Supplemental Material: Hub-Pathway: Transfer Learning from A Hub of Pre-trained Models

Yang Shu, Zhangjie Cao, Ziyang Zhang§, Jianmin Wang, Mingsheng Long✉
School of Software, BNRist, Tsinghua University, China
§Advanced Computing and Storage Lab, Huawei Technologies Co. Ltd
`{shuyang5656,caozhangjie14}@gmail.com, {jimwang,mingsheng}@tsinghua.edu.cn`

A Experiment Results

A.1 Complexity Analysis

Model hub transfer learning usually includes many pre-trained models and thus the computational and the memory cost should be controlled to an acceptable extent. We compare the time complexity and the parameter size of our framework, a single model and the ensemble of all models in Table 1. The experiments are conducted on the CIFAR task with the batch-size of 12. Here we consider two variants of Ensemble: (1) fine-tuning the ensemble of 5 pre-trained models jointly and testing with their ensemble (Ensemble-J), (2) fine-tuning 5 pre-trained models independently and testing with their ensemble (Ensemble-I). Hub-Pathway outperforms the two variants of Ensemble, which shows the effectiveness of the proposed method. It only has a little more parameters than Ensemble methods, but can be used much more efficiently than them. The costs of Ensemble-J all grow with the number of the models, making it inefficient for the problem. Ensemble-I saves the training memory by fine-tuning one model each time, but the computational costs during training and the costs during inference cannot be saved. Hub-Pathway also introduces additional costs than the single model baseline, but the additional costs can be controlled by pathway activation and thus do not scale up with the size of the hub. Even compared with ImageNet, which is a single model, Hub-Pathway only has about twice of the cost. The high efficiency attributes to our design of keeping top k pathways, which only activates a few models for forward-propagation and back-propagation. It also outperforms the two variants of Ensemble on most of the complexity metrics reported above. In all, Hub-Pathway achieves a better balance between performance and efficiency for model hub transfer learning.

Table 1: Performance and complexity of Hub-Pathway, Ensemble and the ImageNet models.

Model	Acc (%) ↑	Params (M) ↓	FLOPs (G) ↓	Memory (M) ↓	Speed (samples/s) ↑
ImageNet	83.41	23.71	4.11	1905	484.92
Ensemble-J	83.87	118.55	20.55	6397	98.64
Ensemble-I	84.50	118.55	20.55	6397	98.64
Hub-Pathway	85.63	128.43	9.11	3537	240.48

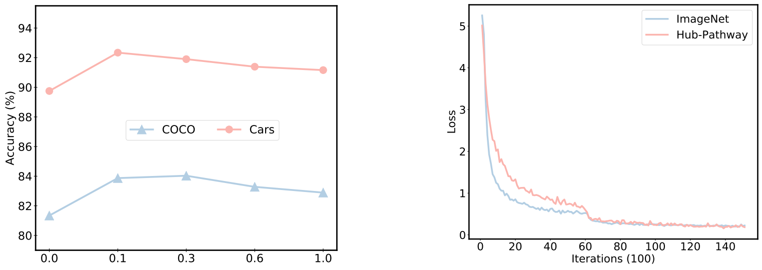
Table 2: Inference time per image (s)

Method	Generator	Pre-trained Models
Single	-	0.012
Ensemble	-	0.056
Hub-Pathway	0.003	0.023

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Table 3: Memory cost in Megabytes

Method	Load Models	Forward Generator	Forward Models
Single	897	-	+1008
Ensemble	1179	-	+5218
Hub-Pathway	1203	+274	+2060



(a) Influence of λ

(b) Training Curve

Figure 1: Some more analysis on Hub-Pathway.

We further report the time cost in the CIFAR task of the image classification dataset in Table 2. We report the time of forwarding one piece of data through the model and separately report the time for the pathway generator and the pre-trained models. We observe that the time for the pathway generator is much smaller than the time for the pre-trained models, showing the efficiency of the pathway generator. Since we only activate top-k models, the time cost scales linearly with k and does not increase with the increasing size of the model hub.

We then test the detailed memory cost in three steps: (1) load the pre-trained models in the memory, (2) forward the data through the pathway generator, (3) forward the data through pre-trained models. We conduct the experiment on the classification task in CIFAR with a batch-size of 12 and report the results in Table 3. Comparing results in Column 1, the additional memory cost of loading multiple pre-trained models is about 300 M, which is relatively smaller than those brought by forwarding data through models (Column 3). In Column 3, the memory cost of Ensemble scales up with the model size, to 5 times of the single model. The cost of Hub-Pathway is controlled by TopK selection with the pathway activation mechanism. In Column 1, the additional cost of storing the generator is also negligible. Compared Column 2 with Column 3, the memory cost of forwarding through the generator is also relatively small compared with forwarding through the pre-trained models.

A.2 Influence of the Trade-off

Hyperparameter λ controls the trade-off between L_{task} and L_{explore}. We tune the hyper-parameter λ on one task in each dataset and fix the hyper-parameter for other tasks to avoid over-tuning. We conduct an experiment on the influence of different λ on the performance and report the results in the Figure 1(a).We conduct the experiments on the COCO and Cars tasks for image classification. We observe that the performance is table around $\lambda = 0.3$. But without λ ($\lambda = 0$), the performance drops much, which indicates that loss L_{explore} is needed in our method to enhance exploration of the hub.

A.3 Convergence Speed

We plot the training curve for the Cars task in the image classification dataset in Figure 1(b). We compare the convergence speed of Hub-Pathway and a single ImageNet model. We observe that in the first stage Hub-Pathway converges a little slower because of pathway finding, but then it converges with a similar speed to the single ImageNet model. As stated in the implementation details, Hub-Pathway and competitors are trained for the same iterations.

B Experiment Details

In this section, we supplement more experiment details on the computing infrastructure.

B.1 Computing Infrastructure

We use PyTorch 1.9.0, torchvison 0.10.0, and CUDA 10.0 libraries for the image classification tasks. We use PyTorch 1.0.0, torchvison 0.2.1, and CUDA 9.0 libraries for the facial landmark detection tasks. We implement all the reinforcement learning experiments based on PyTorch 1.5.0, torchvison 0.6.0, and CUDA 10.0 libraries. We use a machine with 32 CPUs, 256 GB memory, and GPUs of NVIDIA TITAN X.

C Broad Impact

Real-World Applications. Real-world deep learning applications face the problem of insufficient data, while transfer learning could use more diverse resource of data to compensate for the lack of data. Also, as discussed in our introduction, model hub transfer learning is an efficient way to obtain abundant knowledge without much cost. Our proposed Hub-Pathway method focuses on the problem of transfer learning from a hub of pre-trained models and shows consistent improvements under various situations. Thus, with the emergence of pre-trained models available, users from different fields can collect their interested models and perform our method to build the target model for their applications efficiently. Hub-Pathway may also have the potential to be plugged into some existing model hub providers to provide an effective end-to-end transfer learning solution.

Robustness and Limitations. Based on the extensive experiments, we find that Hub-Pathway consistently outperforms the models in the hub, which shows its robustness in different forms of data or tasks. Since the framework does not filter the models in the hub in advance, it may not automatically overcome some inherent flaws of the models, such as the mismatch between the inputs and the models and the copyright issues of the models. This can be solved in practice because users should select models somehow related to the downstream task into the model hub and do necessary data pre-processing. They should also ensure that the models used in the hub are collected with authorization. Although Hub-Pathway controls the additional costs to an acceptable extent in most situations, it still costs more than fine-tuning from one model, and there may be some extreme cases challenging its usage. In cases with extremely low resources, the memory budget may not afford to store or forward even one more model. In cases where the pre-trained models are extremely large, the costs of storing multiple models' parameters may be even higher than the output tensors. In these situations, loading all models into the memory like existing multi-model methods, including Hub-Pathway, may be inefficient. A possible solution would be swapping activated models in the memory to sacrifice time for space. This motivates us to develop a more effective solution that may achieve performance gains with almost no efficiency losses in our future work.

Our work only focuses on the scientific problem, so there is no potential ethical risk.