ViM: Vision Middleware for Unified Downstream Transferring

Yutong Feng[1], Biao Gong[1], Jianwen Jiang[1], Yiliang Lv[1], Yujun Shen[2], Deli Zhao[1], Jingren Zhou[1]
[1]Alibaba Group [2]Ant Group
{fengyutong.fyt, a.biao.gong, jianwen.alan, shenyujun0302, zhaodeli}@gmail.com
{yiliang.lyl, jingren.zhou}@alibaba-inc.com

Abstract

Foundation models are pre-trained on massive data and transferred to downstream tasks via fine-tuning. This work presents ***Vision Middleware (ViM)****, a new learning paradigm that targets unified transferring from a single foundation model to a variety of downstream tasks. ViM consists of a zoo of lightweight plug-in modules, each of which is independently learned on a midstream dataset with a shared frozen backbone. Downstream tasks can then benefit from an adequate aggregation of the module zoo thanks to the rich knowledge inherited from midstream tasks. There are three major advantages of such a design. From the efficiency aspect, the upstream backbone can be trained only once and reused for all downstream tasks without tuning. From the scalability aspect, we can easily append additional modules to ViM with no influence on existing modules. From the performance aspect, ViM can include as many midstream tasks as possible, narrowing the task gap between upstream and downstream. Considering these benefits, we believe that ViM, which the community could maintain and develop together, would serve as a powerful tool to assist foundation models.*

1. Introduction

The *pretrain-finetune* paradigm has served as a general framework across various vision tasks, where models are pre-trained on large-scale datasets and fine-tuned on downstream tasks [64]. Recent efforts have been attracted to build up a *foundation model* [54, 73, 80] with large architecture and massive pre-trained data (*e.g.*, in the scale of billions). Considering the trend of scaling up foundation models, it is costly to fine-tune model for different tasks separately, and worthy of solving tasks with single model. When directly transferred to downstream tasks, foundation models still suffer from the *task-gap* problem. Downstream tasks may require different targets with the upstream, thus could not fully leverage the pre-learned knowledge. As shown in Figure 1, models are observed with *preference* to those tasks similar to their encountered ones. Task preference restricts unified transferring of single foundation model to multiple tasks with varying targets.

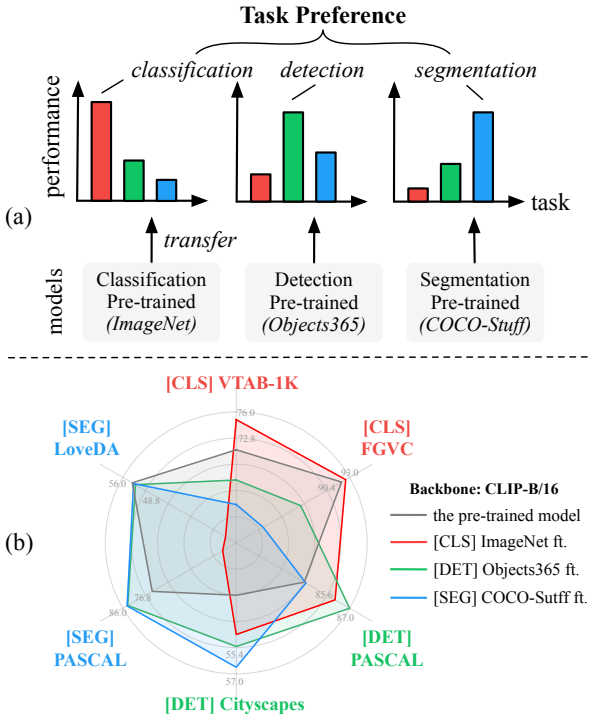

Figure 1. **Task preference problem**: (a) *Single-task pre-trained* models tend to perform better on tasks similar to their upstream. (b) *Intermediate task fine-tuning* further tunes the pre-trained model on specific task, showing similar imbalanced performance.

To bridge the task-gap, existing works address the problem on either upstream or downstream. Upstream works propose multi-task pre-training to simultaneously learn various types of tasks [18, 23, 28, 49, 72]. However, when extending to a new pre-training task, these works require to re-formulate the task I/O and re-train the model together with existing tasks, which is complicated and time-consuming. Downstream works introduce prompt [34, 46,

87] or adapter-based tuning [5, 8, 15, 31, 35] to adapt to downstream tasks with additional parameters. Since the downstream datasets for transferring are generally small (*e.g.*, in the scale of thousands), the appended parameters might be insufficiently trained to master newly-encountered tasks, usually resulting in unsatisfying performance [21,74].

In this paper, we present a unified framework for supporting multiple downstream tasks with single foundation model. After the upstream pre-training, we introduce a collection of midstream tasks based on middle-scale datasets (*e.g.*, in the scale of millions). For each midstream task, a lightweight plug-in module is inserted into the pre-trained backbone, and individually optimized to learn the current task. The foundation model is frozen without any parameter tuning in the above process. Throughout this way, we collect the **Vision Middleware (ViM)**, a module zoo containing knowledge from diverse midstream tasks based on the single foundation model. To fully leverage the knowledge of ViM for downstream transferring, we develop practical strategies to adaptively aggregate ViM modules. Modules correlated with the downstream task would be emphasized for better transferring. ViM is expected to cope with the task-gap problem with sufficient supervision in midstream, and achieve balanced performance on multiple downstream tasks without preference. During the experiments, we build up a ViM consisting of modules trained from 47 midstream tasks in 12 types, which can be grouped into global recognition, local recognition, vision and language understanding and self-supervised learning. We then evaluate with the averaged transferring performance on 30 downstream tasks in 4 types. The experimental results show satisfying performance of ViM with balanced improvements on multiple tasks.

ViM introduces a new paradigm of applying foundation models for unified transferring, which shows the following advantages: (i) For the efficiency, the foundation model is maintained frozen after upstream pre-training, thus without any cost of storing different model parameters for various downstream tasks. (ii) For the scalability, new ViM module can be individually trained and freely appended into the current ViM. The diverse middle-scale datasets from the community also enable us to easily expand ViM into great scale. (iii) For the performance, ViM addresses the task-gap issue via including ViM modules of various midstream tasks, thus can achieve balanced performance for unified downstream transferring. With the above advantages and verified performance, we would like to encourage the community to maintain a public ViM to which all researchers can contribute with their own well-trained ViM modules.

2. Related Work

We review related works from perspectives of different stages for solving the task-gap problem.

Upstream: multi-task pre-training. Multi-task pre-training [56] aims to train single model for supporting multiple tasks. *Methods using individual heads:* Pioneer works study multi-task from single data source [57, 66, 77, 82, 83, 88], where each sample contains multiple labels from different tasks, *e.g.*, boundary, depth and segmentation maps. However, the single source scenario is restricted with expensive annotation and limited scenes. Multi-source multi-task pre-training further combines different tasks from multiple datasets [18,28,39], *e.g.*, ImageNet [10] and COCO [44]. The multi-source scenario is much more free to benefit from diverse supervisions from varying tasks. *Methods using shared head:* More recently, studies on foundation models strive to unify the multi-task, multi-source and multi-modality pre-training [23, 49, 54, 72, 73, 80]. By converting different tasks into the same format of input and output (I/O), *e.g.*, sequence to sequence [49, 72], models can be pre-trained on them with both shared backbone and head. Such a unification removes all the task-specific architectures, thus learns universal representation to master various tasks simultaneously. *Discussion:* Multi-task pre-training involves various tasks to expand the task-coverage of the foundation model. But it would be hard to continuously append new pre-training tasks, since all the tasks should be pre-trained together once again.

Midstream: intermediate task fine-tuning. In the literature of natural language processing (NLP), it has been demonstrated that tuning pre-trained models on additional intermediate tasks can further boost downstream performance [4, 51–53]. Experimental results indicate that not all intermediate tasks can benefit downstream tasks [52], considering the task-similarity, level of semantics, domain distribution, *etc.* Recent application of vision foundation models also introduce intermediate fine-tuning before evaluating their transferrability, *e.g.*, fine-tuning BEiT-3 [73] on Objects365 [59] before transferring to COCO [44]. The intermediate tasks are generally based on larger datasets, which enables fully supervision for understanding tasks. Though with improved performance, since all the parameters of model have been fully adjusted for supporting single type of task, intermediate task fine-tuning can not be unified solution for multi-task transferring.

Downstream: parameter efficient tuning. Downstream solutions aim to adapt model with tuning a few parameters. *Prompt-based tuning:* The prompt tuning is firstly introduced in NLP area to bridge the gap between upstream and downstream tasks [45]. Pioneer works define hand-crafted text templates as *hard prompts* to convert downstream tasks into the format of pre-trained task [2, 16], which might be sub-optimal due to the man-made templates. Continuous prompts are further proposed to assign additional trainable parameters as prompt for downstream tasks [42, 46, 47]. VPT [34] firstly adapts prompt for vision models, where

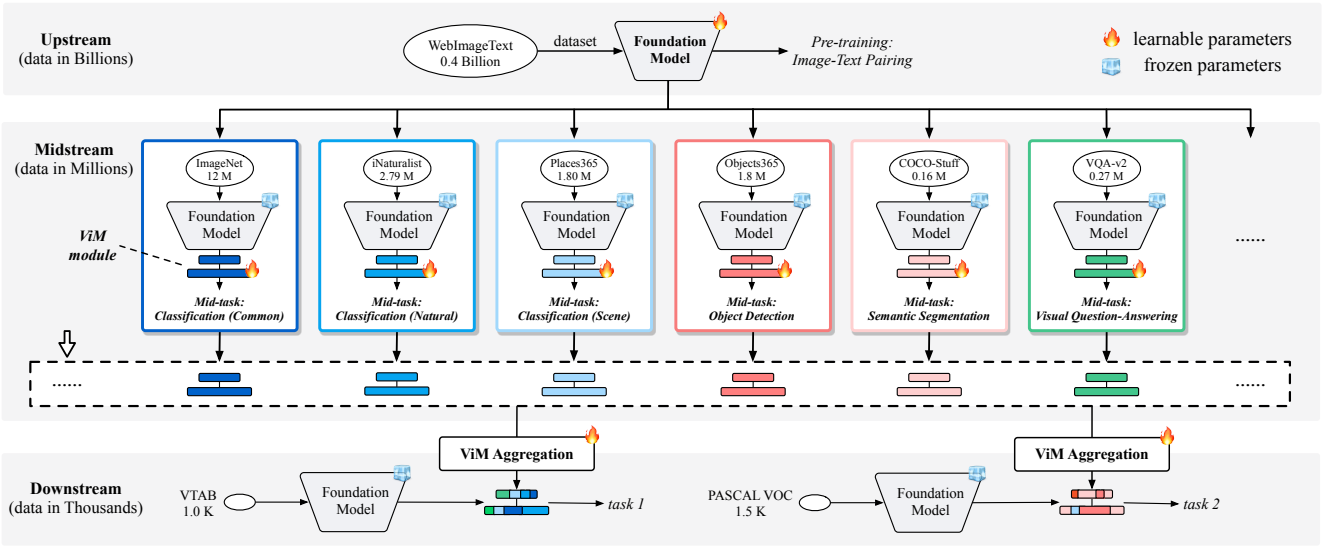

Figure 2. **Vision Middleware** for *unified* downstream transferring. The framework is in three stages based on varying sizes of datasets: (i) In the upstream, a foundation model is pre-trained leveraging a massive dataset (*e.g.*, in billions). (ii) In the midstream, a sequence of different midstream tasks are introduced with middle-scale datasets (*e.g.*, in millions). ViM is collected as a zoo of modules individually trained for each task, and the foundation model is frozen as the same backbone for all mid-tasks. (iii) In the downstream, ViM aggregation adaptively gathers beneficial ViM modules to work for each downstream dataset (*e.g.*, in thousands).

trainable prompts are appended as tokens in each layer. *Adapter-based tuning:* Adapters are lightweight modules inserted into the model for adapting tasks in NLP [31, 32]. Compared with prompt tuning, adapters can directly influence the model architecture for more generalized adaptation. For vision models, adapters have been widely applied on multiple tasks [5, 8, 15, 35, 86] with specific design for vision modalities. *Discussion:* As a downstream solution, parameter efficient tuning is only supervised by the limited scale of downstream samples, which restricts the upper bound of its performance.

3. Vision Middleware

We introduce Vision Middleware (ViM) to formulate a unified framework for supporting various vision tasks with single foundation model, as illustrated in Figure 2. Following the framework, we present an implementation of building up and applying ViM, including the module architecture, midstream training and downstream aggregation.

3.1. Towards Unified Downstream Transferring

We start by reviewing the pretrain-finetune framework, where a foundation model is pre-trained on large-scale upstream dataset and fine-tuned on downstream tasks. Under such a framework, the downstream performance is correlated with the similarity between upstream and downstream tasks. Disparate formats of tasks could lead to unsatisfying or even degraded downstream performance. Such a task-gap restricts a pre-trained model to simultaneously support diverse downstream tasks.

To alleviate the task-gap problem for unified transferring with single foundation model, we introduce Vision Middleware (ViM) between the upstream and downstream stages, which factorizes varying abilities of model in advance to downstream transferring. ViM enables a new paradigm for assisting foundation model in the following three stages:

Upstream: A large-scale dataset with massive samples is leveraged to construct an upstream task $\mathcal{T}_{\text{up}}$, *e.g.*, classification [26], contrastive learning [6, 25, 54] and masked image modeling [1, 24, 76]. Then a foundation model is supervised by task $\mathcal{T}_{\text{up}}$ with the optimized parameters Θ_{up}. Recent efforts concentrating on pre-training have been striving to expand the scale of both model architecture and pre-trained dataset, generating outstanding foundation models with fruitful knowledge. It is noteworthy that the following processes can be used with arbitrary foundation model.

Midstream: In this stage, we introduce a set of midstream tasks, *i.e.*, $\{\mathcal{T}^1_{\text{mid}}, \mathcal{T}^2_{\text{mid}}, ..., \mathcal{T}^M_{\text{mid}}\}$, based on different middle-scale datasets. The midstream tasks are expected to show diverse task formats, and allowed to be freely appended without limitation. For each mid-task $\mathcal{T}^i_{\text{mid}}$, we assign a lightweight ViM module with parameters φ^i_{mid}, and combine it with the frozen foundation model for learning, which can be denoted as $f([\Theta_{\text{up}}, \varphi^i_{\text{mid}}]) \rightarrow \mathcal{T}^i_{\text{mid}}$. In the optimization process, only the parameters of the new ViM module are trainable for adapting to each mid-task. The original parameters of the foundation model are frozen to

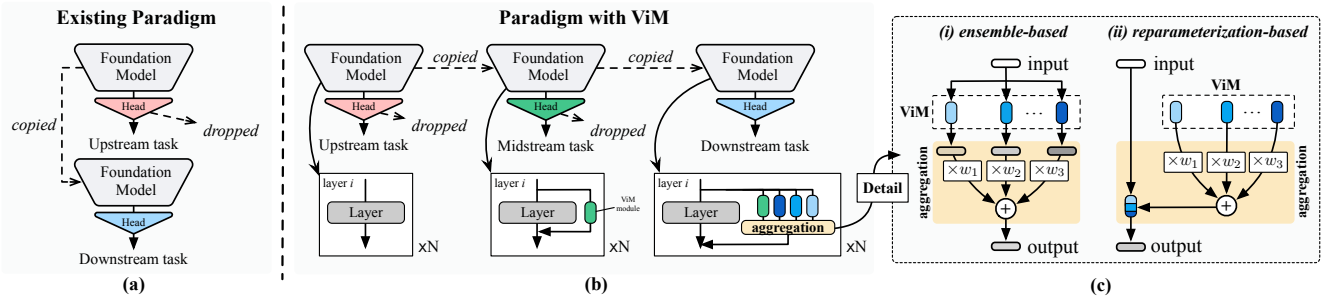

Figure 3. (a) Existing transferring paradigm *v.s.* (b) The detailed implementation of the **ViM transferring paradigm**. ViM modules are implemented as per-layer sub-modules and are aggregated in each layer. (c) Details of two **ViM aggregation** strategies.

avoid damaging the pre-learned representation with task-specific shifting. The ViM module should be in plug-in style towards the foundation model, but is not restricted for its design of architecture. Besides, the module is expected to be relatively lightweight, *i.e.*, $\Sigma_{i=1}^{M}\text{sizeof}(\varphi_{\text{mid}}^{i}) \ll \text{sizeof}(\Theta_{\text{up}})$. Such a lightweight characteristic enables us to collect ViM, *i.e.*, a module zoo $\{\varphi_{\text{mid}}^{i}\}_{i=1}^{M}$, without large memory cost, while at the same time maintaining various abilities of the foundation model.

Downstream: Based on ViM inherited from the midstream, we aim to boost the performance on a wide range of downstream tasks. For each downstream task, ViM modules relevant to current task are expected to be picked up and aggregated together for transferring. Taking one task $\mathcal{T}_{\text{down}}$ as example, the process can be denoted as

$$f([\Theta_{\text{up}},\ \text{agg}(\varphi_{\text{mid}}^{1}, \varphi_{\text{mid}}^{2}, ..., \varphi_{\text{mid}}^{M})]) \rightarrow \mathcal{T}_{\text{down}}, \qquad (1)$$

where $\text{agg}(\cdot)$ is an aggregation function that adaptively incorporates different ViM modules for benefiting $\mathcal{T}_{\text{down}}$. It is noted that there may exist irrelevant modules in the zoo to current downstream task. Since such cases are unknown for the midstream stage, the aggregation process is responsible to filter out these irrelevant modules, and maintain the performance on varying types of downstream tasks.

Scalability: Thanks to the high extensibility, ViM could be maintained online by the community to continuously support different tasks. Here we present a brief conceptual framework: *(i) Constructing ViM Module Zoo.* The zoo stores both the initial and developer uploaded ViM modules. Each module is specified with its training task, dataset and performance. The task-specific head can be optionally stored together for direct usage. *(ii) Benchmarking and Grouping.* A benchmark with diverse tasks is set to evaluate ViM modules. Then modules can be divided into groups based on their strengths, and serve as more lightweight ViMs for specific requirements. *(iii) Uploading Control.* Any uploaded ViM module is firstly tested before appended into the zoo. We would check the parameter sizes, avoid repeating with existing ones, and evaluate its transferability to filter out meaningless modules. *(iv) Usage.* Users can download pre-set ViM groups, or customize their own ViM zoo for specific target. Corresponding algorithm of ViM aggregation is also provided for transferring.

3.2. Constructing and Applying ViM

Following the framework, we present an implementation of how to build up, train and aggregate ViM. It should be noted that the following practice is not the only way of implementing the framework. As an exploration, we demonstrate the potential of ViM for unified downstream transferring, and lay foundations for future works.

In this paper, we take the vision transformer (ViT) [13] as the architecture of foundation model for presentation, which is commonly used in recent studies of pre-training [54, 73, 80]. ViT converts an image into a sequence of patch tokens, and encodes them using stacking of transformer layers with multi-head self-attention (MHSA) [67] and MLP modules.

Architecture of ViM module. The ViM module can be implemented by any lightweight plug-in module to adapt the foundation model to specific tasks. We adopt a recent adapter-based module, ConvPass [35], for its simple architecture and stable performance on vision tasks. As shown in Figure 3 (b), for each layer, an additional sequence of convolution layers are appended as a bypass. It contains an 1×1 convolution squeezing the feature channel, a 3×3 convolution with the same input and output channels, and an 1×1 convolution recovering the channel. The convolutions are concatenated with non-linear activation function, *i.e.*, GELU [29]. We append such a convolution sequence along with both the MHSA and MLP modules of all transformer layers. With extremely small value of the squeezed channel (*e.g.*, $1/96$ of the original dimension), the newly introduced parameters are only 0.4% of the whole model. Though with tiny size of learnable parameters, such a module has been proved with satisfying performance on task adaption.

ViM Training. Based on the robust design of framework, each ViM module could be individually trained via different developing tools. Additional task-specific heads would be

Table 1. **Configurations** of the tasks and datasets for training ViM modules in the midstream. The numbers in brackets after each task type indicate the number of midstream tasks. We only list the example datasets for brevity. Additional task-specific architectures are assigned for adapting to each task, which are dropped after midstream training. Totally, we train on 12 types of tasks containing 47 mid-tasks, generating a zoo of diverse ViM modules with representation for various tasks.

Task Type (#mid-tasks)	Example Datasets	Size	Task-specific Architecture
Image Classification (21)	ImageNet-21K [10, 55], Places365 [48], *etc.*	26M	Linear Classifier
Object Detection (7)	Objects365 [59], LogoDet-3K [69], *etc.*	2.3M	ViTDet [43] *w/* FastRCNN [19]
Instance Segmentation (2)	LVIS [22], COCO [44]	223K	ViTDet [43] *w/* MaskRCNN [27]
Semantic Segmentation (4)	COCO-Stuff164K [3], ADE20K [89], *etc.*	240K	UperNet [75]
Keypoints Estimation (1)	COCO-Keypoint [44]	57K	ViTDet [43] *w/* KeypointRCNN [27]
Depth Estimation (2)	NYU Depth V2 [62], KITTI (Eigen) [14, 17]	48K	Deconvolutions
Visual Question Answering (2)	VQA-v2 [20], GQA [33]	265K	Concat. MLP with BERT [11]
Referring Expression Compre. (3)	RefCOCO, RefCOCOg, RefCOCO+ [81]	60K	MLM with PEVL text encoder [78]
Phrase Grounding (1)	Flickr30K [79]	32K	MLM with PEVL text encoder [78]
Visual Relationship Detection (1)	Visual Genome [41]	101K	MLM with PEVL text encoder [78]
Visual Commonsense Reasoning (1)	VCR [84]	100K	MLM with PEVL text encoder [78]
Self-Supervised Learning (2)	ImageNet-1K [10]	1.3M	MoCo-v2 [7], MAE [24]
TOTAL (47)		**32M**	

introduced depending on the current mid-task, *e.g.*, linear classifier or RoI-Align [27]. These task-specific heads are possibly heavy, but would all be dropped after the midstream stage. Only the trained ViM module is kept for memorizing the representation required by the mid-task. Besides the mid-task supervised modules, we also append a *zero* ViM module with parameters φ^0_{mid}, which is initialized to output all-zero values. The zero ViM module serves as a buffer to leave more space of downstream optimization.

ViM Aggregation. Given the zoo of ViM with varying abilities of tasks, we aim to adaptively select and aggregate them for specific downstream task. Different strategies of aggregation are explored based ensemble and reparameterization, as illustrated in Figure 3 (b). The ***ensemble*** is a common strategy to aggregate models from multiple pre-training sources. Taking the input, we firstly forward it with different ViM modules, then aggregate their outputs. To better activate relevant modules, we adopt the Mixture-of-Experts (MoE) [60] that uses a routing module to pick up top-k beneficial modules. Different from mapping the input to generate aggregation weights as in MoE, we directly optimize a vector of aggregation weights $\mathbf{w} \in \mathbb{R}^M$ for each task, and find it effective in experiments. We activate the ViM modules with top-k weight values and aggregate their outputs with weights after softmax. By default, we activate all the modules into aggregation. The ***reparameterization***-based strategy firstly aggregates the parameters of ViM modules into a single one, then only forward it once with better efficiency. We follow the same implementation of aggregation weights with the ensemble strategy. Since the aggregation is fixed after training, ViM could be employed as a single module during the inference.

4. Experiments

4.1. Experimental Setup

Upstream Setup. We utilize the CLIP [54] pre-trained image encoder as the foundation model, due to its powerful generalization performance. CLIP is pre-trained with image-text contrastive learning on a web-scale dataset with 0.4 billion image-text pairs. For the main experiments, we construct ViM based on the official ViT-B/16 model.

Midstream Setup. As shown in Table 1, we incorporate 12 types of midstream tasks with totally 47 sub-tasks for training ViM. The tasks can grouped into global recognition (*e.g.*, image classification), local recognition (*e.g.*, detection and segmentation), vision-language (*e.g.*, VQA and visual grounding) and self-supervised learning (*e.g.* masked image modeling). Additional task-specific architectures are assigned for the varying targets on each task. Taking the zero ViM module into account, we collect a ViM with size of 48. More details of the midstream configurations and training results are in the *Supplementary Material*.

Downstream Setup. We evaluate the transferrability of the proposed framework on the following downstream tasks: ***(i) Classification:*** We leverage two common benchmarks for fine-grained classification, *i.e.*, the VTAB-1k [85] and FGVC benchmark. VTAB-1k contains 19 datasets divided into Natural, Specialized and Structured groups with different contents. FGVC consists of 5 datasets including the CUB-200-2011 [68], NABirds [65], Oxford Flowers [50], Stanford Dogs [37] and Stanford Cars [40]. Linear classifier is appended as the task head for classification learning. ***(ii) Object Detection:*** We evaluate with the PASCAL [12] dataset (07+12) for common scene, and the Cityscapes [9] dataset for traffic scene. For the task head of detection, ViTDet [43] is used with FastRCNN [19] for PASCAL

Table 2. The **downstream transferring results** on 30 tasks in 4 groups of task types, *i.e.*, image classification (**CLS**), object detection (**DET**), semantic segmentation (**SEG**) and depth estimation (**DEP**). Evaluation metrics for different tasks are shown in the brackets. The last column is highlighted with an averaged metric across all tasks. The best results among methods with frozen CLIP-B/16 backbone are **bold** for each task. **FB** indicates whether the method is based on frozen backbone, which is more flexible for multi-task transferring. For the single-task pre-training, † indicates the results are reported by INTERN [58], while ‡ indicates the results are reported by MuST [18].

Method				CLS (acc.↑)		DET (AP_{50} ↑)		SEG (mIoU↑)		DEP (RMSE↓ / δ_1 ↑)		Avg.
Upstream Pre-training	Midstream	Downstream Tuning	(FB)	VTAB (19)	FGVC (5)	PASCAL	Cityscapes	PASCAL	LoveDA	NYUv2	KITTI	
single-task pre-training												
ImageNet [10] Cls. †	-	Linear	✓	-	-	82.7	-	67.8	-	0.43 / -	3.06 / -	-
CLIP-R50x16 [54] †	-	Linear	✓	-	-	83.6	-	68.7	-	0.39 / -	2.83 / -	-
MoCo-v2 [7] †	-	Linear	✓	-	-	79.1	-	66.9	-	0.43 / -	3.09 / -	-
JFT [63] Cls. ‡	-	Fully	✗	-	-	85.2	-	80.4	-	- / 86.5	-	-
SimCLR (w. JFT) [6] ‡	-	Fully	✗	-	-	84.1	-	74.9	-	- / 84.8	-	-
multi-task pre-training												
MuST (w. JFT) [18]	-	Fully	✗	-	-	87.9	-	82.9	-	- / 89.5	-	-
X-Learner$_{R152}$ [28]	-	Fully	✗	-	-	88.6	-	82.6	-	- / 91.3	-	-
Unified-IO$_{LARGE}$ [49]	-	Fully	✗	-	-	-	-	-	-	0.40 / -	-	-
INTERN [58] (1B param.)	-	Linear	✓	-	-	90.7	-	78.7	-	0.32 / -	2.55 / -	-
intermediate fine-tuning / parameter efficient tuning												
CLIP-B/16 [54]	-	Fully	✗	71.4	92.1	84.2	49.8	74.1	53.0	0.37 / 90.0	2.42 / 96.0	76.3
	ImageNet21K ft.	ConvPass [35]	✓	75.1	**92.6**	86.1	53.4	45.5	23.4	0.44 / 84.8	2.75 / 94.5	69.4
	Objects365 ft.	ConvPass [35]	✓	67.7	87.4	87.0	54.5	84.0	52.0	0.42 / 86.3	2.64 / 94.4	76.7
	COCO-Stuff ft.	ConvPass [35]	✓	64.7	83.1	84.3	56.4	84.3	52.6	0.44 / 85.2	2.82 / 93.9	75.8
	-	Linear	✓	61.7	81.7	48.5	37.6	78.6	49.0	0.61 / 70.1	3.49 / 88.4	64.4
	-	VPT [34]	✓	71.3	85.3	-	-	82.8	49.6	0.50 / 80.0	3.05 / 92.1	-
	-	Adapter [31]	✓	73.1	90.1	84.7	54.4	84.5	51.8	0.43 / 85.4	2.70 / 94.3	77.3
	-	ConvPass [35]	✓	73.1	90.2	84.9	55.1	84.4	53.0	0.41 / 86.9	2.52 / 95.1	77.8
Ours: Vision Middleware (ViM)												
CLIP-B/16 [54]	+ViM	ViM-agg (rep.)	✓	74.8	90.2	85.9	55.6	85.0	52.6	0.39 / 88.7	2.46 / 95.6	78.6
	+ViM	ViM-agg (ens.)	✓	**75.3**	91.7	**87.2**	**56.9**	**86.0**	**54.1**	**0.38 / 89.0**	**2.38 / 96.3**	**79.6**

and MaskRCNN [27] for Cityscapes. ***(iii) Semantic Segmentation:*** We evaluate with the PASCAL [12] (2012 set) dataset for common scene, and the LoveDA [71] dataset for remote sensing. We reshape the feature maps from layer $4, 6, 8, 12$ into varying resolutions with up/down sampling as in [1], then feed them into a semantic FPN module [38] for segmentation map prediction. ***(iv) Depth Estimation:*** We evaluate with the NYU Depth V2 [62] and KITTI [17] (with [14] split) datasets for indoor and outdoor scenes, respectively. We append deconvolution and up-sampling layers to predict the depth map. ***Summary:*** We evaluate with 30 tasks in 4 types. Details of datasets and training configurations are in *Supplementary Material*.

Compared Methods. (i) We firstly compare the results of single or multiple task *pre-training* as reported. Since they are based on various backbones and tuning methods, we list the results for reference. (ii) For the *intermediate task fine-tuning*, we respectively fine-tune the model with the largest datasets of 3 tasks in the midstream, *i.e.*, classification on ImageNet-21K [10, 55], detection on Objects365 [59] and segmentation COCO-Stuff164K [3]. The fine-tuned models are transferred to downstream with SoTA tuning method. (iii) For the *parameter efficient tuning*, we compare with recent SoTA tuning methods for vision [31, 34, 35]. (iv) The two ViM aggregation methods are denoted as ViM-agg (rep. or ens.). We optimize the aggregation weights, activated ViM modules and task head for downstream task learning.

4.2. Downstream Transferring Results

Analysis on Main Results. Table 2 shows the transferring results on all downstream tasks. To investigate the unified transferability, we compute an averaged metric across different tasks (taking the δ_1 for depth estimation). Compared with single-task pre-training, multi-task pre-training models generally show balanced performance on different tasks. As we have discussed, they require to include all the task types in pre-training stage, and lack extensibility to more tasks. The intermediate fine-tuned models achieve great improve-

Table 3. Results of changing the **backbone** to CLIP-L/14.

Backbone	Method	VTAB	-Natural	-Special.	-Struct.
CLIP-L/14	ConvPass	76.81	84.41	87.78	58.25
	ViM	**78.07**	**85.02**	**88.13**	**61.06**

Table 4. Results of **further transferring to midstream tasks**. "Mid." indicates training single ViM module in the midstream.

Method	**IN1K** [10]	**iNat18** [30]	**THDogs** [90]	**Logo** [70]	**FoodX** [36]
Fully	87.89	81.93	92.05	92.39	84.90
Mid.	86.60	78.40	89.94	91.37	83.37
ViM	87.74	81.18	91.37	92.31	84.80

ment on similar tasks, but also decrease dramatically on the remaining tasks. For instance, the ImageNet-21K fine-tuned model performs well on VTAB classification (+3.7% accuracy than fully tuning), but shows −28.6% mIoU on PASCAL segmentation. Such an imbalanced performance is also illustrated in Figure 1 (b). For the parameter efficient tuning, *i.e.*, VPT [34], adapter [31] and ConvPass [35], they show general improvement to linear probing on all tasks. However, such improvements are restricted by the limited downstream data for mastering new tasks.

When introducing ViM in the midstream and conducting ViM aggregation for downstream, we achieve the highest performance on the averaged metric and almost all tasks. It is noteworthy that we conduct fair comparison on all methods with the same CLIP-B/16 backbone. The ensemble-based aggregation shows better improvements with +3.3% than fully tuning without touching the backbone. The reparameterization-based aggregation, though performs slightly worse than ensemble, is also more effective than compared methods, and shows better efficiency on the computation cost during the inference. With a frozen backbone, ViM is also competitive with multi-task pre-training models using fully tuning or larger backbone. On some dense prediction tasks, *e.g.*, PASCAL detection and NYUv2, MuST [18] and X-Learner [28] achieve better results, which is attributed to the natural gap between their convolutional backbone and our plain ViT backbone.

ViM with different backbone. The design of our framework is decoupled with backbone. To further demonstrate the effectiveness of ViM, we also construct a ViM based on the CLIP ViT-L/14 backbone, with the 21 classification mid-tasks. We further transfer it to downstream VTAB classification. Table 3 shows that ViM maintains improvement with larger backbone and higher baseline results.

Transferring ViM to midstream tasks. We also apply the same transferring process back to the midstream tasks for training ViM. Table 4 shows the results on some midstream datasets based on the above ViM with CLIP-L/14. Since the backbone is frozen, the performance of single ViM module trained in midstream is worse than fully tuning.

Table 5. Results of further transferring to **more types of tasks**.

Method	**Results** (%)
Keypoints Estimation on COCO	
ConvPass	AP_{kpt}: 54.37, $AP_{kpt}50$: 80.29, $AP_{kpt}75$: 58.35
ViM	AP_{kpt}: **56.39**, $AP_{kpt}50$: **83.00**, $AP_{kpt}75$: **59.16**
Visual Question-Answering on GQA	
ConvPass	val acc: 66.14
ViM	val acc: **67.35**
Referring Expression Comprehension on RefCOCO	
ConvPass	val acc: 79.92, testA: 84.30, testB: 71.62
ViM	val acc: **81.45**, testA: **84.73** testB: **72.38**

Table 6. Results of transferring to **unseen tasks** in the midstream.

ViM (size)	**CLS** VTAB	**DET** PASCAL	**SEG** PASCAL	**DEP** KITTI
ConvPass (ref.)	73.1	84.9	84.4	2.52 / 95.1
w/o CLS (37)	75.0	-	-	-
w/o DET (39)	-	86.6	-	-
w/o SEG (42)	-	-	85.9	-
w/o DEP (46)	-	-	-	2.51 / 95.5
whole set (48)	75.3	87.2	86.0	2.38 / 96.3

Table 7. Directly transferring **single ViM module**, which is named as [Task]-[Dataset]. The best value of single module is underlined.

Method	**AVG**	**CLS** (VTAB)			**SEG**	
		-Natural	-Special.	-Struct.	PASCAL	LoveDA
CLS-IN1K	71.8	78.9	86.1	57.1	84.1	53.0
CLS-Places	71.5	77.2	85.9	58.7	83.3	52.3
DET-LVIS	70.8	74.4	84.3	56.7	83.9	53.4
SEG-ADE20K	70.4	74.1	84.2	56.2	85.2	52.2
ViM (zoo)	**73.2**	**79.9**	**87.2**	**58.9**	**86.0**	**54.1**

However, with further transferring ViM to midstream tasks, we successfully compensate the performance degradation without touching the backbone, even comparable to fully tuning, demonstrating the effectiveness of our paradigm.

Transferring ViM to unseen tasks. For specific downstream task, the ViM modules trained on similar mid-tasks play important roles. To further investigate their importance, we construct experiments to transfer ViM to unseen tasks, *e.g.*, removing the classification ViM modules and transferring to classification. As shown in Table 6, removing similar midstream modules indeed decreases the corresponding results, but still achieves better performance than compared methods. The results suggest that ViM modules on similar mid-tasks are *important but not the only beneficial modules* for downstream transferring. The detailed study on contribution of modules is in Section 4.4.

Transferring ViM to more types of tasks. To fully investigate the generalization of ViM, we present the downstream performance on more types of tasks. Results in Tab. 5 show

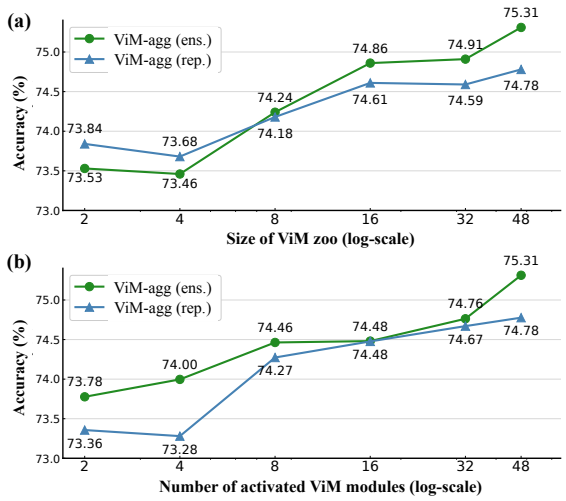

Figure 4. Ablation study on (a) the **ViM zoo size** and (b) the **number of activated ViM modules**, conducted on VTAB classification with two aggregation strategies.

that ViM could consistently improve the transferring results on different tasks.

Transferring with single ViM module. The transferability of single ViM module is studied in Table 7. We observe similar unbalanced performance as the intermediate task fine-tuning, where modules generally perform better only on tasks correlated with their encountered ones.

4.3. Ablation Studies

We conduct ablation studies within the process of ViM aggregation, the results of which are expected to be helpful for further improved implementation of our framework.

On the size of ViM. To verify the significance of expanding ViM with more modules, we construct ViM in varying sizes via randomly adding modules into the zoo. Figure 4 (a) shows the results. The downstream performance is observed to be positively related to the zoo size of ViM, which encourages us to continuously append new ViM modules from diverse mid-tasks for better transferring.

On the number of activated ViM modules. By default, all the ViM modules are activated. We study the influence of activating different number of modules in Figure 4 (b). Both showing increasing performance with more activate modules, the ensemble-based aggregation consistently outperforms reparameterization-based strategy. Therefore, when training with more available memory, it is suggested to increase the number of activated modules and firstly consider the ensemble-based strategy.

On the modeling of aggregation weights. The aggregation weights are set as trainable vectors by default. We also explore to generate weights using a projection of the input embeddings, as in [60, 61]. Table 8 shows the results of

Table 8. Ablation study on **modeling of aggregation weights** with ViM-agg (rep.). "#params" indicates the modeling parameters.

Agg. weights	VTAB	-Natural	-Special.	-Struct.	#params
proj. (linear)	18.49	3.99	37.51	13.97	0.1758 M
proj. (MLP)	73.04	78.95	85.99	54.19	0.4281 M
vector	74.78	79.14	86.21	58.98	0.0002 M

Table 9. Ablation study on **empty ViM modules**.

Method	VTAB	-Natural	-Special.	-Struct.	#params
empty ($\times$48)	71.77	78.55	84.40	52.36	14.98 M
ViM ($\times$48)	75.31	80.18	86.80	58.95	14.98 M

Table 10. Ablation study on **building ViM with Adaptformer**.

Method	VTAB-1K	-Natural	-Special.	-Struct.
Adaptformer-8	72.40	77.83	82.21	57.17
ViM (Adaptformer-8, $\times$6)	74.60	79.44	86.70	57.68

projection using linear layer or MLP. Though with more trainable parameters, it is hard to learn to project the input into proper aggregation weights, especially with a linear projection. Therefore, the trainable vector is a simple yet effective way of modeling aggregation weights.

On the parameters of ViM modules. To demonstrate the improvement of ViM is not introduced by more parameters, we conduct experiment with *empty modules*, which are set as the initial state of ViM modules without midstream training. As shown in Tab. 9, directly introducing more parameters without our midstream training would not result in similar improvement as ViM, and may cause even worse performance for transferring.

On the architecture of ViM modules. The ViM module can be implemented as any plug-able lightweight module. We also construct a ViM zoo based on another choice, Adaptformer [5] with 6 midstream classification tasks, and evaluate on VTAB-1K for transferring. The results in Tab. 10 demonstrate the effectiveness of ViM as a general framework for applying foundation models.

4.4. Visualization Analysis

For further understanding the influence of different ViM modules in the aggregation process, we visualize the aggregation weights of all layers in Figure 5. We label the ViM modules trained with different groups of mid-tasks. It is observed that ViM modules with similar mid-tasks to the current task are emphasized in general, *e.g.*, modules with local recognition abilities show higher weights on downstream detection task. Nevertheless, modules with different mid-tasks could also contribute from their own perspectives and knowledge to further improve the performance. We also observe that ViM modules are aggregated with different

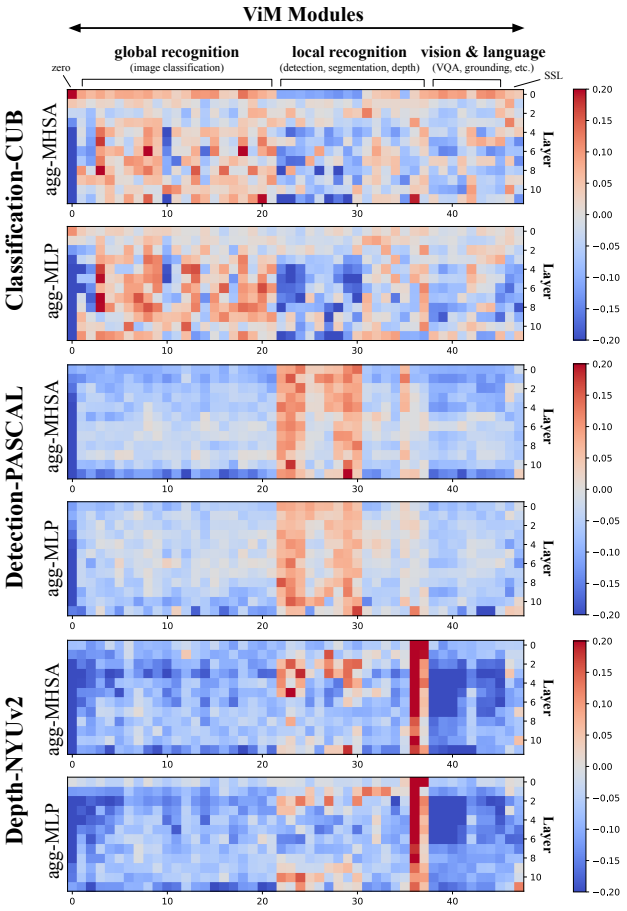

Figure 5. **Visualization of the aggregation weights** on different downstream tasks. The activation of ViM modules along with the MHSA and MLP inside each layer are visualized, respectively.

weights among layers, suggesting that different modules are found beneficial in varying depth of the network, and work in a per-layer cooperation manner.

5. Conclusion

In this paper, we present the Vision Middleware (ViM) in a unified framework to support multiple vision tasks with single foundation model. ViM contains a zoo of lightweight modules, each of which is trained individually on a midstream task for factorizing task-specific knowledge with shared backbone. For downstream transferring, the ViM modules are adaptively aggregated to boost the performance on various downstream tasks. Experimental results indicate that ViM achieves balanced and improved performance. The efficiency, extensibility and effectiveness of ViM encourage the community to maintain a public ViM, where modules are contributed from different researchers to facilitate the application of foundation models.

References

[1] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *Int. Conf. Learn. Represent.*, 2021. 3, 6

[2] 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. *Adv. Neural Inform. Process. Syst.*, pages 1877–1901, 2020. 2

[3] Holger Caesar, Jasper R. R. Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1209–1218, 2018. 5, 6

[4] Ting-Yun Chang and Chi-Jen Lu. Rethinking why intermediate-task fine-tuning works. *arXiv preprint arXiv:2108.11696*, 2021. 2

[5] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022. 2, 3, 8

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *Int. Conf. Mach. Learn.*, pages 1597–1607, 2020. 3, 6

[7] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 5, 6

[8] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022. 2, 3

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3213–3223, 2016. 5

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 248–255, 2009. 2, 5, 6, 7

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics, 2019. 5

[12] Zheng Dong, Ke Xu, Yin Yang, Hujun Bao, Weiwei Xu, and Rynson W. H. Lau. Location-aware single image reflection removal. In *Int. Conf. Comput. Vis.*, pages 4997–5006, 2021. 5, 6

[13] 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 *Int. Conf. Learn. Represent.*, 2020. 4

[14] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Adv. Neural Inform. Process. Syst.*, 2014. 5, 6

[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. *arXiv preprint arXiv:2110.04544*, 2021. 2, 3

[16] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020. 2

[17] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, pages 1231–1237, 2013. 5, 6

[18] Golnaz Ghiasi, Barret Zoph, Ekin D Cubuk, Quoc V Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In *Int. Conf. Comput. Vis.*, pages 8856–8865, 2021. 1, 2, 6, 7

[19] Ross Girshick. Fast r-cnn. In *Int. Conf. Comput. Vis.*, pages 1440–1448, 2015. 5

[20] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6325–6334, 2017. 5

[21] Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*, 2021. 2

[22] Agrim Gupta, Piotr Dollár, and Ross B. Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5356–5364, 2019. 5

[23] Tanmay Gupta, Amita Kamath, Aniruddha Kembhavi, and Derek Hoiem. Towards general purpose vision systems: An end-to-end task-agnostic vision-language architecture. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16399–16409, 2022. 1, 2

[24] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16000–16009, 2022. 3, 5

[25] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9726–9735, 2020. 3

[26] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Int. Conf. Comput. Vis.*, pages 4918–4927, 2019. 3

[27] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, pages 2961–2969, 2017. 5, 6

[28] Yinan He, Gengshi Huang, Siyu Chen, Jianing Teng, Wang Kun, Zhenfei Yin, Lu Sheng, Ziwei Liu, Yu Qiao, and Jing Shao. X-learner: Learning cross sources and tasks for universal visual representation. In *Eur. Conf. Comput. Vis.*, 2022. 1, 2, 6, 7

[29] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 4

[30] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8769–8778, 2018. 7

[31] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *Int. Conf. Mach. Learn.*, pages 2790–2799, 2019. 2, 3, 6, 7

[32] 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 *Int. Conf. Learn. Represent.*, 2021. 3

[33] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6700–6709, 2019. 5

[34] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *Eur. Conf. Comput. Vis.*, 2022. 1, 2, 6, 7

[35] Shibo Jie and Zhi-Hong Deng. Convolutional bypasses are better vision transformer adapters. *arXiv preprint arXiv:2207.07039*, 2022. 2, 3, 4, 6, 7

[36] Parneet Kaur, Karan Sikka, Weijun Wang, Serge Belongie, and Ajay Divakaran. Foodx-251: a dataset for fine-grained food classification. *arXiv preprint arXiv:1907.06167*, 2019. 7

[37] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, 2011. 5

[38] Alexander Kirillov, Ross B. Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6399–6408, 2019. 6

[39] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6129–6138, 2017. 2

[40] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 554–561, 2013. 5

[41] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, pages 32–73, 2017. 5

[42] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 2

[43] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022. 5

[44] 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 *Eur. Conf. Comput. Vis.*, pages 740–755, 2014. 2, 5

[45] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. 2

[46] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021. 1, 2

[47] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021. 2

[48] Alejandro López-Cifuentes, Marcos Escudero-Viñolo, Jesús Bescós, and Álvaro García-Martín. Semantic-aware scene recognition. *Pattern Recognit.*, page 107256, 2020. 5

[49] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*, 2022. 1, 2, 6

[50] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008. 5

[51] Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018. 2

[52] Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. What to pre-train on? efficient intermediate task selection. In *Annual Conference on Empirical Methods in Natural Language Processing*, pages 10585–10605, 2021. 2

[53] Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R Bowman. Intermediate-task transfer learning with pretrained models for natural language understanding: When and why does it work? *arXiv preprint arXiv:2005.00628*, 2020. 2

[54] 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 *Int. Conf. Mach. Learn.*, pages 8748–8763, 2021. 1, 2, 3, 4, 5, 6

[55] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021. 5, 6

[56] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 2

[57] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Int. Conf. Learn. Represent.*, 2014. 2

[58] 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. 6

[59] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Int. Conf. Comput. Vis.*, pages 8430–8439, 2019. 2, 5, 6

[60] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017. 5, 8

[61] Yang Shu, Zhi Kou, Zhangjie Cao, Jianmin Wang, and Mingsheng Long. Zoo-tuning: Adaptive transfer from a zoo of models. In *Int. Conf. Mach. Learn.*, pages 9626–9637, 2021. 8

[62] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. *Eur. Conf. Comput. Vis.*, pages 746–760, 2012. 5, 6

[63] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Int. Conf. Comput. Vis.*, pages 843–852, 2017. 6

[64] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pages 270–279, 2018. 1

[65] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 595–604, 2015. 5

[66] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. In *Eur. Conf. Comput. Vis.*, pages 527–543, 2020. 2

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Adv. Neural Inform. Process. Syst.*, 2017. 4

[68] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[69] Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, and Shuqiang Jiang. Logodet-3k: A large-scale image dataset for logo detection. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, pages 1–19, 2022. 5

[70] Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, Haishuai Wang, and Shuqiang Jiang. Logo-2k+: A large-scale logo dataset for scalable logo classification. In *Assoc. Adv. Artif. Intell.*, pages 6194–6201, 2020. 7

[71] Junjue Wang, Zhuo Zheng, Ailong Ma, Xiaoyan Lu, and Yanfei Zhong. Loveda: A remote sensing land-cover dataset for domain adaptive semantic segmentation. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021*, 2021. 6

[72] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *Int. Conf. Mach. Learn.*, pages 23318–23340, 2022. 1, 2

[73] 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*, 2022. 1, 2, 4

[74] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020. 2

[75] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Eur. Conf. Comput. Vis.*, pages 432–448, 2018. 5

[76] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9653–9663, 2022. 3

[77] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 675–684, 2018. 2

[78] Yuan Yao, Qianyu Chen, Ao Zhang, Wei Ji, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Pevl: Position-enhanced pre-training and prompt tuning for vision-language models. In *Annual Conference on Empirical Methods in Natural Language Processing*, 2022. 5

[79] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, pages 67–78, 2014. 5

[80] 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. 1, 2, 4

[81] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *Eur. Conf. Comput. Vis.*, pages 69–85, 2016. 5

[82] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Adv. Neural Inform. Process. Syst.*, pages 5824–5836, 2020. 2

[83] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3712–3722, 2018. 2

[84] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6720–6731, 2019. 5

[85] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 5

[86] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021. 3

[87] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *arXiv preprint arXiv:2206.04673*, 2022. 1

[88] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang. Joint task-recursive learning for semantic segmentation and depth estimation. In *Eur. Conf. Comput. Vis.*, pages 235–251, 2018. 2

[89] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5122–5130, 2017. 5

[90] Ding-Nan Zou, Song-Hai Zhang, Tai-Jiang Mu, and Min Zhang. A new dataset of dog breed images and a benchmark for finegrained classification. *Computational Visual Media*, pages 477–487, 2020. 7