ModelKeeper: Accelerating DNN Training via Automated Training Warmup

Fan Lai, Yinwei Dai, Harsha V. Madhyastha, Mosharaf Chowdhury University of Michigan

nsdi'23

Introduction

- A DNN model is a graph of tensors.

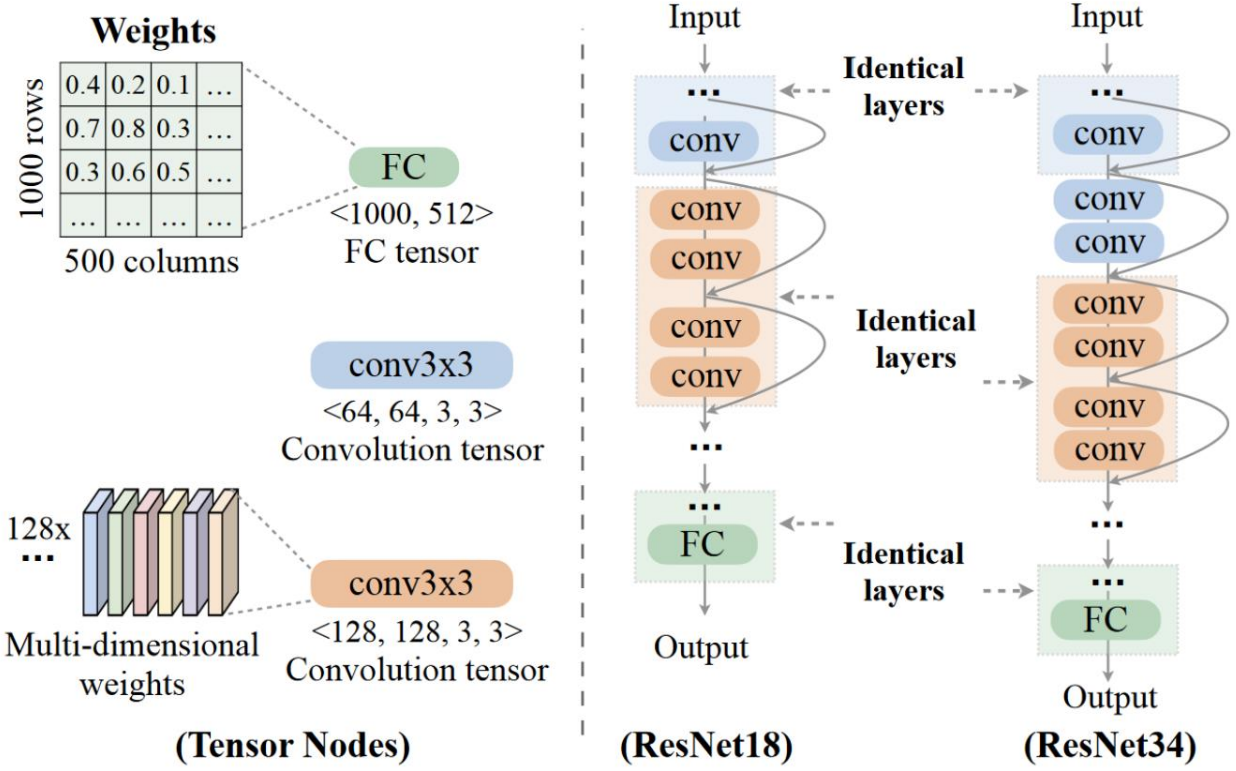


Figure 1: ***A DNN model is essentially a graph of tensors. Model outputs are determined by tensor weights and their control flow.***

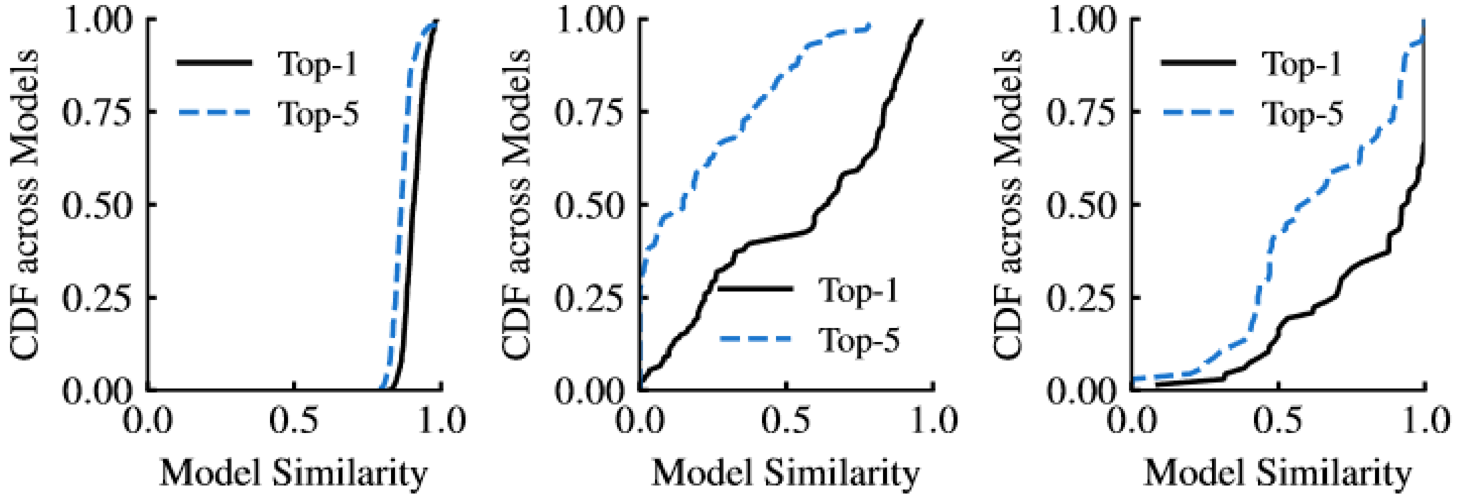
Introduction

- **The natural similarity between models.**
  - **the same NAS process**
  - **the same ML task in different hardware**

- **A key insight: A well-trained model’s weights can warm up the training of a new model.**

Motivation

- **Opportunities for Repurposing Models**

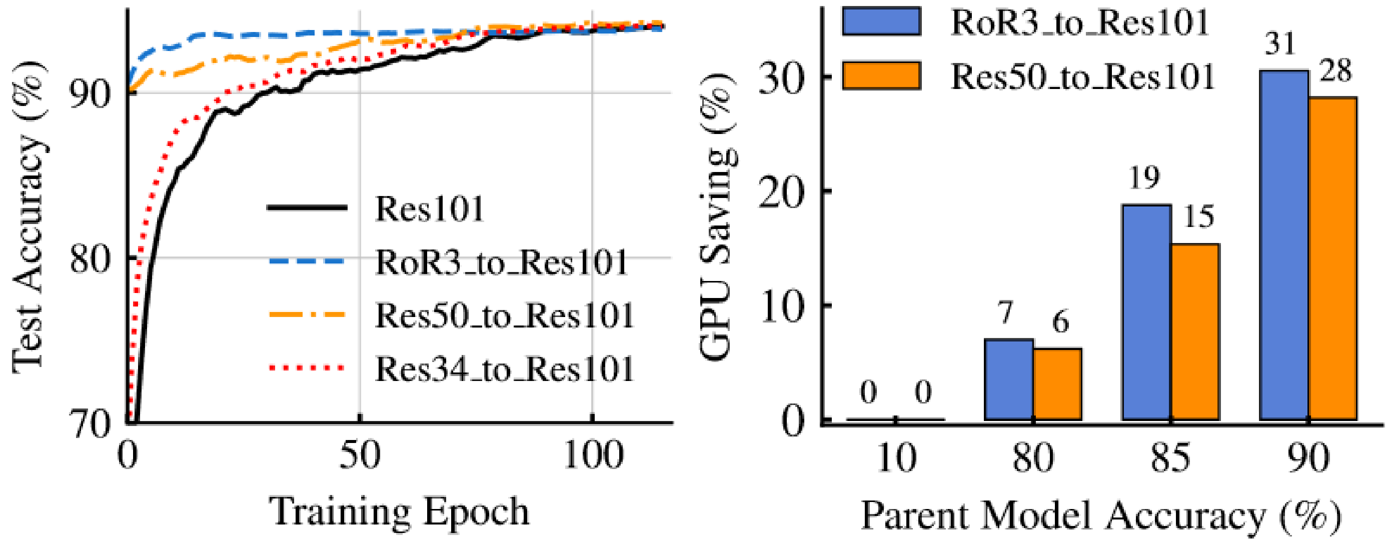


(a) NASBench (b) Imgclsmob (c) HuggingFace

Figure 2: ***Pervasive model similarity in today's model zoos. We measure the top-1 and top-5 architectural similarities of each model to other models, and report the distribution across models. 1 indicates identical model architectures.***

Motivation

- **Opportunities for Repurposing Models**

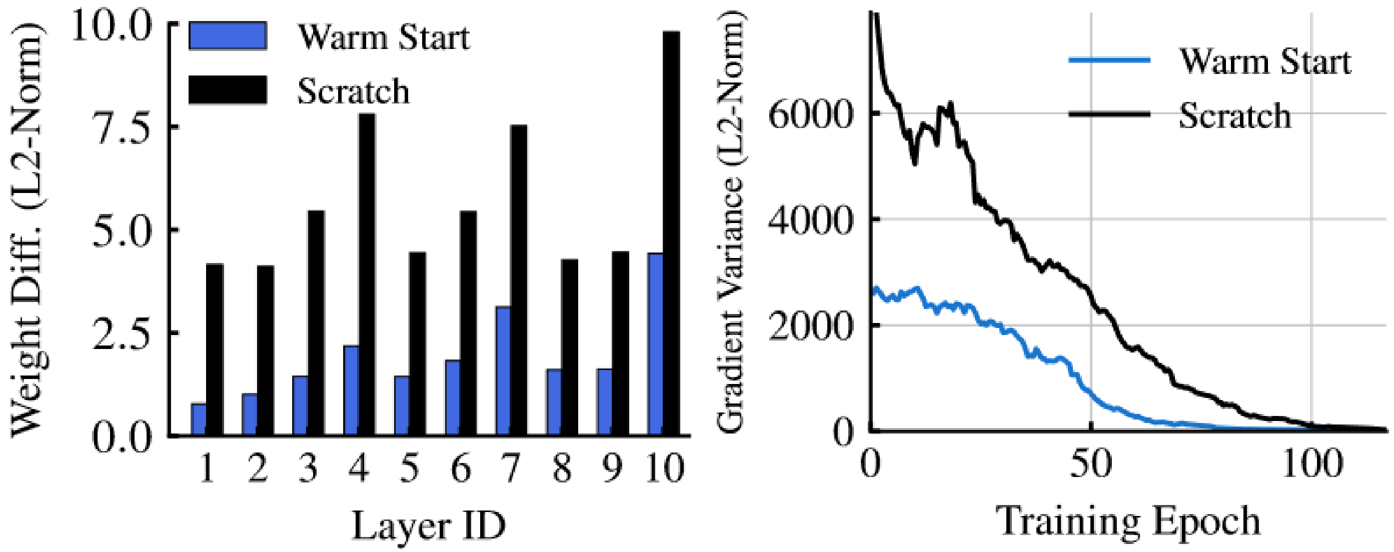


(a) Warm start accelerates training. (b) Parent model accuracy matters.

Figure 3: ***Transferring model weights from well-trained models with similar architectures can accelerate new model training.***

Motivation

- **Opportunities for Repurposing Models**



(a) Smaller divergence to the optimal. (b) Smaller gradient variance.

Figure 4: ***Warm start provides better initial weights search space. We use RoR3 to warm start ResNet101.***

System Overview

- **ModelKeeper Coordinator**
  - ***Model Matcher***
  - ***Model Mapper***
  - ***Zoo Manager***

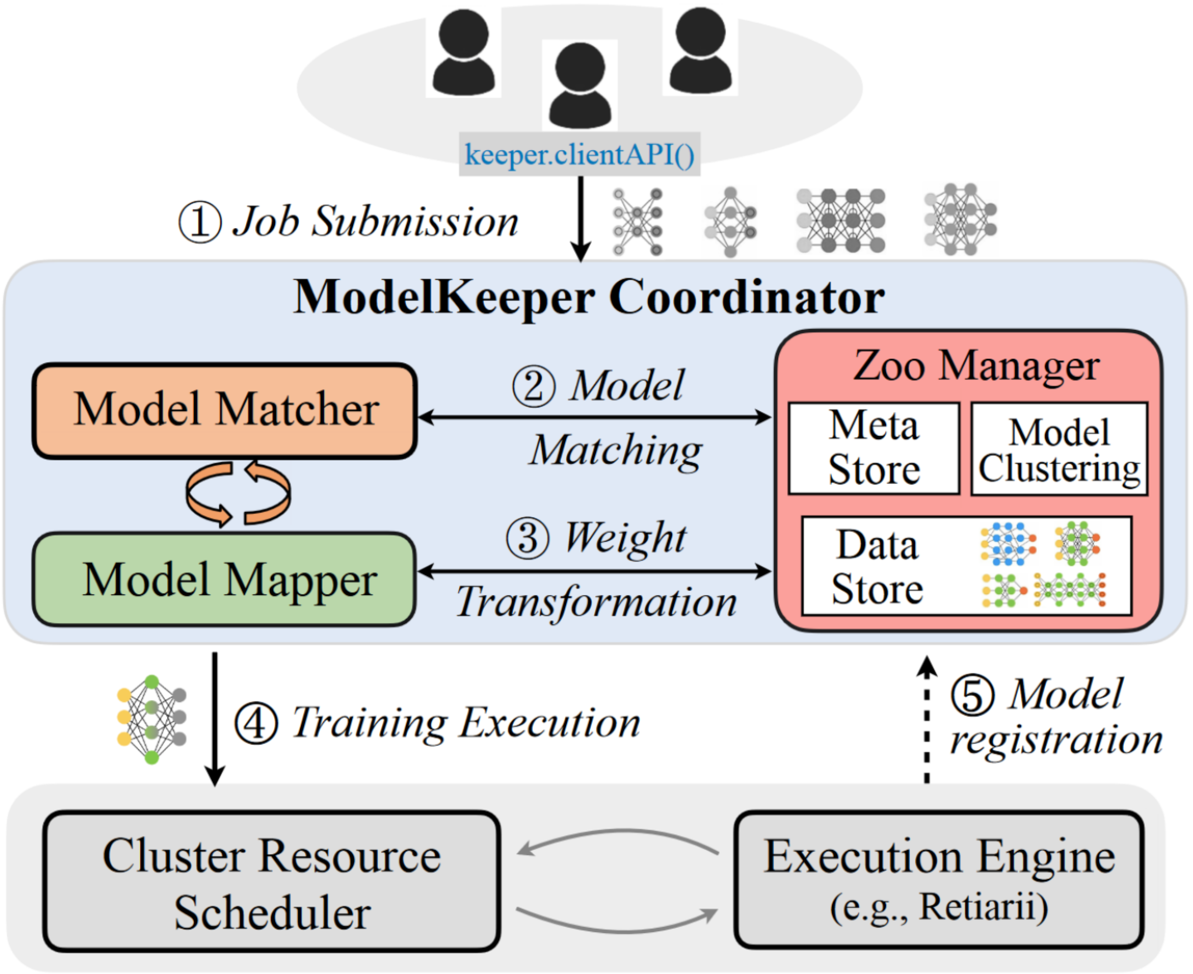


Figure 5: ***ModelKeeper architecture. It can run as a cluster-wide service to serve different users and/or frameworks.***

System Overview

- *Job Submission*
- *Model Matching*
- *Weight Transformation*
- *Training Execution*
- *Model registration*

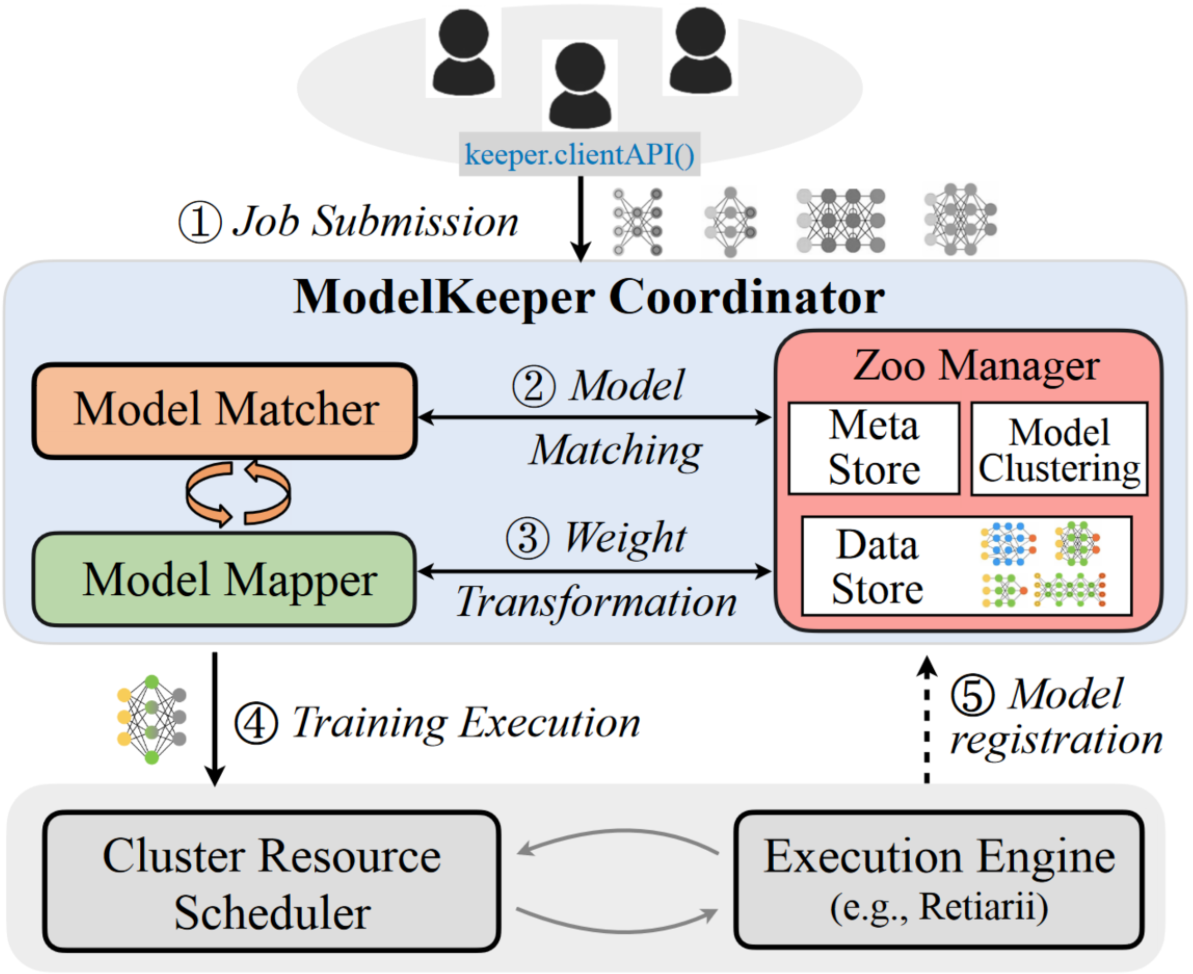


Figure 5: ***ModelKeeper architecture. It can run as a cluster-wide service to serve different users and/or frameworks.***

Implementation

- ***Code***

```
from modelkeeper import ModelKeeperCoordinator
keeper_service = ModelKeeperCoordinator(config)
keeper_service.start()
```

```
from modelkeeper import ModelKeeperClient

def training_with_keeper(model, dataset):
    # Create client session to keeper coordinator
    keeper_client = ModelKeeperClient(coordinator_ip)
    warmed_model, meta = keeper_client.query_for_model(
                model, meta={'data': 'Flowers102',
                'task':'classification', 'tags': None})

    acc = train(warmed_model, dataset) # Training starts

    # Register model to ModelKeeper when training ends
    keeper_client.register_model(warmed_model,
            meta={'data': 'Flowers102', 'accuracy': acc,
            'task': 'classification', 'tags': None})
    keeper_client.stop()
```

Figure 6: ***Code snippet of ModelKeeper client service APIs.***

Model Matcher

- ***Dynamic programming-like heuristics***
  - ***SKIP_CHILD***
  - ***SKIP_PARENT***
  - ***MATCH***

$$\mathbb{M}(i,j) = \max_{k \in parent(i)} \begin{cases} \mathbb{M}(k, j_{parent}) + MATCH(k, j_{parent}) & (1) \\ \mathbb{M}(k, j) + SKIP\_PARENT & (2) \\ \mathbb{M}(i, j_{parent}) + SKIP\_CHILD & (3) \end{cases}$$

$$MATCH(i,j) = \frac{\prod_{dim=1} \min(dim(i), dim(j))}{\prod_{dim=1} \max(dim(i), dim(j))} \quad (\in [0,1])$$

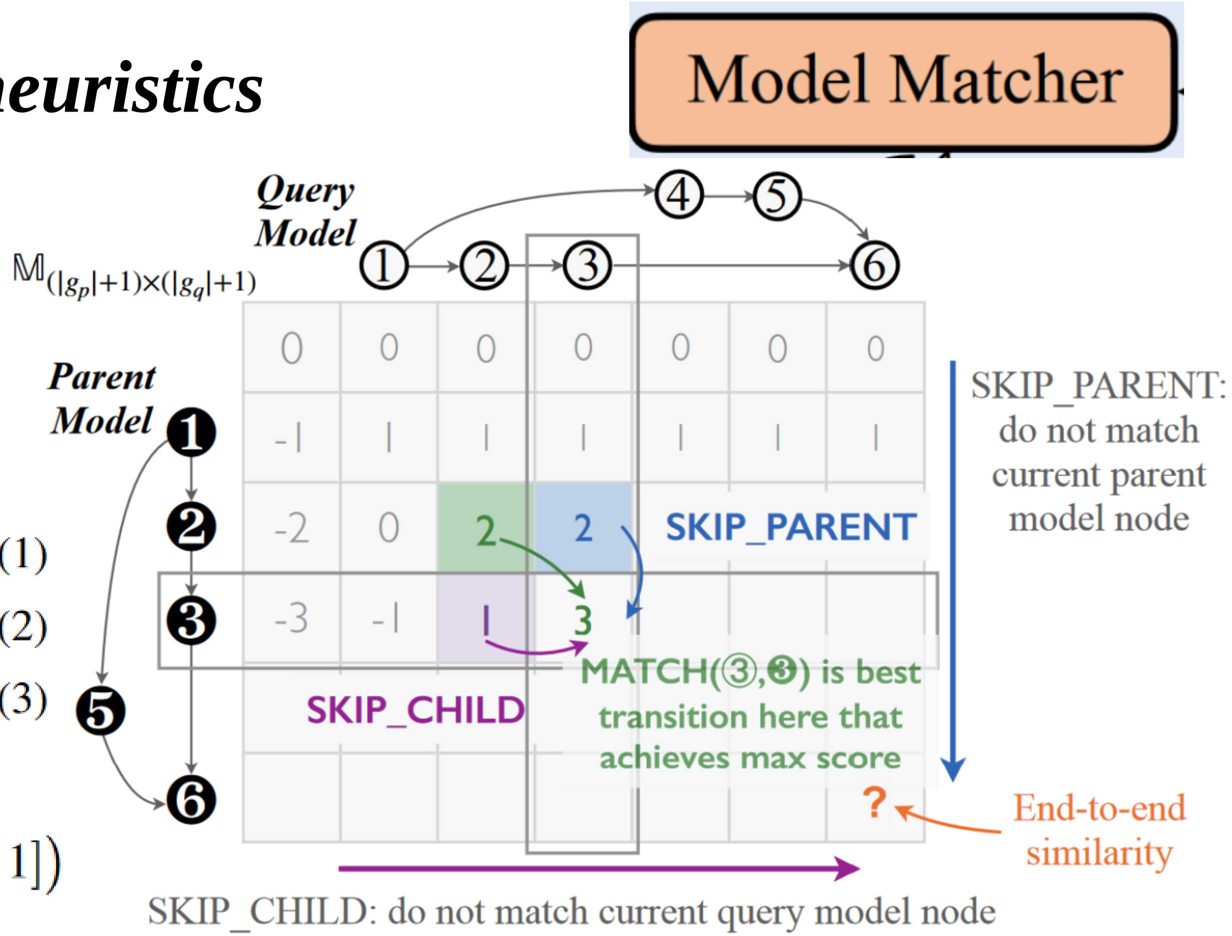


Figure 7: ***ModelKeeper relies on dynamic programming-like heuristics to measure graph-level model architectural similarity.***

Model Matcher

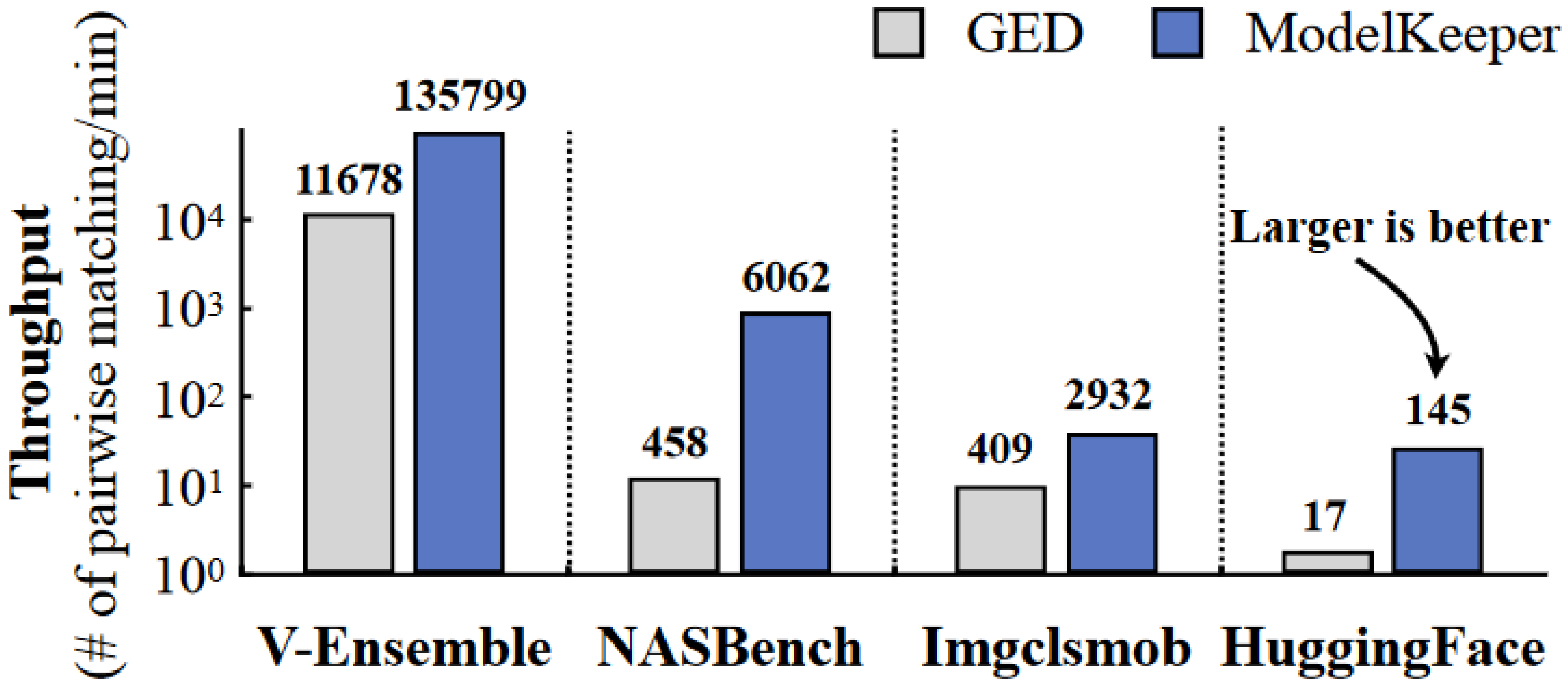


Figure 8: *Keeper is order-of-magnitude more scalable than existing GED. V-Ensemble is a model zoo for ensemble training (§6.1).*

Model Mapper

- **Choose parent model**

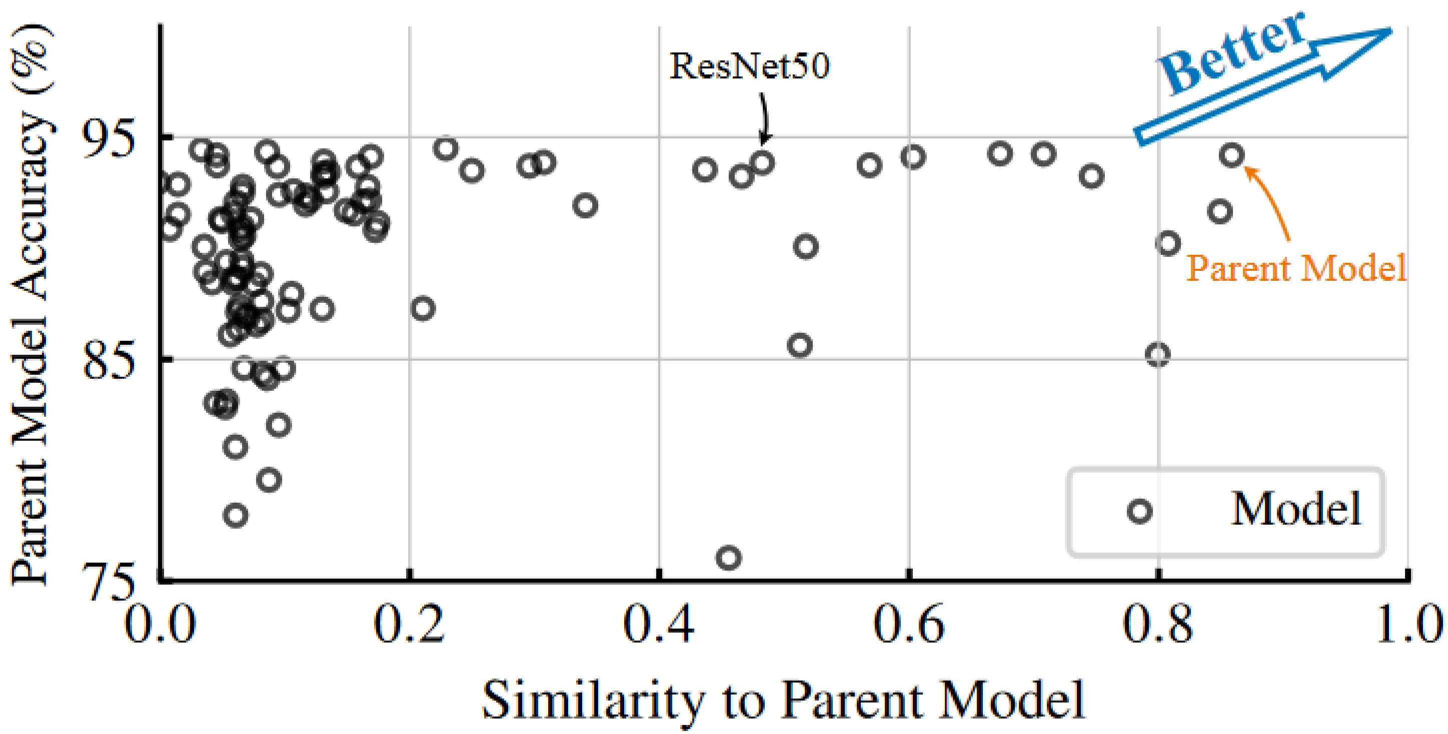


Figure 9: ***Models vary in accuracy and architecture (Imgclsmob zoo). We measure their similarity w.r.t. ResNet101, and prefer to transform a parent model with better similarity and accuracy.***

Model Mapper

- **Transform Operator:** ***Width and depth.***

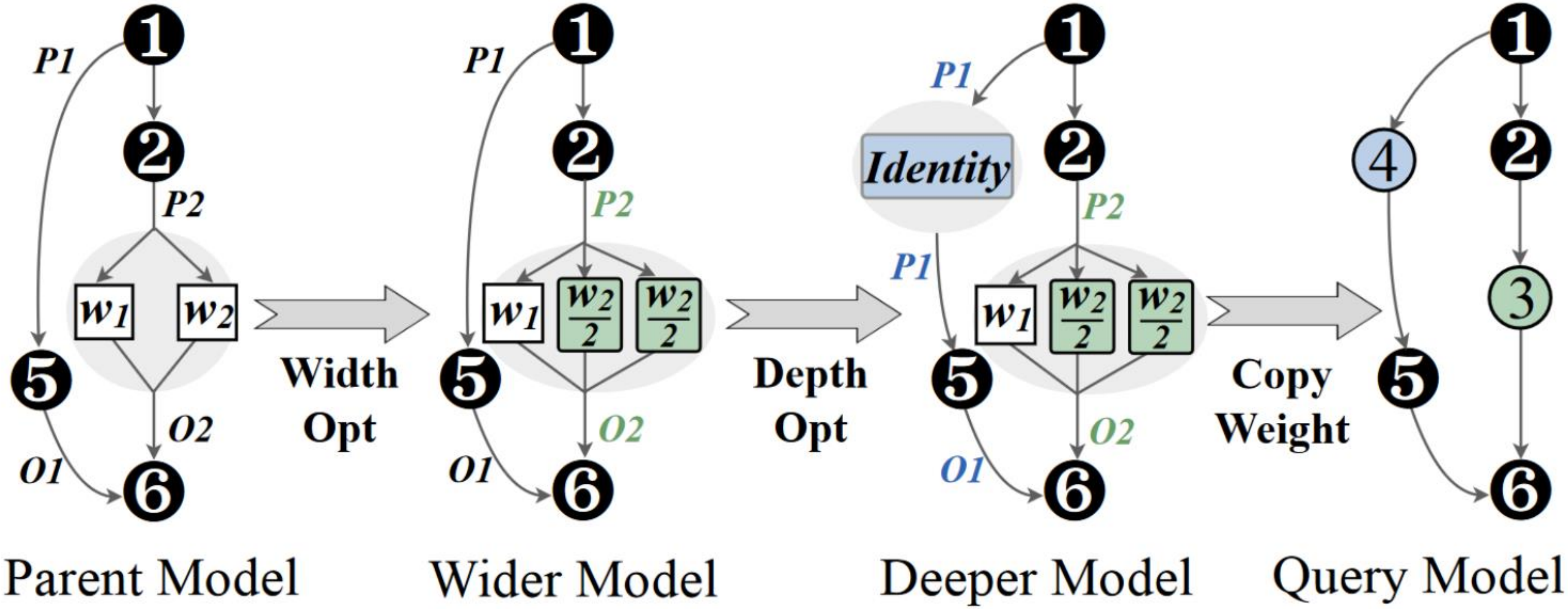


Figure 10: ***Width and depth operator to transform parent model.***

Model Mapper

- **Algorithm - *Select the parent model to transform.***

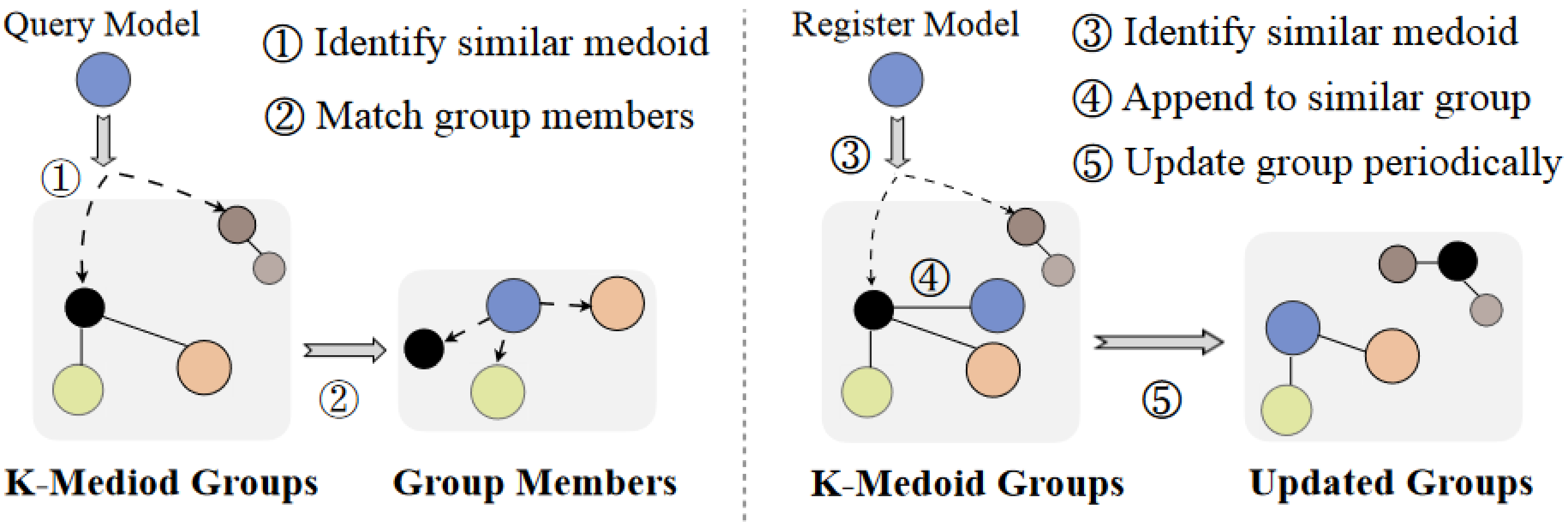
Input: Query model q, Model Zoo M
Output: Warmup Model Weights

```
NumOfBuckets B = 10                                   ▷ Model similarity ∈ [0, 1]
Function GetModelSim (Query q, Models M)
    /* Structure-aware matching for model similarity. */
    topo_query_tensors = SortByTopo(q)
    model_similarity = { }
    for model m ∈ M do
        similarity_table = zeros(|g_m|+1, |g_q|+1)
        for tensor i ∈ CachedModelTopo(m) do
            for tensor j ∈ topo_query_tensors do
                /* Enumerate and merge intersection. */
                similarity_table[i][j] = Equation (1-3)
        model_similarity[m] = similarity_table[|g_m|][|g_q|]
    return model_similarity
Function QueryForModel (Query q, Model Zoo M)
    /* Bucket models in terms of similarities. Pick the model in
       the top-similar bucket with the best performance. */
    model_similarity = GetModelSim(q, M)
    top_similar_bucket =
      BucketBySimilarity(model_similarity, B).first
    for model ∈ top_similar_bucket do
        if model.perf > best_parent.perf then
            best_parent = model
    /* Perform width and depth weight transformation */
    warmup_weights = TransWeight(best_parent, q)
    return warmup_weights
```

Algorithm 1: Select the parent model to transform.

Zoo Manager

- ***Two-Stage Hierarchical Model Matching***



(a) When query model arrives (b) When new zoo model joins

Figure 11: ***Matcher clusters models into groups to reduce the search space, and then performs model matching within groups.***

Zoo Manager

- ***Capping Zoo Size***

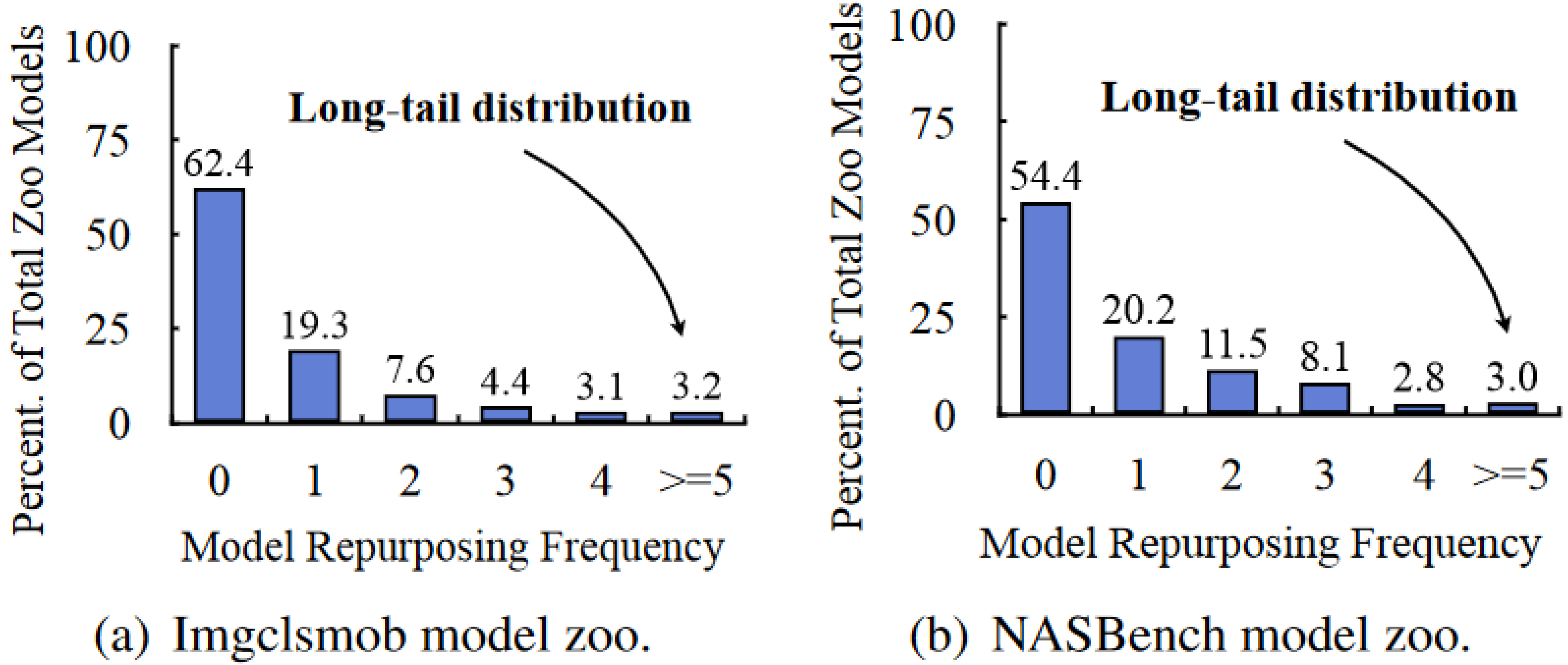


Figure 13: ***A few zoo models are more frequently repurposed as the parent by Keeper. Numbers are from our evaluations (§6.2).***

Experiment

- ***Training Improvements***

Category	Task	Workload	# of Models	Dataset	Avg. Time Improvement	Avg. Acc. Difference
Exploratory Training	Grid Search NAS	NASBench [24]	1,000	CIFAR-100	2.9×	0.39%
	Evolution NAS				2.4×	0.38%
	AK-Bayesian NAS [41]	AutoKeras Zoo [41]	500		4.3×	0.31%
General Training	Image Classification	Imgclsmob [10]	389	Flowers102 [54]	2.8×	0.23%
		Imgclsmob-Small	179	CIFAR-10	2.1×	0.02%
				CIFAR-100	1.6×	0.18%
				ImageNet32 [19]	1.3×	0.03%
	Ensemble Training	V-Ensemble [65]	104	CIFAR-100	1.7×	0.08%
	Language Modeling	HuggingFace [2]	69	WikiText-103 [47]	1.8×	-0.13 perplexity

Table 1: ***Summary of improvements. ModelKeeper improves training execution time without accuracy drop, by reducing the amount of training needed (i.e., GPU Saving). Accuracy difference is defined by Acc.(Keeper) - Acc.(Baseline), and smaller perplexity is better.***

Experiment

- ***components***

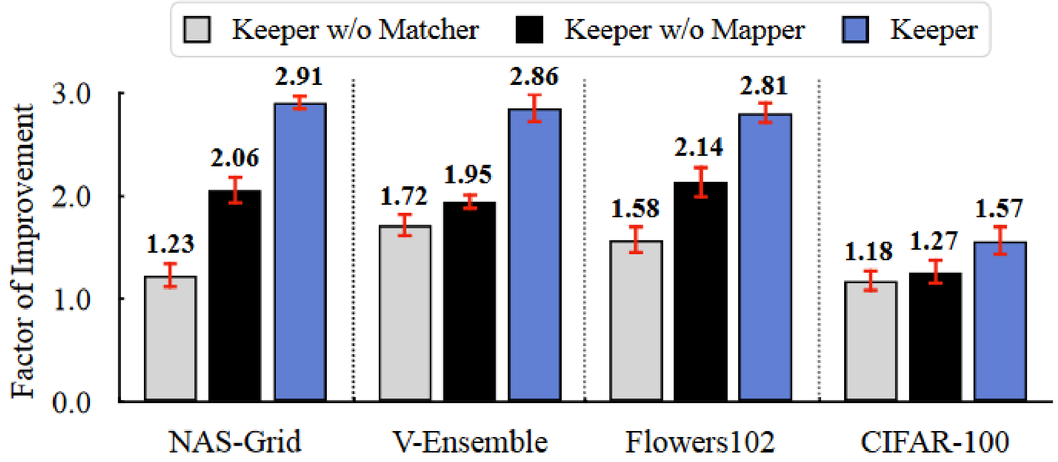


Figure 16: ***Breakdown of Keeper components.***

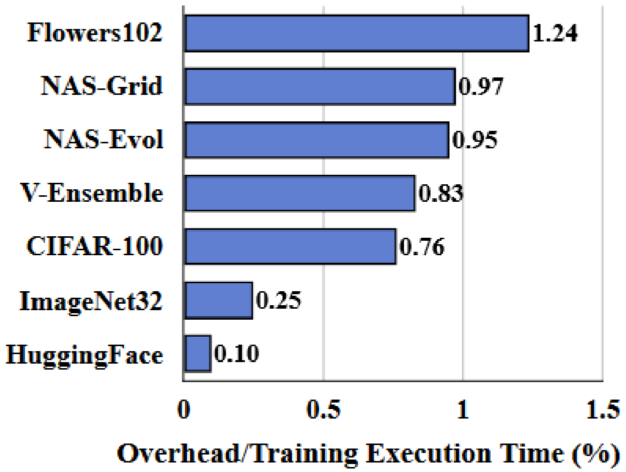


Figure 18: ***Keeper introduces negligible overhead.***

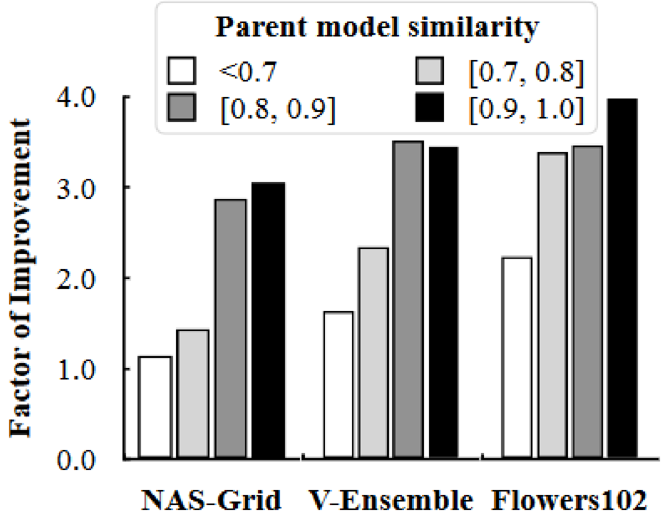


Figure 17: ***Faster training with higher model similarity.***