GAIA: A Transfer Learning System of Object Detection that Fits Your Needs

Xingyuan Bu [2,4*] Junran Peng [1,3*] Junjie Yan [4] Tieniu Tan [1,3] Zhaoxiang Zhang [1,3†]
[1]University of Chinese Academy of Sciences, [2]Beijing Institute of Technology
[3]Center for Research on Intelligent Perception and Computing, CASIA, [4]SenseTime Group Limited

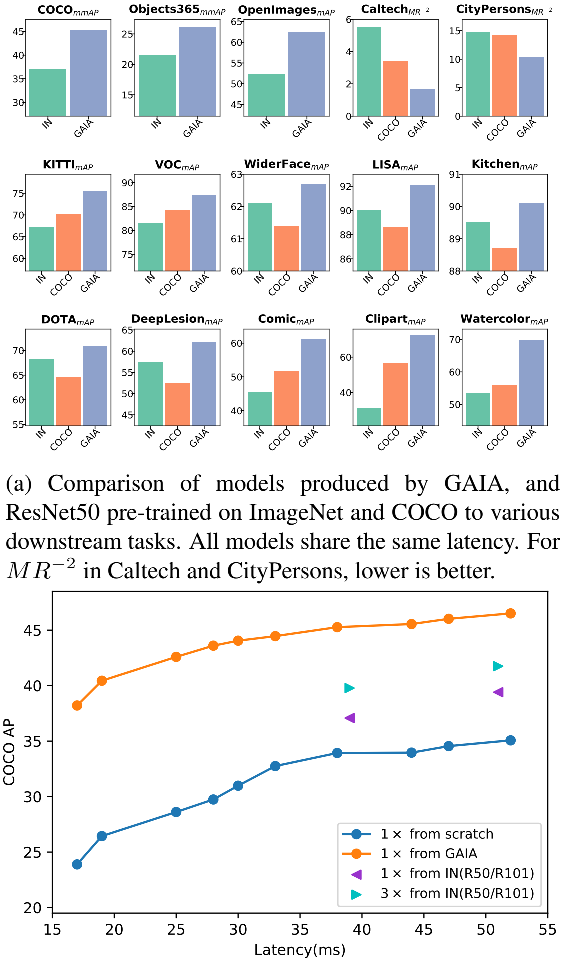
xingyuanbu@gmail.com, jrpeng4ever@126.com
yanjunjie@sensetime.com, zhaoxiang.zhang@ia.ac.cn, tnt@nlpr.ia.ac.cn

Abstract

Transfer learning with pre-training on large-scale datasets has played an increasingly significant role in computer vision and natural language processing recently. However, as there exist numerous application scenarios that have distinctive demands such as certain latency constraints and specialized data distributions, it is prohibitively expensive to take advantage of large-scale pre-training for per-task requirements. In this paper, we focus on the area of object detection and present a transfer learning system named GAIA, which could automatically and efficiently give birth to customized solutions according to heterogeneous downstream needs. GAIA is capable of providing powerful pre-trained weights, selecting models that conform to downstream demands such as latency constraints and specified data domains, and collecting relevant data for practitioners who have very few datapoints for their tasks. With GAIA, we achieve promising results on COCO, Objects365, Open Images, Caltech, CityPersons, and UODB which is a collection of datasets including KITTI, VOC, WiderFace, DOTA, Clipart, Comic, and more. Taking COCO as an example, GAIA is able to efficiently produce models covering a wide range of latency from 16ms to 53ms, and yields AP from 38.2 to 46.5 without whistles and bells. To benefit every practitioner in the community of object detection, GAIA is released at https://github.com/GAIA-vision.

1. Introduction

Transfer learning has been one of the most powerful techniques throughout the history of deep learning. The paradigm of pre-training on large-scale source datasets and finetuning on target datasets or tasks is applied over a wide range of tasks in both computer vision (CV) and natural language processing (NLP). The motivation behind is to endow models the general-purpose abilities and transfer the knowledge to downstream tasks, in order to acquire higher performance or faster convergence. Until recently, the influence



(a) Comparison of models produced by GAIA, and ResNet50 pre-trained on ImageNet and COCO to various downstream tasks. All models share the same latency. For MR^{-2} in Caltech and CityPersons, lower is better.

(b) Comparison of models with different latency on COCO.

Figure 1: Transfer performance of GAIA models that adapt to various downstream needs including specified domains and latency constraints. No whistles and bells are used.

[*]Equal contributions.
[†]Corresponding author.

of transfer learning has moved onto the next level as the scales of data are growing exponentially. In BiT [33] and WSL [43], pre-training on enormous data like JFT-300M and Instagram (3.5B images) has been proved to yield extraordinary improvements over the conventional ImageNet pre-training on downstream tasks. Similar trends are going on in NLP, as shown in BERT [17], T5 [49] and GPT-3 [6].

Although transfer learning with huge scale of pre-training data has shown the great success, it is severely afflicted by its inflexibility on model architecture. As stated in the "no free lunch" theorem [57], no single algorithm is best suited for all possible scenarios and datasets. Different datasets may request for different model architectures, and different application scenarios may request for model of different scales. To take advantage of the transfer learning, these customized models are obliged to be trained from scratch on the whole upstream datasets, which is prohibitively expensive.

This issue is even more serious in object detection, as it is one of the most significant computer vision tasks and covers an wide range of deployment scenarios. In practice, detectors are always supposed to work across various devices with distinctive resource constraints, which requires detectors to have corresponding input sizes and architectures. In addition, the correlation between distributions of object scales and the adapted network architectures are very close in object detection. Therefore, the demand for task-specific architecture adaptation is much stronger in object detection than other tasks such as image classification or semantic segmentation.

In this paper, we introduce a transfer learning system in object detection that aims to harmonize the gap between large-scale upstream training and downstream customization. We name this system "GAIA" as it could swiftly give birth to a variety of specialized solutions to heterogeneous demands, including task-specific architectures and well-prepared pre-trained weights. For users who have very few datapoints for their tasks, GAIA is able to collect relevant data to fuel their needs. Our objective is not to propose a specific method, but more to present an integrated system that brings convenience to practitioners in the community of object detection. There are two components in GAIA: *task-agnostic unification* and *task-specific adaptation*.

To begin with, we conduct the task-agnostic unification on data and architectures respectively. In order to unleash the potential of transfer learning, we collect data from multiple sources and build a huge data pool with a unified label space. The unified label space is formulated based on the *word2vec* similarity, which prevents knowledge conflicts among duplicated categories from distinctive sources and enables data of relevant categories to jointly boost the performance of detector. Besides, covering a wide range of categories provides indicators for task-specific adaptation. To realize the purpose of training plenty of models efficiently on huge upstream data, we adopt the weight sharing scheme which has been widely used in [39, 5, 48, 30, 64, 8, 65], which enables models of different widths and depths to be optimized together. As models may interfere with each other during collective training, we propose an "anchor-based progressive shrinking" to alleviate the problem.

In the task-specific adaptation procedure, GAIA needs to find adapted model architectures according to the given tasks. We quantitatively assess the ranking ability of different search methods based on the Kendall Tau measure [32] and propose an efficient and reliable method of selecting models that surprisingly fit the downstream task, regardless of data domains or latency constraints (Figure 1). To extend the utility of GAIA on the ubiquitous data-scarce scenarios, we develop GAIA an ability of collecting relevant data to downstream tasks from data pool, which yields further improvements.

The contributions of our paper are as follows:

- We demonstrate how transfer learning and weight sharing learning could be well combined, to produce powerful pre-trained models over a variety of architectures simultaneously.
- We propose an efficient and reliable approach of finding the adapted network architectures to specified downstream tasks. Powered by pre-training and task-specific architecture selection, GAIA achieves surprisingly good results over 10 downstream tasks without exclusive tuning on hyper-parameters.
- GAIA has the capability of finding relevant data based on 2 images per category in the downstream tasks to support finetuning. This further extends the utility of GAIA in data-scarce settings.

2. Related Work

2.1. Object Detection

Beginning with R-CNN [24] and its predecessors like Fast-RCNN [23] and Faster-RCNN [52], deep learning grows prosperously in the area of object detection. A great amount of methods are proposed to advance the area, including Mask RCNN [28], FPN [36], DCN [15] and Cascade RCNN [10]. Since it has been years that the most researches are conducted on Pascal VOC or MS-COCO datasets, which are viewed as "small data" by modern standards, there are researchers begin to explore on more challenging issues such as cross-domains object detection or large-scale object detection in the wild. Wang *et al.* [3] develop an universal object detection system that works across a wide range of domains including traffic signs to medical CT images. With the advent of larger datasets of object detection such as Objects365 [53], Open Images [34], and Robust Vision [1], there are works [46] focusing on solving issues in this scenarios such as long-tailed data distribution and multi-labels problems.

2.2. Neural Architecture Search

Neural architecture search(NAS) aims at automating the architecture of network design process under certain constraints. Earlier methods [68, 69, 4, 7] train thousands of candidate networks with distinctive architectures and rely on reinforcement learning or evolution algorithm to discover the optimal architectures. These methods mostly require unimaginably huge computation resources and seem forbidding for most research institutions. Gradient-based methods like DARTS [39] and Proxyless [9] come out that greatly alleviate the problem through training and searching candidate architectures inside a single super-net. However, in real world that heterogeneous tasks exist, these methods requires repetitive searching and training process for each task according to the specified hard-ware platforms and latency constraints. To address this issue, researchers further propose methods [64, 8, 65] that are to produce models across different inference latency all at once. As in area of object detection, NAS methods also spring up such as [13, 47, 22, 55].

2.3. Transfer Learning

Transfer learning has been playing a non-negligible role throughout the history of deep learning. The paradigm of pre-training on ImageNet [16] dataset has immensely pushed forward the development of computer vision, covering a wide range of tasks such as object detection [24, 23, 52, 41, 37, 50],semantic segmentation [42, 11], pose estimation [56] and *etc*. Recently, there are studies [54, 43, 60, 33] focusing on transfer learning on larger scale of data such as JFT-300M and Instagram (3.5B images). With the help of weakly supervised learning [43], knowledge distillation [60] and supervised learning [33], surprisingly good results are achieved. As in object detection, [53, 35] also prove the effectiveness of large-scale data. In addition to transfer pre-trained models, there are methods transferring data directly. In [25, 2], the authors show that more in-domain data could benefit diverse NLP tasks. In [14, 62, 26], extra similar data is selected to provide better pre-trained models. Domain transfer by adversarial training [31] also mitigates the lack of target data.

3. GAIA

We introduce GAIA, a transfer learning framework, and its detailed implementation in this section. GAIA consists of two major components: *task-agnostic unification* and *task-specific adaptation*. In the task-agnostic unification part, we collect data from multiple sources and build a large data pool with a unified label space. Then we utilize the technique of weight sharing learning for training a supernet, which enables models of various architectures to be optimized collectively. In the task-specific adaptation part, GAIA search a most adapted architecture for the given downstream task, initialize the network with weights extracted from the pre-trained supernet, and finetune it on the downstream data. We call this process "task-specific architecture selection" (TSAS). In addition, to help with users who have very few datapoints for their tasks, GAIA is able to collect the most correlated data to the given tasks from the huge data pool as relevant data. We call this ability of GAIA as "task-specific data selection" (TSDS).

3.1. Task-agnostic Unification

3.1.1 Unified Data and Label Space

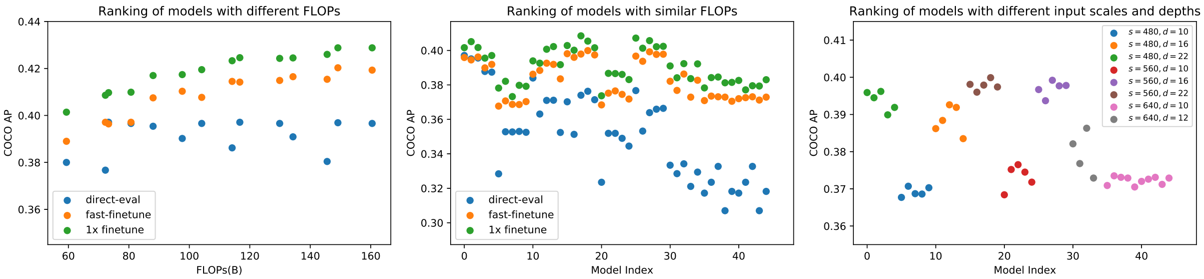
Although steady progresses have been made in distinctive datasets, they are independent of each other and often restricted to specific domains [1]. Therefore, to push forward real-world usability of an object detection system and reduce dataset bias, we merge multiple datasets into a huge data pool with a unified label space $\cup L$.We start with N datasets $D = \{d_1, d_2, \cdots, d_N\}$ and their label spaces $L = \{l_1, l_2, \cdots, l_N\}$. Each label space l_i consists of the categories $\{c_{i1}, c_{i2}, \cdots, c_{i|l_i|}\}$ corresponding to dataset d_i. To obtain the unified label space, we first choose the largest label space among L as the initial $\cup L = \{c_{\cup 1}, c_{\cup 2}, \cdots\}$. Then we map other label spaces into $\cup L$. We mark the p-th category c_{ip} from dataset d_i as an identical category to $c_{\cup q}$ if their *word2vec* [51] similarity is higher than a threshold of 0.8. If there is no similarity greater than 0.8 for c_{ip}, we mark it as a novel category and append c_{ip} into $\cup L$. Finally, all candidate mappings would be verified for reliability.

In the follow-up development, the unified label space is unique but not static as GAIA has the will to cover as many datasets as possible. When an unseen dataset needs to be merged, we repeat the label space mapping process as mentioned above. To fit the pre-trained network to the updated label space, the last *fc* layer extends corresponding new ways while the others remain unchanged.

Using this unified label space rather than training with multiple heads or separate label definitions, reduces the cost of dataset combination and mitigates the potential conflicts between duplicate categories. More importantly, it enables GAIA to provide the indicators for diverse downstream tasks; otherwise, it is impractical to obtain a reliable measure for novel datasets. The unified label space may introduce long-tail and partial annotation problems as claimed in [67]. Nevertheless, we note that these problems barely influence the downstream finetuning in our experiments.

3.1.2 Unified Architecture Training

The search space of supernet in GAIA consists of network depth, layer width, and input scale, which are factors mostly



(a) (b) (c)

Figure 2: Model ranking in different settings. (a) and (b) show the ranking of models with different and similar FLOPs respectively. (c) shows the ranking of "fast-finetuned" models with similar FLOPs but different input scales and depths.

correlated with computation budgets. We follow the regime of weight sharing learning mentioned in [65], that only layers and channels of the lower index are kept when sampling subnets out of supernet. In this way, models of different architectures could be optimized collectively. However, there exists a disturbing problem about weight sharing learning, that different sub-networks are interfering with each other throughout the training process of supernet.

To mitigate the problem, a method named *progressive shrinking* (PS) is proposed in OFA [8], that the search space is gradually loosed, from kernel size to depth, and finally to width. Although it has shown significant effectiveness in task of image classification, it seems not to fit well in object detection. The reason underlying has been mentioned in previous works [40, 15] that the effective receptive field of network matters in object detection, thus a reasonable search space is required to cover a wide range of network depth. The regime that shrinks search space in order of different aspects might not be able to alleviate the interference among models with huge variation in depth, as in this paper the depth of a single stage may vary from 12 to 87. In addition, it is crucial to keep a good compatibility between input scale and depth as mentioned in [55] when sampling models during upstream training.

Taking these understandings into consideration, we propose a training scheme named *anchor-based progressive shrinking* (ABPS) that shrinks multiple search dimensions step by step. To begin with, we select a model anchor and build a subordinate search space surrounding it while keeping the basic compatibility of depth and input scales. After training for a while, we shrink the model anchor, then finetune on the new search space surrounding the model anchor. We repeat the process several times until the entire search space covers a wide latency range.

3.2. Task-specific Adaptation

3.2.1 Task-specific Architecture Selection (TSAS)

After training supernet on the unified data pool, one needs to select high-quality models based on domains of interests and constraints on computation budgets. There two reasons making the architecture selection difficult: First, evaluating more than 300K candidates on each downstream task is prohibitively expensive. Besides, directly evaluate models sampled from supernet may not reflect the true quality of architecture , which is an avoided issue in methods [65, 8]. In this part, we delve into the model ranking correlation and seek for a reliable selection regime that exhibits strong ranking ability. The Kendall Tau [32] index is applied to analyze the ranking correlations quantitatively.

To begin with, we focus on how the ranking correlation behaves among models with different FLOPs, as in most cases, models with larger FLOPs tend to have better precisions if they share similar architecture. Given a search space surrounding a randomly picked model anchor, we split subnets into groups based on BFLOPs, and study the model ranking across different FLOPs. In each group, we randomly evaluate 100 models and pick the best-performing models as the representative. We finetune these models with the standard $1\times$ scheduler and take the results as ranking references. Then we apply a $0.2\times$ "fast-finetuning" scheduler on them and compare the ranking ability with direct evaluation. As shown in Figure 2a, the correlations between results of direct evaluation and $1\times$ are extremely weak and the Kendall Tau is only 0.18, while fast-finetuning dramatically strengthen the correlations and the Kendall Tau is boosted from 0.18 to 0.85. Similar phenomenon is observed when models share the similar FLOPs, as shown in Figure 2b. Thus in GAIA, we rely on results of the fast-finetuned models as indicators for model selection instead of the direct evaluation results, which are totally unreliable.

Since applying fast-finetuning on 300K models is still costly, we need to narrow down the search space to a tiny but instructive one. To this end, we randomly sample models of different input scales and depths around a fixed FLOPs, and apply the fast-finetuning to find the decisive factors. As illustrated in Figure 2c, models with the same input scales and depths tend to attain similar precisions while models with either different input scales or different depths have more diverse precisions. Thus in GAIA,

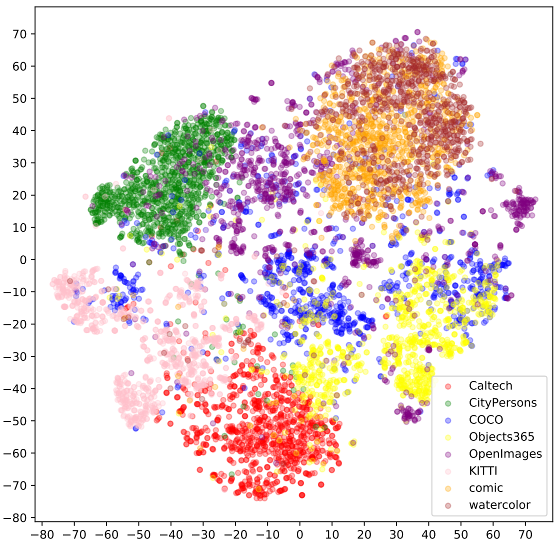


Figure 3: Visualization of output features in *fc7* layer from GAIA using t-SNE. We take *person* category as an example and sample 1000 images for each dataset. The color represents the dataset for each image.

we apply a two-step search scheme to robustly find high-quality models. In the first step, we randomly sample K (usually 5) models as a group for each combination of input scales and total depths in each sub search space while keep these models satisfying the given constraints. We directly evaluate these models and pick the best-performing model in each group. In the second step, we keep the top 50% models among the picked, fast-finetune them and select the best architecture based on the fast-finetuning results.

3.2.2 Task-specific Data Selection (TSDS)

Since GAIA has been able to produce a series of superior architectures according to downstream tasks, we naturally want to take a step further: Is it possible that GAIA could select relevant data to support downstream tasks even more? This is particularly favorable in data-scarce scenarios where very few datapoints are available for the tasks. Given massive upstream datasets D_s and a specific downstream dataset D_t, data selection aims to find the subset $D_s^* \in \mathcal{P}\left(D_s\right)$, where $\mathcal{P}\left(D_s\right)$ is the power set of D_s, such that D_s^* minimizes the risk of a model $\mathcal{F}$ on the task dataset D_t:

$$D_s^* = \underset{D_s^* \in \mathcal{P}(D_s)}{\arg\min}\ \mathbb{E}_{D_t}\left[\mathcal{F}\left(D_t \cup D_s^*\right)\right], \tag{1}$$

where $\mathcal{F}\left(D_s^* \cup D_t\right)$ denotes the model $\mathcal{F}$ trained on the union of D_t and D_s^*, the $\mathbb{E}_{D_t}$ denotes the risk on the validation set of D_t.

Ideally, using more data yields better performance. However, the assumption does not always stand in transfer learning scenario due to the domain gap. The task relationship could be roughly divided into three types: positive correlation, negative correlation, and unrelatedness. If the domain between the source and target tasks are similar, the relationship tends to be positive. If two tasks are dissimilar, the source task may not boost the target task, even hurt the performance. Hence, the essential problem is how to close the gap between two tasks. Some works focus on adversarial training [12] and data synthesis [31], while we turn to exploit the large-scale public object detection datasets by data selection.

We find that models of GAIA under large-scale pre-training implicitly learn the domain representation in both upstream and downstream datasets, suggesting that it is possible to measure the domain similarity. We visualize the output features of *person* in *fc7* layer from GAIA in Figure 3, and the following characteristics can be observed. First, features from common datasets like COCO, Objects365, and Open Images, are apt to occupy diverging space, while features from specific datasets like Caltech, CityPersons, Comic, and Watercolor usually lie in a small and compact space respectively. Second, feature spaces of similar datasets are close to each other. For instance, the Comic shares almost the same space with Watercolor. Third, there are noticeable overlaps of feature space between common and specific datasets, indicating that common datasets may contain similar data as those in specific datasets.

With the ability of domain clustering powered by GAIA, we propose to produce the D_s^* by data selection. To be specific, for each image in $D_s = \{I_{s1}, I_{s2}, \cdots, I_{sP}\}$ and $D_t = \{I_{t1}, I_{t2}, \cdots, I_{tQ}\}$ we compute a represent vector for its every category, where P and Q are the number of images in D_s and D_t, respectively. Those categories are defined in the unified label space $\cup L = \{c_{\cup 1}, c_{\cup 2}, \cdots\}$. For each image and each category, the represent vector $V(I_i, c_{\cup p})$ could be obtained by averaging the output features in *fc7* layer from all instances. Then we find the most relevant data for each category by using two alternative strategies based on the cosine distances between $V(I_{si}, c_{\cup p})$ and $V(I_{tj}, c_{\cup p})$:

- **top-k**. Choosing top-k images from D_s for each I_{tj}.
- **most-similar**. Retrieving the most similar images in all $P \times Q$ relation pairs.

We collect relevant images until the total number meets the expectation, for instance, $|D_s^*| = 1000$.

4. Experiments

4.1. Datasets

Upstream Datasets. GAIA is trained on the union of Open Images [34], Objects365 [53], MS-COCO [38], Caltech [18], and CityPersons [66] under the unified label space. Open Images, Objects365, and MS-COCO are the common detection datasets, containing 500, 365, and 80 categories, respectively. We use the Open Images 2019 challenge split with 1.7 millions images as train set and 40k images as validation. For Objects365, following the official protocol, we use 600k for training and 30k for validation. For COCO, following the standard protocol, 115k subset is used to train, and 5k is used as *minival*. Caltech and CityPersons are two specific datasets for pedestrian detection, containing 42k and 3k train set, 4k and 0.5k validation, respectively. These upstream datasets result in a unified label space with 700 categories.

Table 1: Setting of search space that surrounds different model anchors. Each architecture parameter is sampled from minimum to maximum with certain step. $W[0]$ denotes the width of the network stem.

	AR50	AR77	AR101
D_{min}	[2,2,4,2]	[2,2,11,2]	[2,2,17,2]
D_{anchor}	[3,4,6,3]	[3,4,15,3]	[3,4,23,3]
D_{max}	[4,6,8,4]	[4,6,19,4]	[4,6,29,4]
D_{step}	[1,2,2,1]	[1,2,4,1]	[1,2,6,1]
W_{min}	[32,48,96,192,384]		
W_{anchor}	[64,64,128,256,512]		
W_{max}	[64,80,160,320,640]		
W_{step}	[16,16,32,64,128]		
S_{min}	400	480	560
S_{anchor}	560	640	720
S_{max}	720	800	880
S_{step}	80	80	80

Downstream Tasks. We conduct extensive experiments on Universal Object Detection Benchmark (UODB) [3]. Note that UODB is not used in upstream training phase and consists of 10 diverse sub-datasets: Pascal VOC [19], WiderFace [63], KITTI [20], LISA [44], DOTA [59], Watercolor [31], Clipart [31], Comic [31], Kitchen [21] and DeepLesions [61]. We follow the official data split and metric for all above datasets. Details of each dataset are documented in the Supplementary Material section A.

4.2. Implementation Details

4.2.1 Architecture Space

In all experiments, we use the Faster RCNN [52] with FPN [36] as the base framework. ResNet [29] is adopted as the network prototype in GAIA, and both the task-agnostic unification and task-specific adaptation are applied on it. The reason for choosing ResNet is that ResNet is still the most paradigmatic network architecture in object detection and is especially popular in real-world applications. For *anchor-based progressive shrinking* (ABPS) during training stage, we choose 3 model anchors in our experiments for simplicity and the corresponding search spaces are shown in Table 1. We refer to these model anchors as AR50, AR77 and AR101. In each iteration during training, we randomly pick an available model that obeys one of the prescribed rules in search space for optimization. The rules includes: models with the maximum depth, models with the minimum width and other likenesses. Details of more prescribed rules are shown in the Supplementary Material section B.

4.2.2 Upstream Training

The training of the supernet starts with AR101 for 24 epochs, then the search space shrinks to AR77 and finally to AR50, and we finetune each search space for 13 epochs. Each time we shrink search space, we add a single epoch of warm-up which is essential. We follow the training setting defined in Detectron2 [58] except the input scales are adjusted to $[S_{min} : S_{step} : S_{max}]$ for each sub search space. By convention, no other data augmentation is applied except the standard horizontal flipping. The initial learning rate is set 0.00125 per image and is decayed by a factor of 10 at 16 and 21 epoch. For finetuning in the shrunk search space, the learning rate restarts with the initial value and is decayed by a factor of 10 at 8 and 11 epoch. The supernet of GAIA is trained from scratch, thus we apply the sync-BN which is essential as proved in [27]. We use SGD to optimize the training loss with 0.9 momentum and 0.0001 weight decay. IoU-sampling [45] is applied to make sure that subnets could learn balanced knowledge across object scales. Since the supernet of GAIA are trained to learn a great number of categories, we find that the gradients derived from class-specific supervision are diluted over time compared to that from class-agnostic supervision. To alleviate the issue, we multiply the loss weights of head by a factor of 5.

4.2.3 Downstream Fine-Tuning

Given a downstream dataset and categories of interests, we apply the TSAS to find the most appropriate architecture and extract corresponding weights from supernet. Besides, we conduct a weight surgery on the weights of the last *fc* layer in head to focus on related categories. For categories included in the unified label space of GAIA, we keep the pre-trained weights. For categories that are not in the label space, we find their closest neighbors based on the word vectors, and take the weights for initialization.

The initial learning rate of downstream finetuning is set 0.0001875 per image and is decayed by a factor of 5 after 8 and 11 epoch. We train for 13 epochs in total and all the other configuration is consistent with the training setting of supernet. As for the fast-finetuning in TSAS, we use a warm-up for one epoch and train for 2 epochs. The initial learning rate of downstream finetuning is set 0.0001875 per image and is decayed by a factor of 10 after the first epoch.

4.3. Results on COCO Dataset

Taking COCO dataset as an example, we demonstrate how GAIA is capable of generating high-quality mod-

Table 2: Results of models with different FLOPs on COCO *minival*. All models are trained with 1× scheduler if not specified. Models with "*" are trained with 3× scheduler. The FLOPs is calculated with input size of [3,Scale,Scale], and the latency is measured on the entire validation set of COCO on V100 with batchsize of 1.

Group	Pre-train	Scale	Depth	Width	FLOPs	Latency	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet50	ImageNet	800	[3,4,6,3]	[64, 64, 128, 256, 512]	137.4B	39ms	37.08	59.52	39.74	22.63	40.92	46.51
ResNet101		800	[3,4,23,3]	[64, 64, 128, 256, 512]	188.5B	51ms	39.41	61.62	43.04	24.42	43.61	50.53
ResNet50$*$	ImageNet	800	[3,4,6,3]	[64, 64, 128, 256, 512]	137.4B	39ms	39.78	60.47	42.23	23.27	42.57	50.04
ResNet101$*$		800	[3,4,23,3]	[64, 64, 128, 256, 512]	188.5B	51ms	41.75	62.24	45.77	25.28	45.61	54.39
ResNet50	GAIA	800	[3,4,6,3]	[64, 64, 128, 256, 512]	137.4B	39ms	42.91	64.43	47.12	28.01	47.07	53.81
ResNet101		800	[3,4,23,3]	[64, 64, 128, 256, 512]	188.5B	51ms	46.07	67.28	50.64	29.31	50.45	58.50
30-45B	GAIA	400	[4,4,8,4]	[48,48,96,192,384]	44.3B	17ms	38.21	58.73	40.62	18.15	42.18	54.79
45-60B		480	[4,6,8,4]	[48,48,96,256,384]	59.4B	19ms	40.44	61.07	43.77	21.65	43.66	55.96
60-75B		560	[4,6,15,4]	[48,80,96,192,512]	74.4B	26ms	42.59	64.01	46.07	24.82	46.78	57.42
75-90B		560	[4,6,21,4]	[64,80,96,192,512]	88.1B	28ms	43.60	64.81	47.34	24.97	47.85	58.62
90-105B		560	[4,6,21,4]	[64,80,160,192,512]	101.1B	30ms	44.05	65.15	47.91	25.55	48.69	59.09
105-120B		640	[4,6,21,4]	[64,80,160,192,512]	119.2B	33ms	44.46	65.71	47.49	26.77	48.31	57.21
120-135B		720	[3,4,23,3]	[64,64,128,192,640]	133.9B	38ms	45.27	66.64	49.47	27.97	49.44	57.89
135-150B		800	[4,6,23,3]	[48,48,96,192,640]	149.1B	44ms	45.55	66.89	50.08	28.96	49.94	58.01
150-180B		800	[3,4,23,3]	[64,64,96,256,512]	178.7B	47ms	46.02	67.40	50.52	28.80	50.37	59.01
180-210B		880	[3,4,25,4]	[48,48,96,256,384]	209.8B	53ms	46.41	67.98	50.88	29.83	50.83	58.32

Table 3: Results of models with different whistles and bells on COCO *minival*. All models are trained with 1× scheduler. "ImageNet" and "GAIA" denote the pre-training datasets respectively. The architecture of GAIA-TSAS in table is $\{D$: [4,6,23,3], W:[48,48,128,192,384], S: 800$\}$.

Backbone	ImageNet	GAIA	DCN	Cascade-Head	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Latency
ResNet50	✓				37.0	59.5	39.7	22.6	40.9	46.5	39ms
		✓			42.9	64.4	47.1	28.0	47.1	53.8	
	✓		✓		40.8	63.7	44.3	25.6	44.4	52.3	44ms
		✓	✓		44.8	66.4	49.2	29.0	48.6	57.3	
	✓			✓	40.9	59.7	44.4	24.1	44.5	52.3	47ms
		✓		✓	45.8	64.6	50.1	29.7	49.6	58.4	
	✓		✓	✓	44.9	64.8	49.2	27.2	49.2	59.4	53ms
		✓	✓	✓	47.9	66.9	52.6	31.7	51.5	61.9	
GAIA-TSAS		✓	✓	✓	49.1	68.0	54.0	30.5	53.4	65.0	55ms

Table 4: Results of GAIA on other upstream datasets using ResNet50. We follow the official protocol, *i.e.*, mmAP for Objects365, mAP for Open Images, and MR^{-2}(Miss Rate, lower is better) for Caltech and CityPersons. "TSAS" denotes that an architecture with similar latency as ResNet50 is applied.

Dataset	Pre-train		TSAS	Metric
	ImageNet	GAIA		
Objects365	✓			21.5
		✓		24.0
		✓	✓	**26.1**
Open Images	✓			52.2
		✓		59.5
		✓	✓	**62.4**
Caltech	✓			5.5
		✓		2.2
		✓	✓	**1.7**
CityPersons	✓			14.7
		✓		11.1
		✓	✓	**10.4**

els powered by data unification and architecture adaptation. First, we compare the results of the vanilla ResNet50 and ResNet101 trained with different weight initialization. As shown in Table 2, models with GAIA pre-training yield huge improvements over models with ImageNet pre-training, which is 5.83% for ResNet50 and 6.66% for ResNet101. Since the data of COCO dataset are included in the data pool of supernet, we also compare the results of GAIA with models trained for 3× with ImageNet pre-training for fairness. The improvements are still considerable(+3.23% and +4.22%), indicating that data from other sources are of great help. With the help of TSAS, the improvements are further boosted to 5.49% and 4.66%.

In addition, GAIA is able to produce models across a wide latency range efficiently. Since there are no pre-trained weights for models other than ResNet50 and ResNet101, models with customized architecture are obliged to be trained from scratch. Within 1× the training time, models trained from GAIA outperform models trained from scratch 12.67% on average as shown in Figure 1b.

We also conduct experiments to see whether models generated from GAIA are compatible with whistles and bells. We select DCN [15] and Cascade-RCNN [10] which are two of the most effective methods in object detection. As

Table 5: Results of GAIA on UOBD datasets. † stands for our re-implemented baseline. "COCO" and "GAIA" denote the pre-training datasets, respectively.

Method	KITTI	VOC	WiderFace	LISA	Kitchen	DOTA	DeepLesion	Comic	Clipart	Watercolor	Avg.
Baseline [3]	64.3	78.5	48.8	88.3	87.7	57.5	51.2	45.8	32.1	52.6	60.7
DA [3]	68.0	82.4	51.3	87.6	90.0	56.3	53.4	53.4	55.8	60.6	65.9
Baseline†	67.1	81.5	62.1	90.0	89.5	68.3	57.4	45.5	31.2	53.4	64.6
COCO	70.2	84.2	61.4	88.6	88.7	64.7	52.4	51.6	56.9	56.1	67.5
GAIA	72.9	85.9	62.6	91.2	89.8	69.2	59.4	57.0	67.9	63.5	71.9
GAIA-TSAS	**75.6**	**87.4**	**62.7**	**92.1**	**90.1**	**70.8**	**62.1**	**61.1**	**72.2**	**69.7**	**74.4**

Table 6: Results of task-specific data selection (TSDS). Three strategies, including random, top-k, and most-similar, are used.

pre-train	TSDS	KITTI	Comic	Watercolor	Average
COCO	-	44.9	36.4	45.3	42.2
GAIA	-	45.4	46.9	51.1	47.8
	random	40.3	46.1	49.9	45.4
	top-k	46.7	48.0	51.2	48.6
	most-similar	**48.6**	**48.2**	**53.6**	**50.1**

displayed in Table 3, the gain from GAIA is congruent with these methods. With the help of TSAS, we achieve an AP of 49.1% with only $1\times$ scheduler while keeping the latency almost unchanged.

4.4. Results on Other Upstream Datasets

We also provide the results in other seen datasets in Table 4. Objects365, Open Images, Caltech, and CityPersons are parts of the pre-training datasets for GAIA. We build their baseline from widely-used ImageNet pre-training, and train them individually until they converge and the performances stop growing. Then we apply downstream finetuning for all datasets from GAIA pre-training. We find that GAIA outperforms the baseline in these four datasets by 2.5%, 8.8%, 3.3%, and 3.6%, respectively. The significant improvement in Open Images is mainly caused by the gain from those long-tail categories. Moreover, with the ability of task-specific architecture selection (TSAS), GAIA yields additional $0.5\% \sim 2.9\%$ improvements. Detailed comparisons with state-of-the-art are available in the supplementary material.

4.5. Transfer Learning on Downstream Datasets

To evaluate the generalizability of GAIA, we conduct experiments on downstream datasets from UODB [3]. Table 5 reports the mAP and GAIA shows great success on various datasets. It can be seen that our re-implemented ImageNet pre-trained baseline achieves 64.6% mAP on average which is 3.9% higher than the UODB baseline, and the COCO pre-training brings 2.9% improvements. Therefore, these counterparts are strong enough to validate the effectiveness of GAIA. Empowered by the unified label space and large-scale dataset pre-training, GAIA can steadily improve the performance by 4.4% on average. Furthermore, TSAS of GAIA yields another 2.5% improvements overall.

4.6. Data Selection for Data-scarce Scenarios

In this section, we evaluate our proposed task-specific data selection (TSDS) in few datapoints settings. From the unseen UODB data source, we pick three datasets with small label space for convenience. We start our investigation with 10 images per datasets as the baseline. The images are randomly sampled from the corresponding dataset while making each category have at least 2 images. As shown in Table 6, the average mAP of GAIA without TSDS outperforms the COCO pre-trained baseline by 5.6%. With data selection strategies, it is obvious that choosing relevant data is crucial. Randomly selecting data is harmful to the performance because it introduces much out-domain data and disturbs the target domain learning. It is also nice to see that top-k and most-similar strategies bring additional gains by $0.8\% \sim 2.3\%$ on average, showing the benefit of large-scale pre-trained GAIA for the data selection.

5. Conclusion

In this work, we revisit the efficacious generalizability of transfer learning and its limitation to downstream customization, and harmonize the gap between generalist and specialist models. We present a transfer learning system named GAIA, which could automatically and efficiently give birth to specialized solutions according to heterogeneous downstream needs. Constructing GAIA involves a heavy workload, thus it leaves much space for future work to make it better. It could also be extended to more architectures like MobileNetV3, more frameworks like GAIA-YOLO, and more vision tasks like GAIA-Seg. In the end, we sincerely hope that our work could substantially help more practitioners in the community of object detection.

6. Acknowledgements

This work was supported in part by the Major Project for New Generation of AI (No.2018AAA0100400), the National Natural Science Foundation of China (No. 61836014, No. 61773375).

References

[1] Robust vision challenge. http://www.robustvision.net/, 2020. 2, 3

[2] Roee Aharoni and Yoav Goldberg. Unsupervised domain clusters in pretrained language models. In *ACL*, 2020. 3

[3] Xudong ang, Zhaowei Cai, Dashan Gao, and Nuno Vasconcelos. Towards universal object detection by domain attention. *CVPR*, pages 7281–7290, 2019. 2, 6, 8

[4] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. In *ICLR*, 2017. 3

[5] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *ICML*, 2018. 2

[6] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020. 2

[7] Han Cai, Tianyao Chen, Weinan Zhang, Yong Yu, and Jun Wang. Efficient architecture search by network transformation. In *AAAI*, 2018. 3

[8] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once for all: Train one network and specialize it for efficient deployment. In *ICLR*, 2020. 2, 3, 4

[9] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *ICLR*, 2019. 3

[10] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *CVPR*, pages 6154–6162, 2018. 2, 7

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 40:834–848, 2018. 3

[12] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. *CVPR*, pages 3339–3348, 2018. 5

[13] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. Detnas: Backbone search for object detection. In *NeurIPS*, 2019. 3

[14] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. *CVPR*, pages 4109–4118, 2018. 3

[15] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. *ICCV*, pages 764–773, 2017. 2, 4, 7

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 2

[18] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE TPAMI*, 34:743–761, 2012. 5

[19] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88:303–338, 2009. 6

[20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. *CVPR*, pages 3354–3361, 2012. 6

[21] Georgios Georgakis, Md Alimoor Reza, Arsalan Mousavian, Phi-Hung Le, and Jana Košecká. Multiview rgb-d dataset for object instance detection. *3DV*, pages 426–434, 2016. 6

[22] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. *CVPR*, pages 7029–7038, 2019. 3

[23] Ross Girshick. Fast r-cnn. *ICCV*, pages 1440–1448, 2015. 2, 3

[24] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR*, pages 580–587, 2014. 2, 3

[25] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*, 2020. 3

[26] Irtiza Hasan, Shengcai Liao, Jinpeng Li, Saad Ullah Akram, and Ling Shao. Pedestrian detection: The elephant in the room. *ArXiv*, abs/2003.08799, 2020. 3

[27] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. *ICCV*, pages 4917–4926, 2019. 6

[28] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. *IEEE TPAMI*, 42:386–397, 2020. 2

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2016. 6

[30] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016. 2

[31] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. *CVPR*, pages 5001–5009, 2018. 3, 5, 6

[32] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938. 2, 4

[33] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, 2020. 2, 3

[34] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *IJCV*, 128:1956–1981, 2020. 2, 5

[35] Hengduo Li, Bharat Singh, Mahyar Najibi, Zuxuan Wu, and Larry S Davis. An analysis of pre-training on object detection. *ArXiv*, abs/1904.05871, 2019. 3

[36] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid

networks for object detection. *CVPR*, pages 936–944, 2017. 2, 6

[37] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *ICCV*, pages 2999–3007, 2017. 3

[38] 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. *ECCV*, pages 740–755, 2014. 5

[39] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *ArXiv*, abs/1806.09055, 2019. 2, 3

[40] Songtao Liu, Di Huang, et al. Receptive field block net for accurate and fast object detection. In *ECCV*, 2018. 4

[41] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. *ECCV*, 2016. 3

[42] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, pages 3431–3440, 2015. 3

[43] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 2, 3

[44] Andreas Mogelmose, Mohan Manubhai Trivedi, and Thomas B Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE TITS*, 13:1484–1497, 2012. 6

[45] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *CVPR*, 2019. 6

[46] Junran Peng, Xingyuan Bu, Ming Sun, Zhaoxiang Zhang, Tieniu Tan, and Junjie Yan. Large-scale object detection in the wild from imbalanced multi-labels. *CVPR*, pages 9706–9715, 2020. 2

[47] Junran Peng, Ming Sun, ZHAO-XIANG ZHANG, Tieniu Tan, and Junjie Yan. Efficient neural architecture transformation search in channel-level for object detection. In *NeurIPS*, 2019. 3

[48] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *ICML*, 2018. 2

[49] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683, 2019. 2

[50] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CVPR*, pages 779–788, 2016. 3

[51] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *LREC Workshop*, pages 45–50, 2010. 3

[52] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE TPAMI*, 39:1137–1149, 2015. 2, 3, 6

[53] 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 *ICCV*, pages 8430–8439, 2019. 2, 3, 5

[54] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. *ICCV*, pages 843–852, 2017. 3

[55] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. *CVPR*, pages 10778–10787, 2020. 3, 4

[56] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *CVPR*, pages 1653–1660, 2014. 3

[57] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE TEVC*, 1:67–82, 1997. 2

[58] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[59] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. *CVPR*, pages 3974–3983, 2018. 6

[60] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *ArXiv*, abs/1905.00546, 2019. 3

[61] Ke Yan, Xiaosong Wang, Le Lu, Ling Zhang, Adam P Harrison, Mohammadhadi Bagheri, and Ronald M Summers. Deep lesion graphs in the wild: Relationship learning and organization of significant radiology image findings in a diverse large-scale lesion database. *CVPR*, pages 9261–9270, 2018. 6

[62] Xi Yan, David Acuna, and Sanja Fidler. Neural data server: A large-scale search engine for transfer learning data. In *CVPR*, pages 3893–3902, 2020. 3

[63] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. *CVPR*, pages 5525–5533, 2016. 6

[64] Jiahui Yu and Thomas S Huang. Universally slimmable networks and improved training techniques. *ICCV*, pages 1803–1811, 2019. 2, 3

[65] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. In *ECCV*, 2020. 2, 3, 4

[66] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. Citypersons: A diverse dataset for pedestrian detection. *CVPR*, pages 4457–4465, 2017. 5

[67] Xiangyun Zhao, Samuel Schulter, Gaurav Sharma, Yi-Hsuan Tsai, Manmohan Chandraker, and Ying Wu. Object detection with a unified label space from multiple datasets. In *ECCV*, 2020. 3

[68] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *ICLR*, 2017. 3

[69] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. *CVPR*, pages 8697–8710, 2018. 3