets	VQA v2.0		NLVR2		COCO		Flickr30k		COCO			
	Metrics	test-dev (%)	test-std (%)	dev (%)	test-P (%)	I-TR (%)	T-IR (%)	I-TR (%)	T-IR (%)	BLEU@4	METEOR	CIDEr	SPICE
VLMo (Base)		76.64	76.89	82.77	83.34	74.8	57.2	92.3	79.3	-	-	-	-
VLMo (Base)-OTMV		76.53	76.81	82.72	83.29	74.8	57.0	92.3	79.2	-	-	-	-
VLMo (Large)		79.94	79.98	85.64	86.86	78.2	60.6	95.3	84.5	-	-	-	-
VLMo (Large)-OTMV		79.92	79.99	85.66	86.83	78.3	60.6	95.4	84.6	-	-	-	-
BLIP		78.25	78.32	82.15	82.24	81.2	64.1	97.2	87.5	39.70	31.36	133.30	23.77
BLIP-OTMV		78.19	78.27	82.03	82.17	81.1	64.0	97.2	87.4	39.61	31.37	133.32	23.71
BEiT-3 (Base)		78.45	78.52	84.61	85.28	79.1	61.4	96.3	86.2	39.35	31.08	133.60	24.17
BEiT-3 (Base)-OTMV		78.32	78.41	84.53	85.22	79.0	61.2	96.2	86.1	39.31	31.01	133.49	24.10
BEiT-3 (Large)		82.53	82.58	89.22	89.95	82.1	63.4	97.2	88.1	41.46	32.02	143.29	25.23
BEiT-3 (Large)-OTMV		82.50	82.59	89.19	89.96	82.1	63.4	97.3	88.1	41.49	32.04	143.30	25.27

Table 1. Comparison between the naive VLM models with OTMV-tuned versions to verify the efficacy of OTMV tuning workflow. The models with ***-OTMV** means tuned by OTMV workflow. (All values are averaged with 5 runs. Variances: 0.04)

thy split [23] is applied for both datasets. The corresponding Top-1 image-to-text (I-TR) and text-to-image (T-IR) retrieval scores are reported in Table 1.

Image Captioning is the task of generating a textual description that accurately describes the content of an image. This task requires the VLM models to understand and interpret visual information, then produce coherent and contextually relevant captions. Following the reference VLMs to go through the pre-training stage, then the models are fine-tuned and evaluated with Karpathy test split on COCO dataset. BLEU@4, METEOR, CIDEr, and SPICE metrics are reported in Table 1.

From the results in Table 1, even the OTMV tuning workflow introduces the weight tensors sharing and OTMV gates, but there is ***negligible accuracy influence*** against the baseline VLM models.

4.3. Dynamic expansion capability for OTMV

To evaluate the efficacy of OTMV-tuned VLM's dynamic expansion capacity, we check the accuracy of OTMV-tuned VLM expanded variants and compare them to other efficient models compressed by typical VLM compression techniques, DistillVLM [14], EfficientVLM [52], and UPop [44]. Figure 3 shows the comparison for the visual question answering task on VQA v2.0, and Figure 4 shows the image caption task on COCO.

Methods	EfficientVLM	DistillVLM	UPop	OTMV
Tuning Cost	149.1 (23.3x)	131.8 (20.6x)	188.8 (29.5x)	6.4 (1x)

Table 2. Tuning cost for OTMV and other model compression methods. The unit of tuning cost is *A100 GPU Day*.

Figures 3 and 4 show shows that OTMV has obviously higher metrics than those compressed by DistillVLM, EfficientVLM, and UPop for the VLM variants with the same model parameter size. Moreover, with similar model sizes, the OTMV expanded BEiT-3 Large variant is ***even slightly better*** than the BEiT-3 Base model. Another benefit of the OTMV is that it can ***easily expand more variants*** than the other method. For example, we expand 24 variants[4] from

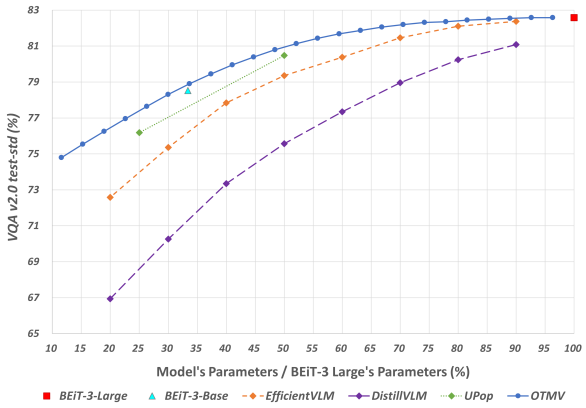

Figure 3. Comparison of OTMV tuned and expanded BEiT-3 Large variants with other efficient variants obtained with compression on visual question answering task (VQA v2.0 dataset).

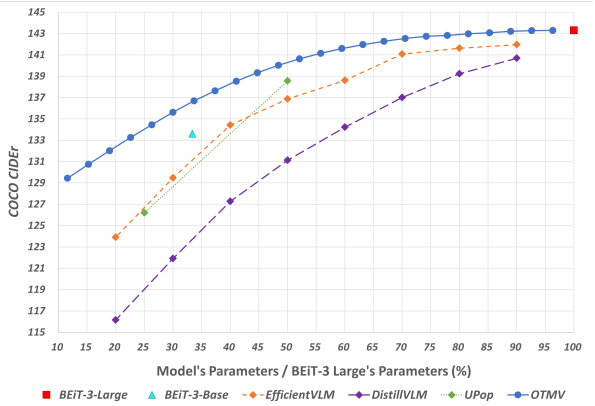

Figure 4. Comparison of OTMV tuned and expanded BEiT-3 Large variants with other efficient variants obtained with compression on image captioning task (COCO dataset).

the once-tuning OTMV model, while for the other compression methods, if they want to get the same number of compact variants, they need to compress multiple times. We compare the tuning cost[5] in Table 2. The OTMV workflow

[4]The OTMV-tuned BEiT-3 Large model can expand more than 24 variants by expanding and skipping weight tensors during inference. We just expanded 24 representative variants to compare with the prior art.

[5]For a fair comparison, we assume all the tunings happen in fine-tuning stages to align with prior methods.

Models	Multi-head Attention Sharing Strategies	Tasks	Visual Question Answering		Visual Reasoning		Image Captioning	
		Datasets	VQA v2.0		NLVR2		COCO	
		Metrics	test-dev (%)	test-std (%)	dev (%)	test-P (%)	BLEU@4	CIDEr
BEiT-3 (Base)	*Across Different (Vision)*		72.77 (-5.55)	72.58 (-5.83)	54.96 (-29.57)	55.01 (-30.21)	38.48 (-0.83)	128.77 (-4.72)
	Across Different (Language)		53.17 (-25.15)	53.14 (-25.24)	51.02 (-33.51)	51.11 (-34.11)	22.93 (-16.38)	68.20 (-65.29)
	Within Same Modalities		78.32	78.41	84.53	85.22	39.31	133.49
BEiT-3 (Large)	*Across Different (Vision)*		78.25 (-4.25)	78.38 (-4.21)	57.59 (-31.60)	57.57 (-32.39)	40.06 (-1.43)	134.39 (-8.91)
	Across Different (Language)		56.12 (-26.38)	56.21 (-26.38)	53.63 (-35.56)	53.75 (-36.21)	23.20 (-18.29)	72.21 (-71.09)
	Within Same Modalities		82.50	82.59	89.19	89.96	41.49	143.30

Table 3. Comparison among different weight tensors sharing strategies for the multi-head attention layers in VLM blocks.

has a much lower cost for computation resources. The cost is significantly higher if we want to train 24 variants for a fair comparison, roughly 73800 *A100 GPU Days* (11530× larger than OTMV).

4.4. Ablation of weight-sharing strategies

When introducing the workflow of OTMV, we mention that only the weight tensors *within the same modality transformer blocks* for the ***multi-head attention*** layers are shared. To prove the rationality of this weight-sharing strategy, we make an ablation study to compare with other choices for the multi-head attention layers:

- ***Within Same Modalities***, i.e., vision-modality transformer blocks share the weight tensors within vision-modality transformer blocks, language-modality transformer blocks share the weight tensors within language-modality transformer blocks.
- ***Across Different Modalities (Vision)*** means both vision-modality and language-modality transformer blocks share the weight tensors from the vision-modality blocks.
- ***Across Different Modalities (Language)*** means both vision-modality and language-modality blocks share the weight tensors from the language-modality blocks.

We choose the BEiT-3 (Base and Large) as the target models and evaluate the accuracy of OTMV-tuned versions with different weight-sharing strategies. The ablation comparison is shown in Table 3.

From the ablation experiment results, the default weight-sharing strategy (i.e., *Within Same Modalities*) is ***apparently better*** than the other weight-sharing strategies across different modalities. It proves that weight-sharing with mixed vision modality and language modality will ***damage the modality alignment*** in the original VLM model. A more interesting finding is that the *Across Different Modalities (Vision)* strategy brings much less accuracy degradation than the *Across Different Modalities (Language)* strategy. That means the shared weight tensors from language modality can hardly interpret the visual structures and features in vision modality. However, using the shared weight tensors from vision modality can *partially* interpret some features in language modality, though not perfectly.

In OTMV workflow, we share the weight tensors across the language-modality and vision-modality transformer blocks by default for the ***feed-forward*** layers. We also make a quick ablation to compare with the *"Within Same Modalities"* weight-sharing strategy. For the feed-forward layers, weight tensors shared across different modalities or within the same modalities ***make marginal differences*** in the final accuracy metrics on various benchmarks (less than ***0.4%***). That means the ***multi-head attention*** layers are responsible for extracting the essential features from the vision-modality and language-modality, which ***share nearly no common patterns***. While the ***feed-forward*** layers are responsible for the nonlinear transformations of the attention outputs, and are ***not directly related*** to the alignment of the different modalities. So they will not cause apparent accuracy degradation when sharing across modalities.

4.5. Deployment of OTMV-tuned VLM variants

Because the OTMV-tuned VLM variants use the OTMV gates to skip the weight layers in the inference process, there is ***no reliance on the specific software stack support for actual deployment***. With the real saving of the VLM models' memory cost on hardware, it can bring a considerable performance boost. For example, the OTMV-tuned BEiT-3 Large variant with 60% parameters saving can have a 63% performance improvement on the A100 GPU.

5. Conclusions

OTMV tuning workflow offers a game-changing solution for adapting VLMs to different scenarios with minimal training and deployment costs while maintaining high accuracy. With the dynamic weight expansion capacity injection, many weight tensors are substituted with the key weight tensors multiplied by learnable OTMV coefficients. Such an approximation of the original VLM *reduces memory requirements for industrial hardware*. Experiments prove that the OTMV significantly enhances the efficiency and flexibility of VLMs, making it a valuable contribution to various vision-language tasks. Moreover, OTMV workflow can generate multiple efficient variants in a single tuning effort. By evaluating these variants' performance on industrial toolkits [26, 37, 38], we can find potential performance optimization opportunities for the kernels and software stacks in these toolkits. With the help of OTMV, such performance optimization efficiency is greatly improved.

Acknowledgements This work is supported by Shanghai Science and Technology Commission Explorer Program Project (24TS1401300), Shanghai Natural Science Foundation (No. 23ZR1402900), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). The computations in this research were performed using the CFFF platform of Fudan University.

References

[1] 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*, pages 2425–2433, 2015. 1

[2] Yang Bai, Yang Zhou, Jun Zhou, Rick Siow Mong Goh, Daniel Shu Wei Ting, and Yong Liu. From generalist to specialist: Adapting vision language models via task-specific visual instruction tuning. *arXiv preprint arXiv:2410.06456*, 2024. 2

[3] Hangbo Bao, Wenhui Wang, Li Dong, Qiang Liu, Owais Khan Mohammed, Kriti Aggarwal, Subhojit Som, Songhao Piao, and Furu Wei. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. *Advances in Neural Information Processing Systems*, 35:32897–32912, 2022. 1, 3, 6

[4] Hangbo Bao, Wenhui Wang, Li Dong, and Furu Wei. Vl-beit: Generative vision-language pretraining. *arXiv preprint arXiv:2206.01127*, 2022. 1

[5] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020. 3

[6] Ruisi Cai, Saurav Muralidharan, Greg Heinrich, Hongxu Yin, Zhangyang Wang, Jan Kautz, and Pavlo Molchanov. Flextron: Many-in-one flexible large language model. *arXiv preprint arXiv:2406.10260*, 2024. 3

[7] Qingqing Cao, Bhargavi Paranjape, and Hannaneh Hajishirzi. Pumer: Pruning and merging tokens for efficient vision language models. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023. 2

[8] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024. 2

[9] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[10] Fnu Devvrit, Sneha Kudugunta, Aditya Kusupati, Tim Dettmers, Kaifeng Chen, Inderjit S Dhillon, Yulia Tsvetkov, Hannaneh Hajishirzi, Sham M Kakade, Ali Farhadi, et al. Matformer: Nested transformer for elastic inference. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ NeurIPS 2023)*, 2023. 3

[11] 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. In *International Conference on Learning Representations*, 2020. 3

[12] Sivan Doveh, Assaf Arbelle, Sivan Harary, Eli Schwartz, Roei Herzig, Raja Giryes, Rogerio Feris, Rameswar Panda, Shimon Ullman, and Leonid Karlinsky. Teaching structured vision & language concepts to vision & language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2657–2668, 2023. 2

[13] Yifan Du, Zikang Liu, Junyi Li, and Wayne Xin Zhao. A survey of vision-language pre-trained models. *arXiv preprint arXiv:2202.10936*, 2022. 1

[14] 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*, pages 1428–1438, 2021. 3, 7

[15] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *International Journal of Computer Vision*, 132(2): 581–595, 2024. 2

[16] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017. 6

[17] Quentin Grail, Julien Perez, and Eric Gaussier. Globalizing bert-based transformer architectures for long document summarization. In *Proceedings of the 16th conference of the European chapter of the Association for Computational Linguistics: Main volume*, pages 1792–1810, 2021. 1

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[19] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 2, 3

[20] Yu Huang, Chenzhuang Du, Zihui Xue, Xuanyao Chen, Hang Zhao, and Longbo Huang. What makes multi-modal learning better than single (provably). *Advances in Neural Information Processing Systems*, 34:10944–10956, 2021. 1

[21] 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*, pages 4904–4916. PMLR, 2021. 2

[22] Jingjing Jiang and Nanning Zheng. Mixphm: redundancy-aware parameter-efficient tuning for low-resource visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24203–24213, 2023. 2

[23] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015. 7

[24] Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. Non-autoregressive machine translation with disentangled context transformer. In *International conference on machine learning*, pages 5144–5155. PMLR, 2020. 1

[25] Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022. 2

[26] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023. 8

[27] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3041–3050, 2023. 1

[28] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022. 2, 6

[29] Haokun Lin, Haoli Bai, Zhili Liu, Lu Hou, Muyi Sun, Linqi Song, Ying Wei, and Zhenan Sun. Mope-clip: Structured pruning for efficient vision-language models with module-wise pruning error metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27370–27380, 2024. 2

[30] Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26689–26699, 2024. 1, 2

[31] 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*, pages 740–755. Springer, 2014. 6

[32] Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. A joint neural model for information extraction with global features. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 7999–8009, 2020. 1

[33] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022. 1

[34] Tahira Naseem, Srinivas Ravishankar, Nandana Mihindukulasooriya, Ibrahim Abdelaziz, Young-Suk Lee, Pavan Kapanipathi, Salim Roukos, Alfio Gliozzo, and Alexander Gray. A semantics-aware transformer model of relation linking for knowledge base question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 256–262, 2021. 1

[35] NVIDIA. NVIDIA Tesla V100 GPU Architecture, 2017. 5

[36] NVIDIA. NVIDIA A100 Tensor Core GPU Architecture, 2020. 5

[37] NVIDIA. NVIDIA TensorRT, 2022. 8

[38] NVIDIA. NVIDIA TensorRT-LLM, 2023. 8

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems-Autodiff Workshop*, 2017. 5

[40] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015. 6

[41] 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*, pages 8748–8763. PMLR, 2021. 2

[42] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[43] Oindrila Saha, Grant Van Horn, and Subhransu Maji. Improved zero-shot classification by adapting vlms with text descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17542–17552, 2024. 2

[44] Dachuan Shi, Chaofan Tao, Ying Jin, Zhendong Yang, Chun Yuan, and Jiaqi Wang. Upop: Unified and progressive pruning for compressing vision-language transformers. In *International Conference on Machine Learning*, pages 31292–31311. PMLR, 2023. 3, 7

[45] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7262–7272, 2021. 1

[46] Dirk Jan Struik. *A source book in mathematics, 1200-1800*. Princeton University Press, 2014. 2, 3

[47] 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. 6

[48] Shenghuan Sun, Alexander Schubert, Gregory M Goldgof, Zhiqing Sun, Thomas Hartvigsen, Atul J Butte, and Ahmed Alaa. Dr-llava: Visual instruction tuning with symbolic clinical grounding. *arXiv preprint arXiv:2405.19567*, 2024. 2

[49] Moming Tang, Chengyu Wang, Jianing Wang, Chuanqi Tan, Songfang Huang, Cen Chen, and Weining Qian. Xtreme-clip: Extremely parameter-efficient tuning for low-resource vision language understanding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6368–6376, 2023. 2

[50] Brook Taylor. *Methodus incrementorum directa & inversa*. Inny, 1717. 2

[51] Mojtaba Valipour, Mehdi Rezagholizadeh, Hossein Rajabzadeh, Marzieh Tahaei, Boxing Chen, and Ali Ghodsi. Sortednet, a place for every network and every network in its place: Towards a generalized solution for training many-in-one neural networks. *arXiv preprint arXiv:2309.00255*, 2023. 2

[52] Tiannan Wang, Wangchunshu Zhou, Yan Zeng, and Xinsong Zhang. Efficientvlm: Fast and accurate vision-language models via knowledge distillation and modal-adaptive pruning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13899–13913, 2023. 2, 3, 7

[53] 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 vision and vision-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19175–19186, 2023. 1, 2, 3, 6

[54] Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. Filip: Fine-grained interactive language-image pre-training. *arXiv preprint arXiv:2111.07783*, 2021. 2

[55] Da Yin, Feng Gao, Govind Thattai, Michael Johnston, and Kai-Wei Chang. Givl: Improving geographical inclusivity of vision-language models with pre-training methods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10951–10961, 2023. 2

[56] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019. 3

[57] Maxime Zanella and Ismail Ben Ayed. Low-rank few-shot adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1593–1603, 2024. 2

[58] Yan Zeng, Xinsong Zhang, Hang Li, Jiawei Wang, Jipeng Zhang, and Wangchunshu Zhou. X 2-vlm: All-in-one pre-trained model for vision-language tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2

[59] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 2

[60] Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6748–6758, 2023. 1