

# UFO: Unified Feature Optimization

Teng Xi\*, Yifan Sun\*, Deli Yu\*, Bi Li\*, Nan Peng\*, Gang Zhang\*\*,  
Xinyu Zhang, Zhigang Wang, Jinwen Chen, Jian Wang, Lufei Liu,  
Haocheng Feng, Junyu Han, Jingtuo Liu, Errui Ding, and Jingdong Wang

Baidu Inc.

Code: <https://github.com/PaddlePaddle/VIMER/tree/main/UFO>

**Abstract.** This paper proposes a novel Unified Feature Optimization (UFO) paradigm for training and deploying deep models under real-world and large-scale scenarios, which requires a collection of multiple AI functions. UFO aims to benefit each single task with a large-scale pre-training on all tasks. Compared with existing foundation models, UFO has two points of emphasis, *i.e.*, relatively smaller model size and NO adaptation cost: 1) UFO squeezes a wide range of tasks into a moderate-sized unified model in a multi-task learning manner and further trims the model size when transferred to down-stream tasks. 2) UFO does not emphasize transfer to novel tasks. Instead, it aims to make the trimmed model dedicated for one or more already-seen task. To this end, it directly selects partial modules in the unified model, requiring completely NO adaptation cost. With these two characteristics, UFO provides great convenience for flexible deployment, while maintaining the benefits of large-scale pretraining. A key merit of UFO is that the trimming process not only reduces the model size and inference consumption, but also even improves the accuracy on certain tasks. Specifically, UFO considers the multi-task training and brings a two-fold impact on the unified model: some closely-related tasks have mutual benefits, while some tasks have conflicts against each other. UFO manages to reduce the conflicts and preserve the mutual benefits through a novel Network Architecture Search (NAS) method. Experiments on a wide range of deep representation learning tasks (*i.e.*, face recognition, person re-identification, vehicle re-identification and product retrieval) show that the model trimmed from UFO achieves higher accuracy than its single-task-trained counterpart and yet has smaller model size, validating the concept of UFO. Besides, UFO also supported the release of **17 billion parameters computer vision (CV) foundation model** which is the largest CV model in the industry.

**Keywords:** Train and Deploy; Foundation Model; Multi-task Learning; Unified Feature Optimization;

---

\* Equal contribution

\*\* Corresponding to {zhanggang03,xiteng01}@baidu.com

## 1 Introduction

Training and deploying are two essential procedures for artificial intelligence (AI) applications based on deep learning. A realistic AI system usually consists of multiple tasks. The naive train-and-deploy strategy is to train a respective deep model on each single sub-task for individual deployment. Given that some sub-tasks are actually correlated, this naive strategy wastes their mutual benefits. A feasible approach to benefit individual tasks with the large-scale multi-task data is the foundation model. In this paper, we refer the foundation model as “a model that is trained on broad data at scale and can be adapted to a wide range of downstream tasks”, according to [3].

However, foundation model has some burden for deployment, *e.g.*, it maintains the huge foundation model size and requires additional adaptation costs when transferred to down-stream tasks.

This paper presents a novel train-and-deploy paradigm, named Unified Feature Optimization (UFO), to benefit down-stream tasks with large-scale multi-task pretraining. Compared to foundation model, UFO has two different points of emphasis, *i.e.*, relatively smaller model size and NO adaptation cost. **1) *Small model size.*** UFO does not use a tremendous network. Instead, it squeezes a wide range of tasks into a moderate-sized unified model, and further trims the model size for down-stream applications, so that the inference will be more efficient. **2) *No adaptation cost.*** UFO does not emphasize transferring to novel tasks. Instead, it aims to make the trimmed model dedicated for already-seen sub-tasks. Without fine-tuning or prompt-based learning, UFO directly selects partial components from the already-learned unified model and thus requires completely no adaptation cost.

With the advantages of small model size and no adaptation cost, UFO provides great convenience for flexible deployment while maintaining the benefits of large-scale pretraining. Although the advantage of no adaptation cost is constrained to the already-seen sub-tasks, it does compromise great benefits for realistic AI development. For example, in the smart city prototype, like vision-based smart city, the system needs the collaboration of face, body and car to provide comprehensive understanding of the state of the city. Moreover, in spite that UFO lays no emphasis on the mode of transferring to novel down-stream tasks, it is compatible to this mode through existing foundation model techniques, which is not the major concern of this paper. Given their orthogonal advantages, we believe UFO and foundation model can well co-operate with each other to bring another wave of development.

As an early exploration, this paper presents the concept of UFO with focus on deep representation learning, as shown in Fig. 1. Deep representation learning is fundamental for a lot of AI applications, *e.g.*, face recognition [2,24,7], person / vehicle re-identification [19,19,18,22,17] and fine-grained image retrieval [26]. We base our UFO on the vision transformer (ViT) [10] architecture. UFO first trains a unified model (*i.e.*, the supernet) on a variety of deep representation tasks in a multi-task learning manner. Afterwards, UFO learns to trim the supernet to get a dedicated sub-net for partial sub-tasks. Given a ViT backbone, the trimming

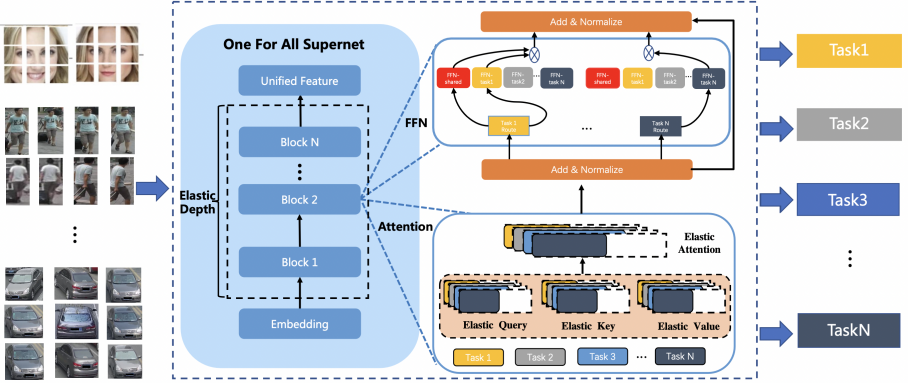


Fig. 1. Overview of the UFO paradigm.

object can be sub-block of the transformer, attention heads and FFN channels from coarse granularity to fine granularity, as illustrated in Fig. 1. Moreover, UFO integrates another trimming strategy at the FFN path level. Following [12], UFO uses multiple FFN paths in parallel when training the supernet and allows trimming some FFN paths for down-stream tasks. Although these trimming strategies are popular, UFO is the first to integrate them and thus provides great trimming flexibility.

An important advantage of UFO is that the trimming process not only reduces the model size and inference consumption, but also improves the accuracy on its dedicated sub-tasks. It is non-trivial because trimming the model (without further fine-tuning) usually compromises the accuracy. To this end, UFO considers that the multi-task training brings two-fold impacts on the supernet. On the one hand, some tasks are closely related to each other and thus have mutual benefits. On the other hand, some tasks have significant divergence and thus have mutual conflicts. During the trimming, UFO manages to reduce the conflicts and to preserve the mutual benefits through a novel Network Architecture Search (NAS) method. Specifically, we design a search space for the UFO, which first introduces FFN paths together with the supernet. Accordingly, we propose an end-to-end training strategy for UFO, which is different from previous multi-stage approaches [4, 20]. Meanwhile, we also propose a novel evaluation metric for UFO, which is flexible to any requirements of practical application. Experiments on a wide range of deep representation learning tasks show that UFO achieves higher accuracy with the smaller trimmed model than the single-task-trained counterpart. It confirms that while UFO gains the additional advantage of flexible deployment, it maintains the benefits of large-scale pretraining.

The contributions of the paper are summarized as follows:

- We propose a novel train-and-deploy paradigm, named Unified Feature Optimization (UFO), to benefit down-stream tasks with large-scale pretraining.

UFO emphasizes the advantage of small model size and no adaptation cost, which significantly promotes flexible deployment.

- We propose a novel trimming process in UFO, dedicated to preserve the mutual benefits and discard the mutual conflicts from the multi-task unified model by the proposed NAS method.
- We propose a novel evaluation metric to measure the correlations among tasks, which provides basic and effective analyses for the trimming process.
- We experiment on 10+ benchmarks from face, person, vehicle and product. Comprehensive analyses and extensive experiments clearly show the effectiveness of our UFO.

## 2 Related Work

The development of smart city has led to significant demand on the optimizations of multiple objectives to facility integrated solutions of diverse, real-world problems. With an overall increase in number of models and tasks, significant computing and inference cost are required for deploying specific models for specific tasks, especially deployed on embedded sensors or devices where computational and power resources may be limited. One way to solve this problem is the development of foundation models, which refer to models trained from broad data at scale that is capable of being adapted to a wide range of down-stream tasks. Existing works try to overcome these challenges from the following two aspects.

### 2.1 Training strategy

Tuning weights of different task losses is an effective method. Kendall et al. [23] propose a principled approach to tune the weights of multiple loss functions by considering the homoscedastic uncertainty of each task. Dynamic Task Prioritization [14] automatically prioritizes more difficult tasks by adaptively adjusting the mixing weight of each task’s loss objective. Other works adopt gradient-based methods to combat the challenge. GradNorm [6] automatically balances the training of different task losses in deep multi-task models by dynamically tuning their gradient magnitudes. Sener et al. [37] explicitly cast multi-task learning as gradient-based multi-objective optimization, with the overall objective of finding a Pareto optimal solution to minimizing all task losses. Based on the observation that models with lower variance in the angles between task gradients perform better, Suteu et al. [40] propose a novel gradient regularization of enforcing nearly orthogonal gradients. To avoid the interference of gradients from different losses, PCGrad [43] projects a task’s gradient onto the normal plane of the gradient of other tasks that have a conflicting gradient.

In contrast to these methods, our method designs a novel model structure, which adaptively specifies correlations or conflicts among all tasks, and obtains competitive results even with ordinary training strategy.



## 2.2 Model structure

Some works [11,34,29,13] adopts the manner of soft parameter sharing. They allow each task to have separate model and parameters, but enforce each model can access the information inside other models by regularizers [11,34] or NAS-searched structures [13].

Other works [31,33,39,30] use a shared part of backbone parameters with task-specific modules, which is called hard parameter sharing. The first five convolutional layers are shared and task-specific fully-connected layers are used for each task in the method of Deep Relationship Networks[31]. Lu et al. [33] starts with a thin network and dynamically grows it during the training phase by creating new branches for tasks. Besides the area of computer vision, [39,30] use shared encoders with task-specific layers across multiple NLP tasks.

Beyond the two kind of ways, Task-MOE [25] proposes an architecture which combines both the shared and task-specific modules for multi-task learning. Specifically, it shares the Self-Attention modules and selects task-specific FFN modules based on a task-level router.

All these works consider adding components by encouraging the information interaction between single tasks or introducing task-specific modules, but miss the idea of reducing modules. By contrast, we extract subnet by reducing incompatible weights and keeping complementary weights from a supernet. Similar to Task-MOE, our method also adopts task-level routers to select specific FFNs. However, our method extracts the most suitable sub-weights of Self-Attention for each task, while Task-MOE share the complete one among all tasks.

## 3 Methodology

UFO consists of two steps, *i.e.*, training a multi-task supernet, and extracting a dedicated sub-network for down-stream task deployment. Under this novel training and deploying paradigm, UFO aims to preserve the mutual benefit of multi-task pretraining and remove the mutual conflict between different tasks. To this end, we employ a Neural Architecture Search (NAS) method to search for the sub-network from the supernet. Specifically, we introduce the architecture of UFO supernet as well as its search space in Section 3.1. We note that different from the search space for single task, the UFO search space is to accommodate multiple sub-networks for various downstream tasks. Given the architecture of UFO supernet, Section 3.2 explains how to train the supernet on all the tasks in a multi-task learning manner. Finally, Section 3.3 elaborates on learning the sub-network extraction based on NAS. It allows UFO to directly extract a corresponding sub-network through architecture prediction, given the desired down-stream tasks (as well as the model size and inference speed).

### 3.1 The architecture and search space of UFO supernet

As shown in figure 1, we base the UFO supernet on the vision transformer (ViT). Since the sub-network selects partial modules from the supernet and inherits

the corresponding parameters during the deployment, it is important that the supernet provides a large space for searching and extracting the sub-networks.

Prior transformer-based NAS usually provides three searching directions, *i.e.*, elastic depths, elastic attention heads and elastic expansion ratios of the Feed Forward Networks (FFN) [25]. In addition to these commonly-used searching directions, we introduce a novel search direction, *i.e.*, flexible FFN paths. In other words, UFO combines three commonly-used search directions and a novel one, and thus provides a large searching space. Consequently, the sub networks can reduce FFN paths, FFN weights, attention weights or even the whole sub blocks of the vision transformer. We explain these searching directions in details as below.

The architecture space is consisted with a set of architectures, and is denoted as  $\mathcal{A} = \{a_1, a_2, \dots, a_{n_a}\}$ ,  $n_a = |\mathcal{A}|$ . Let  $\mathcal{H}$  be set of head numbers and  $\mathcal{M}$  be the set of mlp ratios in FFN, where  $\mathcal{H} = \{h_1, h_2, \dots, h_{n_h}\}$ ,  $n_h = |\mathcal{H}|$  and  $\mathcal{M} = \{m_1, m_2, \dots, m_{n_m}\}$ ,  $n_m = |\mathcal{M}|$ . Let  $\mathcal{T}$  be the set of target tasks, where  $\mathcal{T} = \{t_1, t_2, \dots, t_{n_t}\}$ ,  $n_t = |\mathcal{T}|$ . Let  $\mathcal{G}$  be the set of gate choice of FFN paths, where  $\mathcal{G} = \{g_0, g_1, \dots, g_{n_g}\}$ ,  $n_g = |\mathcal{G}|$ . Finally, let  $\mathcal{D} = \{0, 1\}$  be the set of drop choice to denote whether the entire layer will be dropped. Then, the search space  $\mathcal{A}$  can be denoted as follows:  $\mathcal{A} = \{[[h_1, m_1, g_1, d_1], [h_2, m_2, g_2, d_2], \dots, [h_l, m_l, g_l, d_l]], h_i \in \mathcal{H}, m_i \in \mathcal{M}, g_i \in \mathcal{G}, d_i \in \mathcal{D}, \forall i \in \{1, 2, \dots, l\}\}$ , where  $l$  is the numbers of layers. In summary,  $\mathcal{G}$  determines the FFN paths of different tasks of  $\mathcal{T}$ . Furthermore,  $\mathcal{H}$  and  $\mathcal{M}$  determine the model size of different sub networks. Besides,  $\mathcal{D}$  controls the depth of sub networks to further reduce the model size.

Given the input  $\mathbf{x}_i^t$  of task  $t$ , an arch  $a$  is sampled from  $\mathcal{A}$ , and then the consecutive blocks of the arch are computed as:

$$\begin{aligned}\hat{\mathbf{x}}_i^t &= d_l * \text{MHSA}(\text{LN}(\mathbf{x}_i^t), h_l) + \mathbf{x}_i^t \\ \mathbf{x}_{i+1}^t &= d_l * \text{FFNs}(\text{LN}(\hat{\mathbf{x}}_i^t), m_i, g_i^t) + \hat{\mathbf{x}}_i^t\end{aligned}\quad (1)$$

### 3.2 Multi-task training of the UFO supernet

In this subsection, we will describe how to train multi-task supernet. As shown in subsection 3.1, the supernet in UFO is quite different from other single-task supernets. Accordingly, the training strategy of UFO is also different in two aspects of sub-network sampling and data sampling.

**Sub-network sampling.** The sub-network sampling is involved with the sampling of  $(m_l, h_l, d_l, g_l)$ . Similar to weight entanglement mechanism [5], the weights of the arch  $a$  are shared with the weight of supernet for their common parts with respect to the sampling of  $m_l$  and  $g_l$ . However, as the supernet do not have FFN-paths in the existing training strategies [5,20,41], there are serious competitions among shared attention weights. Thus, their supernet has to be trained in a step-by-step way. In the UFO, the FFN-paths relieve the competition of shared attention. Thus, the UFO can be trained in an end-to-end way.

However, the supernet is hard to converge if we directly sample sub-networks from  $\mathcal{A}$  with respect to  $g_l$ , because the total number of FFN paths is  $|\mathcal{T}| \times (2^{|\mathcal{G}|} -$

1)<sup>*l*</sup>. Thus, we set constraint for the path gate of each task, where each task in  $\mathcal{T}$  only has 3 choices for each layer, i.e. shared FFN only, task specific FFN only or both. To be specific, we use gumbel-softmax on learn-able gate weights to sample probability distribution for task  $t$ . Thus, the output of FFNs (we ignore layer/block idx  $i$ ) can be defined as:

$$\text{FFNs}(\cdot) = p^t[0]\text{FFN}_{\text{shared}}(\cdot) + p^t[1]\text{FFN}_{\text{task-specific}}^t(\cdot) \quad (2)$$

After training, the learned gate weights determine the single choice of gates by argmax or choose both gates. In this way, the total number of FFN paths is reduced from  $|\mathcal{T}| \times (2^{|\mathcal{G}|} - 1)^l$  to  $|\mathcal{T}| \times 3^l$ .

**Data sampling** There are five existing data sampling strategies in [1]. The accumulating gradient strategy is the most promising among them. It accumulates gradients from all task data in one optimizer step, and can achieve better optimization trade-off between different tasks than other methods, e.g. task-by-task and alternating methods. Inspire by the thought, we propose a similar but different strategy of forming batch, and it is called heterogeneous batch type. To be specific, we sample some data from all tasks of  $\mathcal{T}$  to form a mini batch with a weight roughly proportional to the size of the task datasets, respectively. Then, these mini batch is concatenated into a batch data, which is feed into the backbone. Next, the obtained features are separated and feed into  $|\mathcal{T}|$  task-specific head networks, each of which is responsible for the output of a task. Finally, we calculate the loss of  $|\mathcal{T}|$  tasks, sum it up for the shared transform backbone network, and finish a backward step to obtain gradients, which are used to update the shared parameters.

### 3.3 Extracting the sub-network for down-stream task deploying

In this subsection, we will introduce how to select optimal dedicated models from supernet according to requirement of practical applications.

Our target is to find optimal architecture  $a$  of  $\mathcal{A}$  under flops and parameter constraints and the average performance is maximized.

Let  $f_t(a)$  be the performance of architecture  $a$  on task  $t$ ,  $\forall t \in \mathcal{T}$ ,  $\forall a \in \mathcal{A}$ . Then, let  $f_{\mathbf{t}}(a)$  be the performance of architecture  $a$  on task set  $\mathbf{t}$ ,  $\forall \mathbf{t} \subset \mathcal{T}$ ,  $\forall a \in \mathcal{A}$ .

Except for the extreme performances for target tasks, we also care about the generalized performances on other tasks. Thus, let  $\text{avg\_}f(\mathbf{t}, a)$  be the comprehensive performance of architecture on all tasks, where:

$$\text{avg\_}f(\mathbf{t}, a) = \lambda f_{\mathbf{t}}(a) + (1 - \lambda) f_{\mathcal{T} \setminus \mathbf{t}}(a) \quad (3)$$

$$= \lambda \sum_{t_1 \in \mathbf{t}} f_{t_1}(a) + (1 - \lambda) \sum_{t_2 \in \mathcal{T} \setminus \mathbf{t}} f_{t_2}(a) \quad (4)$$

$$, \forall \mathbf{t} \subset \mathcal{T}, \forall a \in \mathcal{A} \quad (5)$$

---

**Algorithm 1:** Multi-task Searching Algorithm (MSA)

---

**Input:**  $\mathcal{T}, \mathcal{A}, \mathbf{t}$ ;  
**Output:**  $a\_best$ ;  
1 Generate a sub set of architectures  $\mathcal{S}$  from  $\mathcal{A}$ ,  $\mathcal{S} \subset \mathcal{A}$ .  
2 Initialize performance predictors  $pre(a, t)$  for each task,  $\forall t \in \mathcal{T}, \forall a \in \mathcal{A}$ .  
3 Initialize  $ready\_flag\_t$  for each task,  $\forall t \in \mathcal{T}$ .  
4 **for**  $n = 1; n \leq k; n++$  **do**  
5     Sample architectures from  $\mathcal{S}$ .  
6     **for**  $t \in \mathcal{T}$  **do**  
7         Train predictor of task  $t$ .  
8         Calculate predicted architectures ranks of task  $t$ .  
9         Calculate Kendall tau between ground truth ranks and predicted ranks  
              for task  $t$ ,  $kd\_t$ .  
10         **if**  $kd\_t \geq thre\_t$  **then**  
11             |  $ready\_flag\_t = 1$ ;  
12         **end**  
13     **end**  
14     **if**  $ready\_flag\_t == 1, \forall t \in \mathcal{T}$  or  $n == k$  **then**  
15         Calculate objective function of ORP for all architectures in  $\mathcal{S}$   
              according to Eq. 13.  
16         Select best architecture  $a\_best$ .  
17         Return  $a\_best$ .  
18     **end**  
19 **end**

---

$\lambda$  is set to  $1/|\mathcal{T}|$  by default, and can be flexibly adjusted according to different tasks.

Then, we can formulate as follows:

$$\max \quad \lambda \sum_{t_1 \in \mathbf{t}} f_{t_1}(a) + (1 - \lambda) \sum_{t_2 \in \mathcal{T} \setminus \mathbf{t}} f_{t_2}(a) \quad (6)$$

$$\text{s.t. } a \in \mathcal{A}, \quad (7)$$

$$flops(a) \leq constraint\_flops \quad (8)$$

$$parameters(a) \leq constraint\_parameters \quad (9)$$

Nevertheless, as  $avg\_f(\mathbf{t}, a)$  has different metrics for different tasks which can not be added directly, we use rank instead of performance.

Similarly, let  $r_t(a)$  be the rank of performance of architecture  $a$  on task  $t$ ,  $\forall t \in \mathcal{T}, \forall a \in \mathcal{A}$ . Then, let  $r_{\mathbf{t}}(a)$  be the rank of performance of architecture  $a$  on task set  $\mathbf{t}$ ,  $\forall \mathbf{t} \subset \mathcal{T}, \forall a \in \mathcal{A}$ .

Similarly, we also care about the generalized rank on other tasks. Accordingly, let  $avg\_r(\mathbf{t}, a)$  be the comprehensive rank of architecture on all tasks,

where:

$$avg\_r(\mathbf{t}, a) = \lambda r_{\mathbf{t}}(a) + (1 - \lambda) r_{\mathcal{T} \setminus \mathbf{t}}(a) \quad (10)$$

$$= \lambda \sum_{t_1 \in \mathbf{t}} r_{t_1}(a) + (1 - \lambda) \sum_{t_2 \in \mathcal{T} \setminus \mathbf{t}} r_{t_2}(a) \quad (11)$$

$$, \forall \mathbf{t} \subset \mathcal{T}, \forall a \in \mathcal{A} \quad (12)$$

Finally, we formulate the optimal rank problem (ORP) as follows:

$$\min \lambda \sum_{t_1 \in \mathbf{t}} r_{t_1}(a) + (1 - \lambda) \sum_{t_2 \in \mathcal{T} \setminus \mathbf{t}} r_{t_2}(a) \quad (13)$$

$$\text{s.t. } a \in \mathcal{A}, \quad (14)$$

$$flops(a) \leq constraint\_flops \quad (15)$$

$$parameters(a) \leq constraint\_parameters \quad (16)$$

Then, a multi-task searching algorithm (MSA) is proposed to solve the ORP. Algorithm 1 shows the pseudo code of MSA. As the search space is huge, we first generate a sub set of architectures  $\mathcal{S}$  from  $\mathcal{A}$ ,  $\mathcal{S} \in \mathcal{A}$ . Then, we sample architectures from  $\mathcal{S}$  to train task specific performance predictors separately based on GP-NAS [27]. We utilize Kendall tau to measure the accuracy of the predictors. When the predictors are all well trained, we calculate the objective function of ORP for all architectures in  $\mathcal{S}$  according to Eq. 13 and select best architecture  $a\_best$ . In the experiment section, we will evaluate the performance of task specific predictors thoroughly.

## 4 Experiments

### 4.1 Settings

**Training dataset** MS1M-V3 (MS1M-RetinaFace) [15,9,8], Market1501-Train [44], MSMT17-Train [42], Veri-776-Train [28], VehicleID-Train [28], VeriWild-Train [32] and SOP-Train [36] are used as training dataset. The detailed information is shown in table 1.

**Table 1.** Training Dataset

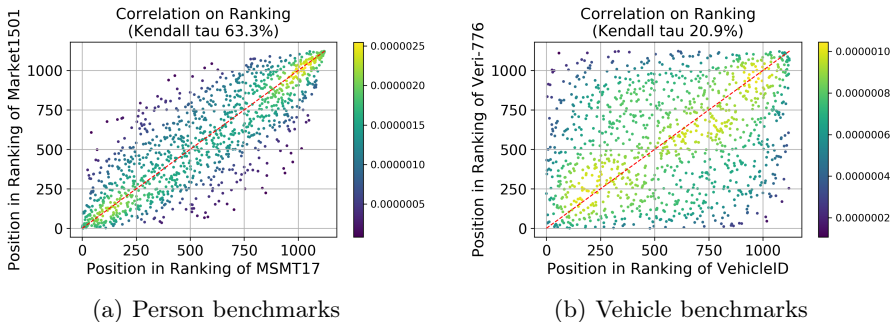
Tasks	Datasets	Img Number	ID Number
Face	MS1M-V3	5,179,510	93,431
Person	Market1501-Train	12,936	751
Person	MSMT17-Train	30,248	1,041
Vehicle	Veri-776-Train	37,778	576
Vehicle	VehicleID-Train	113,346	13,164
Vehicle	VeriWild-Train	277,797	30,671
Products	SOP-Train	59,551	11,318

**Table 2.** Test Dataset

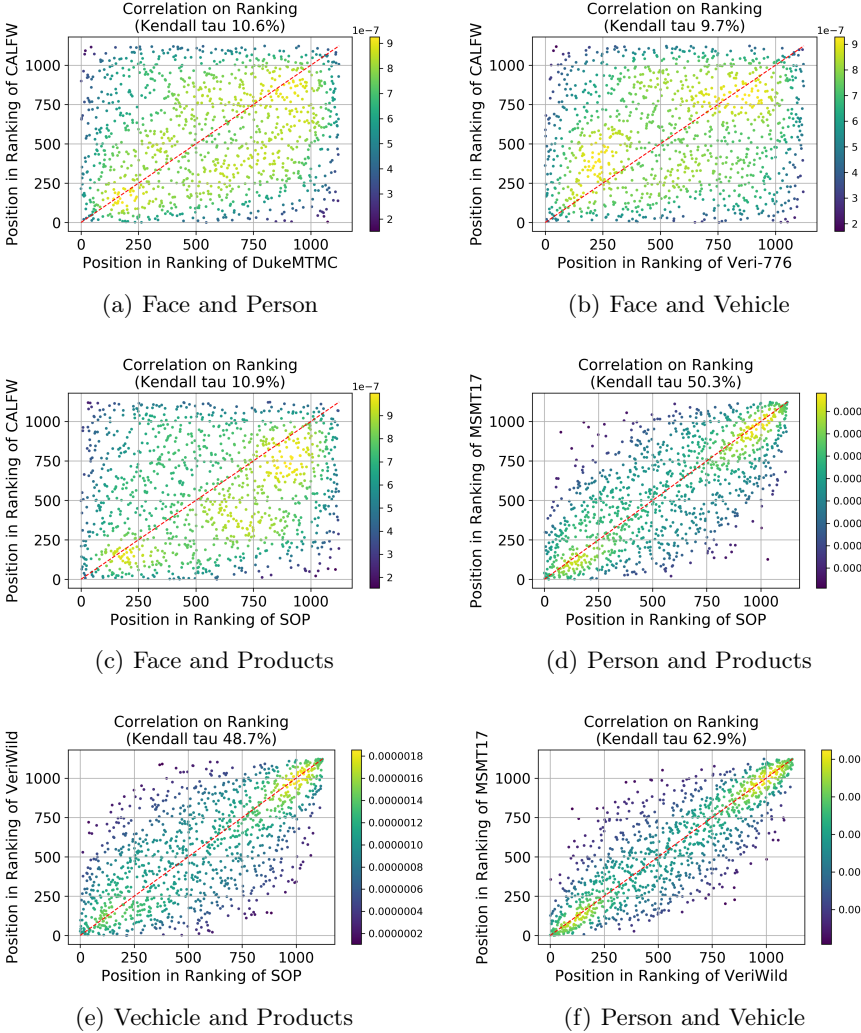
Tasks	Datasets	Img Number	ID Number
Face	LFW	12,000	-
Face	CPLFW	12,000	-
Face	CFP-FF	14,000	-
Face	CFP-FP	14,000	-
Face	CALFW	12,000	-
Face	AGEDB-30	12,000	-
Person	Market1501-Test	19,281	750
Person	MSMT17-Test	93,820	3,060
Vehicle	Veri-776-Test	13,257	200
Vehicle	VehicleID-Test	19,777	2,400
Vehicle	VeriWild-Test	138,517	10,000
Products	SOP-Test	60,502	11,316

**Table 3.** Training Configurations

Face/Person/Vehicle/Products	
Input Size	$256 \times 256$
Batch Size	1024/512/512/512
Augmentation	Flipping + Random Erasing + AutoAug
Model	ViT-base
Feature Dim	768
Loss	CosFace Loss/(CosFace Loss + Triplet Loss)*3
Optimizer	SGD
Init LR	0.2
LR scheduler	Warmup + Cosine LR
Iterations	100,000

**Fig. 2.** Ranking correlation within tasks.

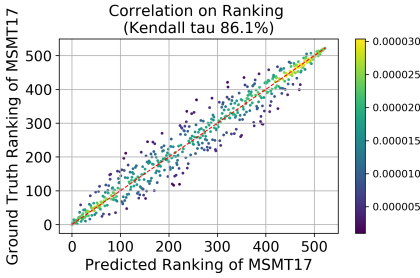
**Test dataset** Accordingly, we use LFW [21], CPLFW [45], CFP [38], CALFW [46], AGEDB-30 [35], Market1501-Test [44], MSMT17-Test [42], Veri-776-Test [28], VehicleID-Test [28], VeriWild-Test [32] and SOP-Test [36] as test dataset. The detailed information is shown in table 2.



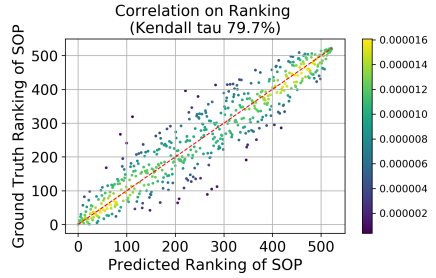
**Fig. 3.** Selected ranking correlation cross tasks

**Search space** The search space is set as follows:  $\mathcal{H} = \{10, 11, 12\}$ ,  $\mathcal{M} = \{3, 3.5, 4\}$ ,  $\mathcal{T} = \{t_1, t_2, t_3, t_4\}$ ,  $\mathcal{G} = \{g_{share}, g_1, g_2, g_3, g_4\}$  and the drop choices of the first 10 layers are all set to 1 and  $\lambda$  is set to  $1/4$ . The subset size  $|\mathcal{S}|$  is set to 10,000 and we sample 500 sub networks from  $\mathcal{S}$  for training where each sampled sub network is evaluated on all benchmarks. The sampled networks and  $\mathcal{S}$  are released as one of the first multi-task NAS benchmark and has supported the performance prediction track of the second lightweight NAS challenge of CVPR 2022 (<https://cvpr-nas.com/competition>).

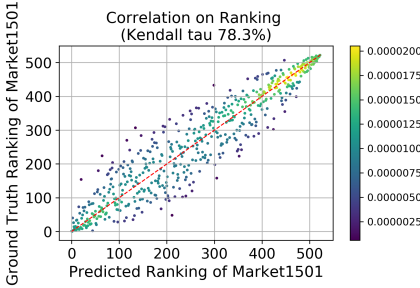
**Sampling strategy** We sample data from four tasks to form a batch, input the batch to the shared transformer backbone network, and finally separate four



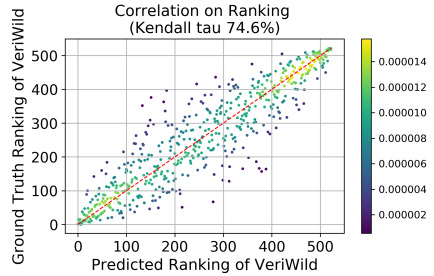
(a) Ranking correlation between ground truth and prediction on MSMT17.



(b) Ranking correlation between ground truth and prediction on SOP.



(c) Ranking correlation between ground truth and prediction on Market1501.



(d) Ranking correlation between ground truth and prediction on VeriWild.

**Fig. 4.** Performance of task specific predictors on selected benchmarks.

head networks, each of which is responsible for the output of a task. The four tasks separately calculate the loss and sum it up as the total loss.

**Training configurations** Because the input size and model structure used by different tasks are quite different. From the model optimization level, the batch size, learning rate and even the optimizer are all different. In order to facilitate subsequent multi-task training, we first unify the model structure and optimization method of each task. In particular, we use transformer as the backbone network. The unified configurations are shown in table 3.

Experiments with bigger backbones and with more dataset can be found in <https://github.com/PaddlePaddle/VIMER/tree/main/UF0>.

## 4.2 Correlation within and cross tasks

As different benchmark has different metric, we utilize Kendall tau to measure ranking correlation within and cross tasks. In this experiment, we have sampled 1000+ architectures from  $\mathcal{S}$  with different reduced parameters and evaluated the performance of each architecture on all benchmarks.

We first conduct experiment on person benchmarks and vehicle benchmarks. As shown in figure 2, person benchmarks are highly correlated while, vehicle benchmark are slightly correlated.



**Table 4.** The impact of FFN paths.

Datasets	All shared	UFO-Face		UFO-Person		UFO-SOP	
		with	w/o	with	w/o	with	w/o
CALFW	95.86	<b>96.00</b>	95.95	<b>96.02</b>	95.92	95.90	95.92
CPLFW	93.53	93.62	93.65	<b>93.62</b>	93.37	93.30	93.37
Market1501	88.19	<b>89.33</b>	87.57	<b>89.44</b>	87.62	<b>89.44</b>	87.62
MSMT17	60.08	<b>64.24</b>	61.13	<b>64.84</b>	61.38	<b>64.66</b>	61.38
Veri-776	86.11	<b>87.90</b>	86.48	<b>88.03</b>	86.50	<b>87.95</b>	86.50
VehicleID	86.27	<b>85.43</b>	85.40	<b>87.00</b>	85.46	<b>86.41</b>	85.46
VeriWild	68.43	<b>69.60</b>	67.85	<b>69.72</b>	67.92	<b>69.69</b>	67.92
SOP	86.64	<b>86.09</b>	83.47	<b>86.19</b>	83.50	<b>86.24</b>	83.53
Flops reduction relative to ViT-base	0%	29.52%	27.38%	26.94%	27.38%	27.15%	27.38%
Param reduction relative to ViT-base	0%	29.39%	27.25%	26.88%	27.31%	26.88%	27.10%

Then, we conduct experiments on benchmarks between different tasks. As shown in figure 3, the benchmark of face is slightly correlated to all other tasks. While, certain benchmarks of person, vehicle and products are highly correlated.

### 4.3 The performances of task specific predictors

Figure 4 illustrates the performance of task specific predictors. In the experiment, we sample 500 sub networks for training and 500+ sub networks (no overlap) for testing where each sampled sub network is evaluated on all tasks. Then, we measure the correlation between the predicted rankings and ground truth rankings. As shown in the figure, the predictors are with very good accuracy.

### 4.4 The impact of FFN paths

As shown in table 4, FFN paths is of significant importance in multi-task learning, compared with all shared approach. The supernet with FFN paths can significantly improve the performances. One possible explanation is that, FFN paths relieve the competition of shared attention weights.

### 4.5 Compared with SOTA results

In this subsection, we compare UFO with previous SOTA results <sup>1</sup> on 10 benchmarks, that are SOTA results on CALFW [2], CPLFW [24], CFP-FF [7], Market1501 [19], MSMT17 [18], Veri-776 [22], VehicleID [17], VeriWild [17] and SOP [26]. As shown in table 5, UFO reaches SOTA result on CFP-FF and creates 4 new SOTA results on CPLFW, Veri-776, VehicleID and SOP. We can averagely reduce 51.08% flops and 42.34% parameters (relative to supernet). Besides,

<sup>1</sup> without rerank strategy and external data from <https://paperswithcode.com/>

**Table 5.** Compared with SOTA results

	SOTA	UFO for CPLFW	UFO for CFP-FF	UFO for Veri776	UFO for VehicleID	UFO for SOP
CALFW	96.20	95.78	95.98	95.95	95.83	95.95
CPLFW	93.37	93.65	93.40	93.60	93.47	93.63
CFP-FF	99.89	99.87	99.89	99.83	99.86	99.86
Market1501	91.50	89.45	89.52	89.46	89.51	89.50
MSMT17	69.40	64.84	64.86	64.87	65.03	65.06
Veri-776	87.10	88.03	88.01	88.18	88.09	88.06
VehicleID	80.50	85.26	86.32	86.18	87.04	86.44
VeriWild	77.30	69.74	69.71	69.73	69.82	69.74
SOP	85.90	86.19	86.22	86.21	86.25	86.39
Flops reduction relative to ViT-base	-	18.90%	25.00%	11.50%	20.00%	24.00%
Param reduction relative to ViT-base	-	17.60%	23.90%	10.30%	18.90%	22.80%
Flops reduction relative to supernet	-	50.84%	54.54%	46.36%	51.51%	53.93%
Param reduction relative to supernet	-	41.97%	46.40%	36.83%	42.88%	45.63%

we also compare UFO against two recent state-of-the-art multi-task methods, *i.e.*, Switch Transformers [12] and DSelect-k [16] (based on their publicly available code). Our UFO surpasses Switch Transformers/DSelect-k by 1.28/1.05% (CALFW), 1.61/1.30% (CPLFW), 0.29/0.25% (CFP-FF), 1.43/0.98% (Market-1501), 1.79/2.18% (MSMT17), 5.39/6.27% (Veri-776), 6.28/6.47% (VehicleID), 12.32/12.57% (VeriWild), 0.03/0.62% (SOP), respectively.

## 5 Conclusions

This paper proposes a novel train-and-deploy paradigm named Unified Feature Optimization (UFO) to benefit down-stream tasks with large-scale pretraining. UFO maintains the benefit of large-scale pretraining and provides high convenience for flexible deployment: it transfers the multi-task trained supernet to a dedicated model for any already-seen sub-tasks without no adaptation cost and reduces the model size during adaptation. On deep representation learning tasks, we explore an early prototype of UFO based on Vision Transformer (ViT) and Neural Architecture Search (NAS) techniques. Specifically, UFO integrates multiple trimming strategies to enhance the trimming flexibility for ViT and employs a novel performance-ranking based method for NAS. Experimental results show that the sub-model trimmed from supernet surpasses its single-task-trained counterpart and has smaller model size. That being said, we also note that the accuracy improvement and the size reduction is still marginal and call for more efforts from the research community to explore UFO. Besides, UFO also supported the release of **17 billion parameters computer vision (CV) foundation model** which is the largest CV model in the industry.

## A Multi-task NAS benchmark

The sampled sub networks in UFO are released as one of the first multi-task NAS benchmark and has supported the performance prediction track of the second lightweight NAS challenge of CVPR 2022 (<https://cvpr-nas.com/competition>). The Multi-task NAS benchmark is released in **Baidu AI Studio** which is a one-stop developer platform based on Baidu’s deep learning platform PaddlePaddle. Baidu AI Studio provides free online courses, free computing power support and non-stop competitions to encourage the development of deep learning.

## B More Experiments

UFO also utilized bigger backbone and more dataset and released the **VIMER-UFO**, a task-MoE based 17 billion parameters computer vision foundation model, which supports extraction of lightweight models by sparse activation and achieves SOTA on 28 datasets across a battery of visual recognition tasks.

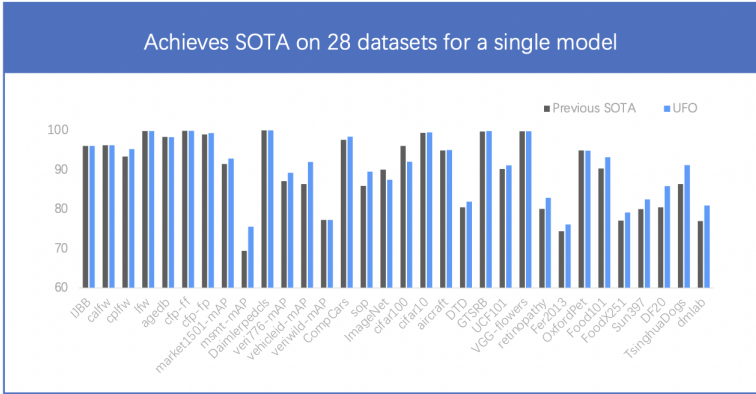


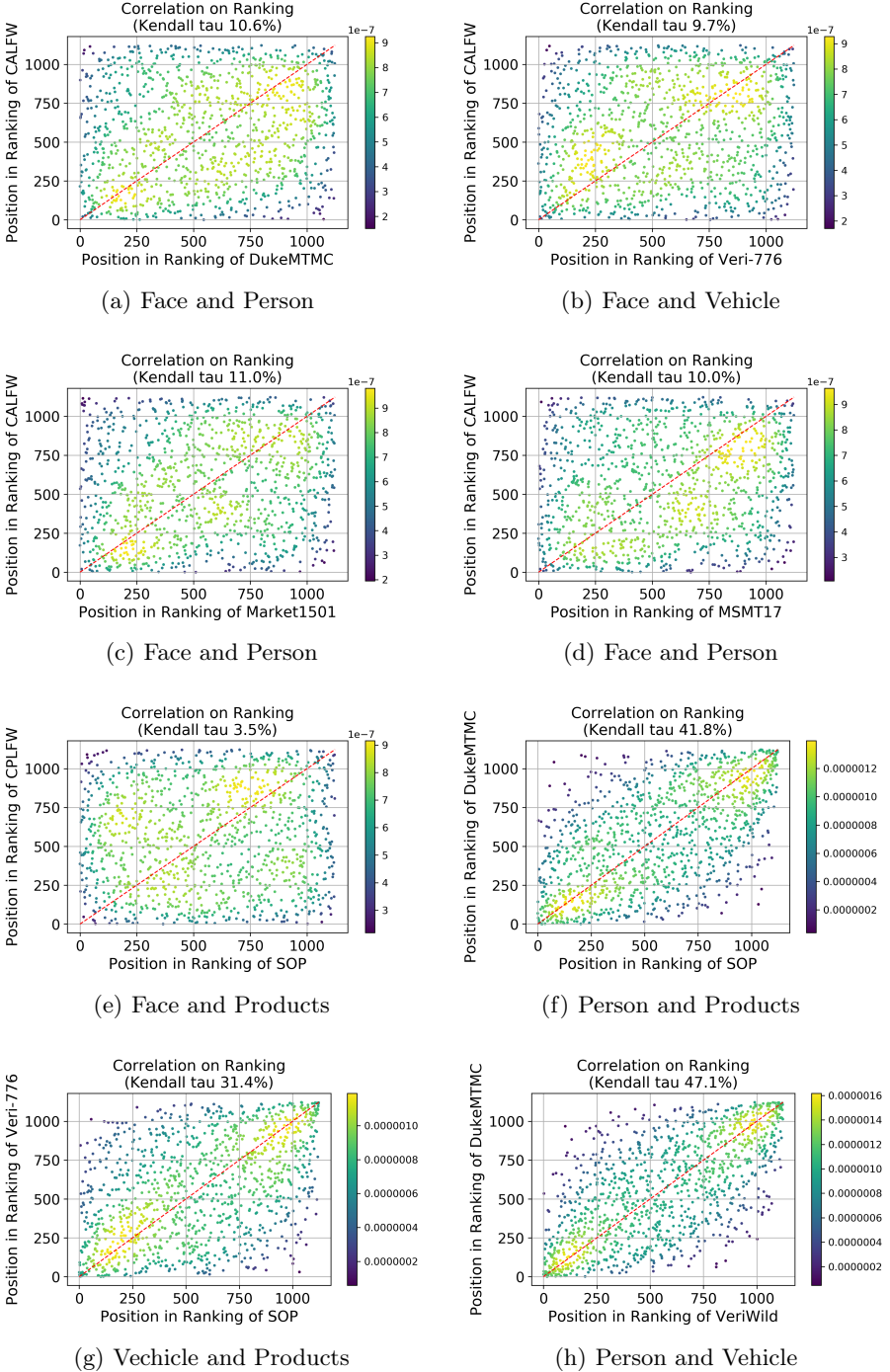
Fig. 5. VIMER-UFO achieves SOTA on 28 datasets for a single model.

## C All cross tasks correlations

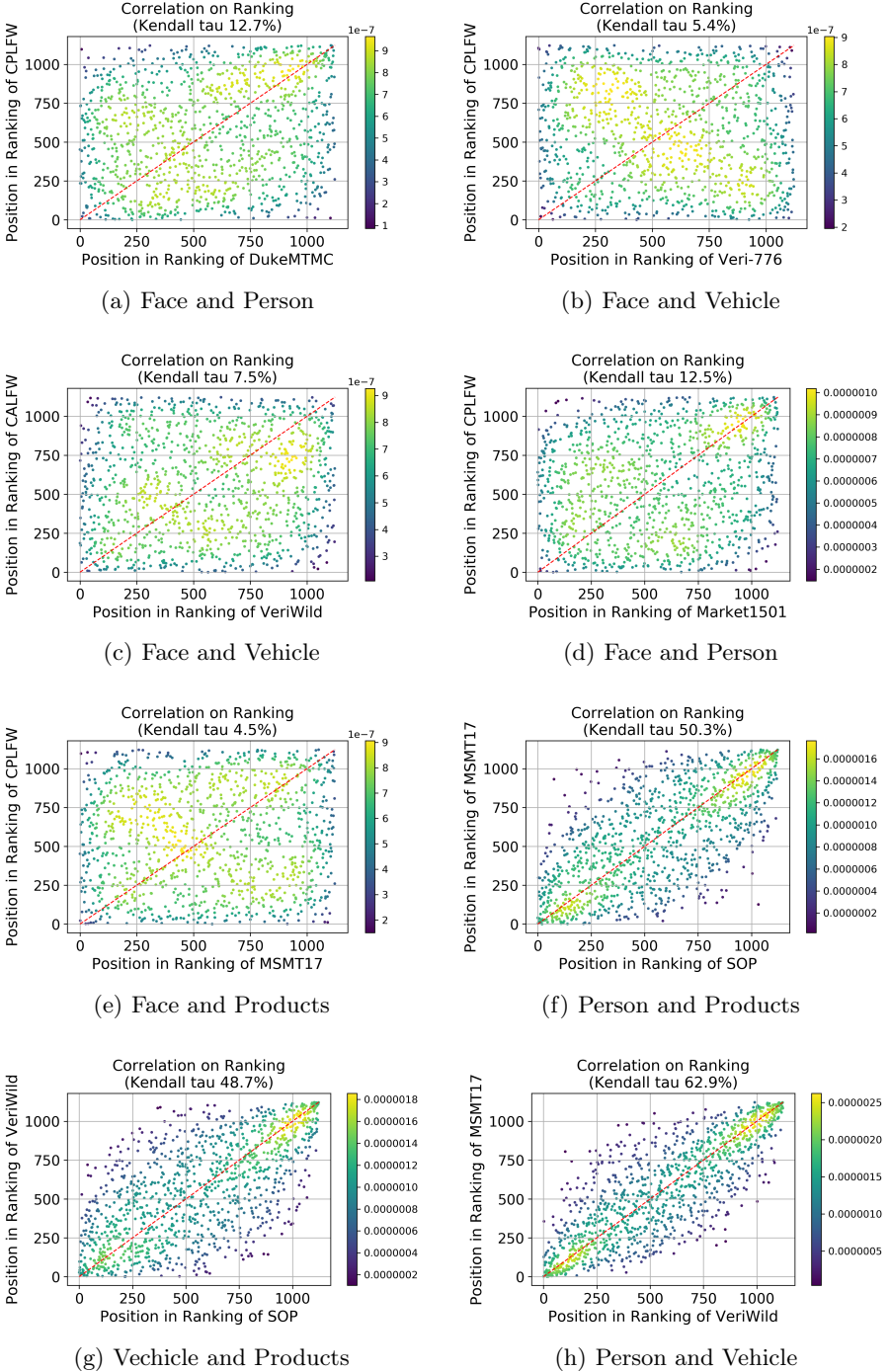
In this section, we will show all cross tasks correlations. As shown in figure 6, 7 and 8, the benchmark of face is slightly correlated to all other tasks. While, certain benchmarks of person, vehicle and products are highly correlated.

## D The performances of task specific predictors

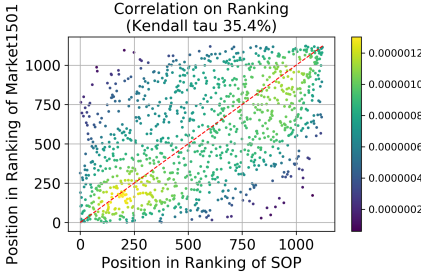
Figure 9, illustrate the performances of the task specific predictors. We measure the correlation between the predicted rankings and ground truth rankings of selected benchmarks. As shown in the figure, the predictors all have very good accuracy except for CPLFW.



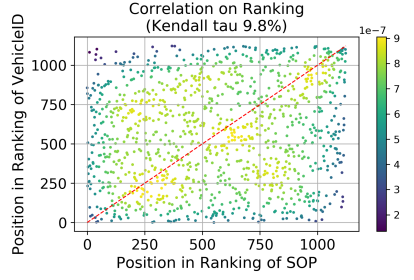
**Fig. 6.** All cross tasks correlations part one.



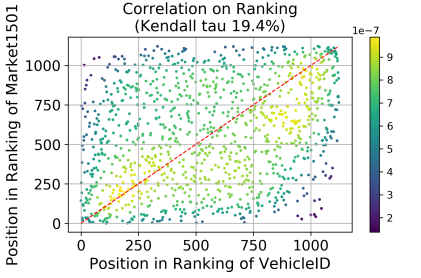
**Fig. 7.** All cross tasks correlations part two.



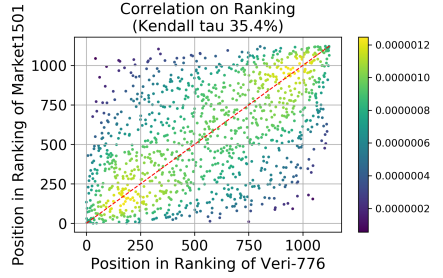
(a) Person and Products



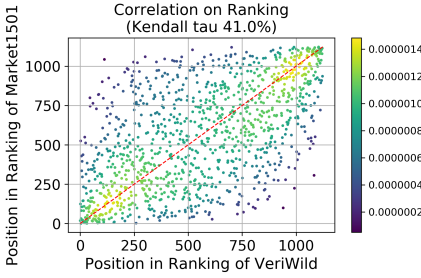
(b) Vehicle and Products



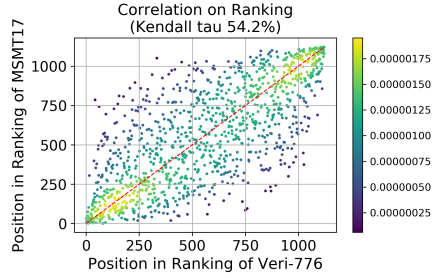
(c) Person and Vehicle



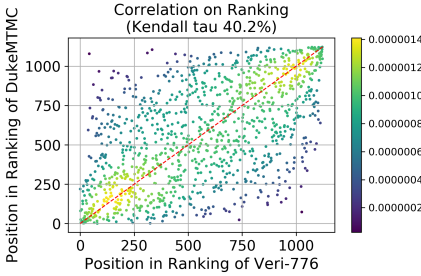
(d) Person and Vehicle



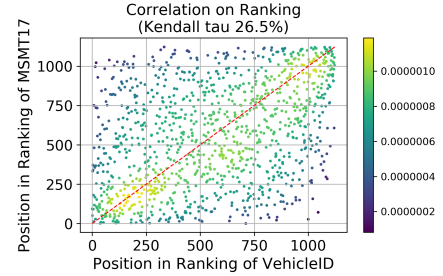
(e) Person and Vehicle



(f) Person and Vehicle

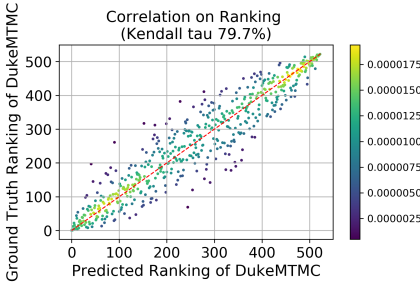


(g) Person and Vehicle

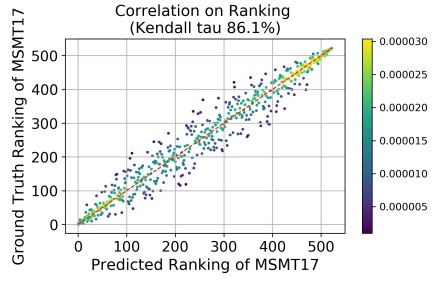


(h) Person and Vehicle

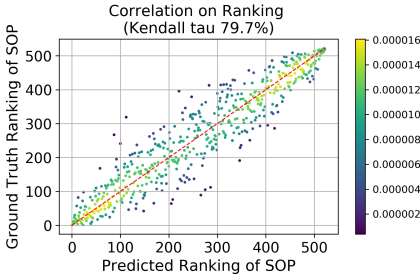
**Fig. 8.** All cross tasks correlations part three.



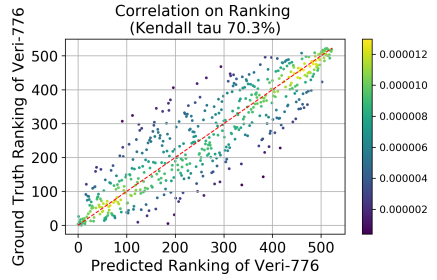
(a) Predictions on MTMC.



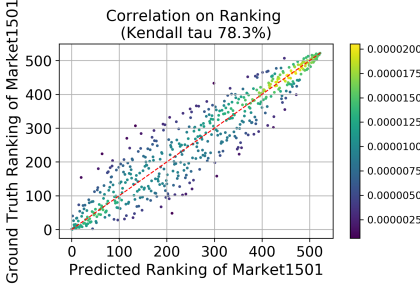
(b) Predictions on MSMT17.



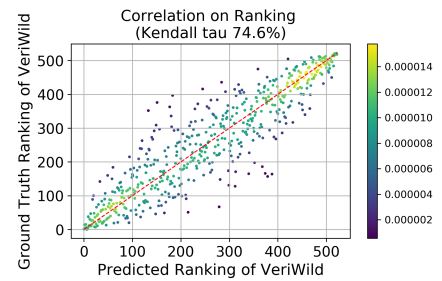
(c) Predictions on SOP.



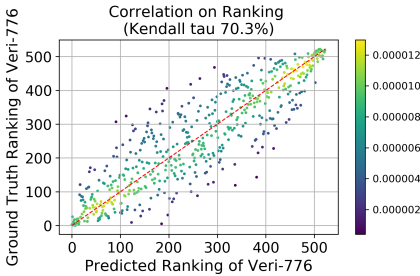
(d) Predictions on Veri-776.



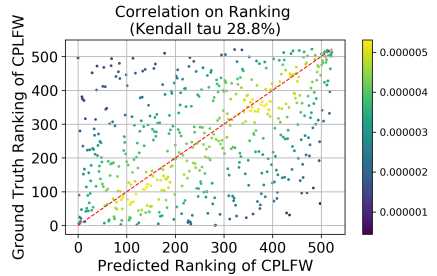
(e) Predictions on Market1501.



(f) Predictions on VeriWild.



(g) Predictions on VehicleID.



(h) Predictions on CPLFW.

**Fig. 9.** Performance of task specific predictors on more benchmarks.



## References

1. Abnar, S., Dehghani, M., Neyshabur, B., Sedghi, H.: Exploring the limits of large scale pre-training. arXiv preprint arXiv:2110.02095 (2021)
2. An, X., Zhu, X., Gao, Y., Xiao, Y., Zhao, Y., Feng, Z., Wu, L., Qin, B., Zhang, M., Zhang, D., et al.: Partial fc: Training 10 million identities on a single machine. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1445–1449 (2021)
3. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)
4. Cai, H., Gan, C., Wang, T., Zhang, Z., Han, S.: Once-for-all: Train one network and specialize it for efficient deployment. arXiv preprint arXiv:1908.09791 (2019)
5. Chen, M., Peng, H., Fu, J., Ling, H.: Autoformer: Searching transformers for visual recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12270–12280 (2021)
6. Chen, Z., Badrinarayanan, V., Lee, C.Y., Rabinovich, A.: Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In: International Conference on Machine Learning. pp. 794–803. PMLR (2018)
7. Chrysos, G.G., Moschoglou, S., Bouritsas, G., Deng, J., Panagakis, Y., Zafeiriou, S.P.: Deep polynomial neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
8. Deng, J., Guo, J., Ververas, E., Kotsia, I., Zafeiriou, S.: Retinaface: Single-shot multi-level face localisation in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5203–5212 (2020)
9. Deng, J., Guo, J., Zhang, D., Deng, Y., Lu, X., Shi, S.: Lightweight face recognition challenge. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0 (2019)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
11. Duong, L., Cohn, T., Bird, S., Cook, P.: Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In: Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (volume 2: short papers). pp. 845–850 (2015)
12. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. arXiv preprint arXiv:2101.03961 (2021)
13. Gao, Y., Bai, H., Jie, Z., Ma, J., Jia, K., Liu, W.: Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. pp. 11543–11552 (2020)
14. Guo, M., Haque, A., Huang, D.A., Yeung, S., Fei-Fei, L.: Dynamic task prioritization for multitask learning. In: Proceedings of the European conference on computer vision (ECCV). pp. 270–287 (2018)
15. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In: European conference on computer vision. pp. 87–102. Springer (2016)



16. Hazimeh, H., Zhao, Z., Chowdhery, A., Sathiamoorthy, M., Chen, Y., Mazumder, R., Hong, L., Chi, E.: Dselect-k: Differentiable selection in the mixture of experts with applications to multi-task learning. *Advances in Neural Information Processing Systems* **34**, 29335–29347 (2021)
17. He, L., Liao, X., Liu, W., Liu, X., Cheng, P., Mei, T.: Fastreid: A pytorch toolbox for general instance re-identification. *arXiv preprint arXiv:2006.02631* (2020)
18. He, S., Luo, H., Wang, P., Wang, F., Li, H., Jiang, W.: Transreid: Transformer-based object re-identification. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 15013–15022 (2021)
19. Herzog, F., Ji, X., Teepe, T., Hörmann, S., Gilg, J., Rigoll, G.: Lightweight multi-branch network for person re-identification. In: *2021 IEEE International Conference on Image Processing (ICIP)*. pp. 1129–1133. IEEE (2021)
20. Hou, L., Huang, Z., Shang, L., Jiang, X., Chen, X., Liu, Q.: Dynabert: Dynamic bert with adaptive width and depth. *Advances in Neural Information Processing Systems* **33**, 9782–9793 (2020)
21. Huang, G.B., Mattar, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In: *Workshop on faces in 'Real-Life' Images: detection, alignment, and recognition* (2008)
22. Huynh, S.V.: A strong baseline for vehicle re-identification. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 4147–4154 (2021)
23. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 7482–7491 (2018)
24. Kim, Y., Park, W., Roh, M.C., Shin, J.: Groupface: Learning latent groups and constructing group-based representations for face recognition. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 5621–5630 (2020)
25. Kudugunta, S., Huang, Y., Bapna, A., Krikun, M., Lepikhin, D., Luong, M.T., Firat, O.: Beyond distillation: Task-level mixture-of-experts for efficient inference. *arXiv preprint arXiv:2110.03742* (2021)
26. Lee, J., Won, T., Lee, T.K., Lee, H., Gu, G., Hong, K.: Compounding the performance improvements of assembled techniques in a convolutional neural network. *arXiv preprint arXiv:2001.06268* (2020)
27. Li, Z., Xi, T., Deng, J., Zhang, G., Wen, S., He, R.: Gp-nas: Gaussian process based neural architecture search. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 11933–11942 (2020)
28. Liu, H., Tian, Y., Yang, Y., Pang, L., Huang, T.: Deep relative distance learning: Tell the difference between similar vehicles. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 2167–2175 (2016)
29. Liu, P., Qiu, X., Huang, X.: Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101* (2016)
30. Liu, X., He, P., Chen, W., Gao, J.: Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504* (2019)
31. Long, M., Wang, J.: Learning multiple tasks with deep relationship networks. *arXiv preprint arXiv:1506.02117* **2**(1) (2015)
32. Lou, Y., Bai, Y., Liu, J., Wang, S., Duan, L.: Veri-wild: A large dataset and a new method for vehicle re-identification in the wild. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 3235–3243 (2019)

33. Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., Feris, R.: Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5334–5343 (2017)
34. Misra, I., Shrivastava, A., Gupta, A., Hebert, M.: Cross-stitch networks for multi-task learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3994–4003 (2016)
35. Moschoglou, S., Papaioannou, A., Sagonas, C., Deng, J., Kotsia, I., Zafeiriou, S.: Agedb: the first manually collected, in-the-wild age database. In: proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 51–59 (2017)
36. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4004–4012 (2016)
37. Sener, O., Koltun, V.: Multi-task learning as multi-objective optimization. *Advances in neural information processing systems* **31** (2018)
38. Sengupta, S., Chen, J.C., Castillo, C., Patel, V.M., Chellappa, R., Jacobs, D.W.: Frontal to profile face verification in the wild. In: 2016 IEEE winter conference on applications of computer vision (WACV). pp. 1–9. IEEE (2016)
39. Subramanian, S., Trischler, A., Bengio, Y., Pal, C.J.: Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079* (2018)
40. Suteu, M., Guo, Y.: Regularizing deep multi-task networks using orthogonal gradients. *arXiv preprint arXiv:1912.06844* (2019)
41. Wang, H., Wu, Z., Liu, Z., Cai, H., Zhu, L., Gan, C., Han, S.: Hat: Hardware-aware transformers for efficient natural language processing. *arXiv preprint arXiv:2005.14187* (2020)
42. Wei, L., Zhang, S., Gao, W., Tian, Q.: Person transfer gan to bridge domain gap for person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 79–88 (2018)
43. Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., Finn, C.: Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems* **33**, 5824–5836 (2020)
44. Zheng, L., Shen, L., Tian, L., Wang, S., Bu, J., Tian, Q.: Person re-identification meets image search. *arXiv preprint arXiv:1502.02171* (2015)
45. Zheng, T., Deng, W.: Cross-pose lfw: A database for studying cross-pose face recognition in unconstrained environments. *Beijing University of Posts and Telecommunications, Tech. Rep* **5**, 7 (2018)
46. Zheng, T., Deng, W., Hu, J.: Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments. *arXiv preprint arXiv:1708.08197* (2017)